AGH UNIVERSITY OF KRAKOW

FIELD OF SCIENCE ENGINEERING AND TECHNOLOGY

SCIENTIFIC DISCIPLINE INFORMATION AND COMMUNICATION TECHNOLOGY

DOCTORAL DISSERTATION

Towards Lifelong Anomaly Detection in Challenging Scenarios and its Application in Cybersecurity

Author: Kamil Faber

First supervisor: dr hab. inż. Bartłomiej Śnieżyński, prof. AGH
Auxiliary supervisor: dr Roberto Corizzo

Completed in: Faculty of Computer Science

Kraków, 2024

I am very grateful for many people whose involvement and support helped me in preparing this thesis. I would like to thank especially my supervisors, Prof. Bartłomiej Śnieżyński and Prof. Roberto Corizzo, for leading me throughout this process and for everything I have learned from them.
I would also like to thank my parents, brother, and friends, who provided me with support and encouragement during this journey.

Abstract

The ability to detect anomalies is a critical tool that brings significant advantages in many real-world domains, such as monitoring cyber-physical systems, human conditions, and network traffic. Many of these applications are characterized by evolving and dynamic environments to which the model has to adapt. Current research in anomaly detection has been split between offline learning, which deals with static datasets, and online learning, which focuses on adaptation to the most recent data in evolving conditions.

On the other hand, lifelong learning is gaining attention as a way to build comprehensive models that can successfully operate in evolving real-world scenarios. In contrast to online learning, lifelong learning focuses not only on adapting to the most recent data but also on retaining previously acquired knowledge. Although this aspect could be beneficial to build effective and robust anomaly detection models, lifelong learning research primarily focuses on image classification and reinforcement learning domains. The limited scope addressed by lifelong learning works thus far creates a gap in understanding whether such techniques and capabilities can be fruitfully exploited in anomaly detection. Moreover, anomaly detection introduces distinct challenges, such as the continuous evolution of what is considered "normal" and the scarce availability of anomaly instances, which significantly differ from those encountered in lifelong image classification and reinforcement learning.

In this dissertation, we address this gap by exploring, motivating, and discussing lifelong anomaly detection and creating methods working in challenging scenarios with a particular focus on cybersecurity applications. More specifically, we provide foundations with regard to scenarios, strategies, and metrics. We also showcase the benefits of simultaneous adaptation and knowledge retention provided by lifelong learning. Moreover, we design VLAD – a method that answers the problem of challenging scenarios in which the model has to recognize a change in data that indicates a need for a model to adapt. As a support tool for this problem, we create a LIFEWATCH algorithm capable of detecting changes and recognizing whether new data is something new or a reoccurrence of an already encountered situation. We also address the problem of contaminated training data by devising an active lifelong anomaly detection framework that leverages the external oracle to mitigate the contamination. Finally, we present our framework for distributed lifelong intrusion detection, which showcases how collaborative lifelong learning can improve the detection abilities of distributed intrusion detection systems.

This dissertation is based on 8 published and thematically related scientific papers and consists of two parts. The first part starts with motivation and background and follows with an extended summary of all contributions. Then, the second part presents all the papers included in the cycle.

Streszczenie

Umiejętność wykrywania anomalii jest kluczowym narzędziem przynoszącym znaczące korzyści w wielu zastosowaniach, takich jak monitorowanie systemów cyberfizycznych, stanu zdrowotnego ludzi oraz ruchu w sieci. Wiele z tych zastosowań charakteryzuje się ewoluującym i dynamicznym środowiskiem, do którego model musi się dostosowywać. Aktualne badania w zakresie wykrywania anomalii zostały podzielone na dwa kierunki: uczenie offline, które radzi sobie ze statycznymi zestawami danych; oraz uczenie online, które koncentruje się na adaptacji do najnowszych danych w scenariuszach ze zmieniającymi się warunkami.

W ostatnich latach pojawił się nowy paradygmat uczenia maszynowego – lifelong learning, który zdobywa uwagę jako sposób na budowanie kompleksowych modeli, które mogą skutecznie działać w ewoluujących rzeczywistych scenariuszach. W przeciwieństwie do uczenia online, lifelong learning koncentruje się nie tylko na adaptacji do najnowszych danych, ale także na zachowaniu wcześniej nabytej wiedzy. Chociaż ten aspekt mógłby być korzystny dla budowania skutecznych modeli wykrywania anomalii, badania nad lifelong learning głównie koncentrują się na dziedzinach klasyfikacji obrazów i uczenia ze wzmocnieniem. Takie ograniczenie zakresu istniejących badań tworzy lukę w wiedzy na temat wykorzystania takich technik w wykrywaniu anomalii. Co więcej, wykrywanie anomalii wprowadza specyficzne dla siebie wyzwania, takie jak ciągła ewolucja danych uważanych za „normalne" oraz niewielka liczba anomalii dostępnych w zbiorach treningowych. Te wyzwania znacznie odbiegają od tych napotkanych w klasyfikacji obrazów i uczeniu ze wzmocnieniem przez całe życie.

W tej dysertacji zajmujemy się tą luką, prowadząc badania w zakresie połączenia wykrywania anomalii oraz lifelong learningu i tworząc algorytmy działające w trudnych scenariuszach, ze szczególnym uwzględnieniem zastosowań w cyberbezpieczeństwie. Formułujemy podstawowe pojęcia w zakresie scenariuszy, strategii i metryk. Prezentujemy także korzyści płynące z jednoczesnej adaptacji i zachowania wiedzy, które zapewnia lifelong learning. Ponadto prezentujemy oryginalną metodę VLAD, będącą odpowiedzią na problem trudnych scenariuszy, w których algorytm musi rozpoznać zmianę w danych, wskazującą na potrzebę adaptacji. Jako narzędzie wspomagające ten problem proponujemy algorytm LIFEWATCH, który jest zdolny do wykrywania zmian i rozpoznawania, czy napotkane dane są zupełnie nową sytuacją, czy też ponownym wystąpieniem już wcześniej napotkanej. Adresujemy również problem zanieczyszczonych danych uczących, tworząc metodę redukcji problemu zanieczyszczenia poprzez uczenie aktywne. Na koniec prezentujemy propozycję algorytmu wykorzystującego współdzielenie wiedzy oraz lifelong learning w zadaniu wykrywania włamań w środowisku rozproszonym.

Ta dysertacja składa się z cyklu 8 tematycznie powiązanych publikacji. Pierwsza część rozprawy rozpoczyna się od wprowadzenia i motywacji dla naszych badań, a następnie zawiera opis aktualnego stanu wiedzy oraz rozszerzone podsumowanie wszystkich publikacji. Druga część obejmuje wszystkie artykuły włączone do cyklu.

Contents

I Part I: Extended summary **1**

1 Introduction **3**
1.1 Motivation of the research . . . 3
1.2 Thesis of the dissertation . . . 5
1.3 Set of publications . . . 6
1.4 Structure and contributions . . . 7

2 Background **9**
2.1 Lifelong learning . . . 9
2.2 Anomaly detection . . . 11
2.3 Lifelong anomaly detection . . . 13

3 Lifelong anomaly detection: challenges, perspectives and insights **15**
3.1 Challenges of lifelong anomaly detection . . . 15
3.2 Lifelong learning scenarios . . . 18
3.3 Evaluation protocol and metrics . . . 19
3.4 Experimental results overview . . . 21

4 Benchmarking lifelong learning methods **25**
4.1 M2I and I2M benchmarks . . . 25
4.2 Experimental results overview . . . 26

5 Change point detection for lifelong learning **29**
5.1 WATCH . . . 30
5.2 LIFEWATCH – lifelong change point detection . . . 31

6 VLAD – concept-agnostic lifelong anomaly detection **37**
6.1 Method overview . . . 37
6.2 Experimental results overview . . . 42

7 Active lifelong anomaly detection **45**
7.1 Active lifelong anomaly detection framework . . . 45
7.2 Experimental results overview . . . 46

8 Lifelong distributed collaborative intrusion detection **51**
8.1 Autoencoder-based IDS for cloud and mobile devices . . . 52
8.2 Collaborative framework for distributed lifelong intrusion detection . . . 53

9 Conclusions **59**

II Part II: Contributions **61**

10 Lifelong Learning for Anomaly Detection: New Challenges, Perspectives, and Insights **63**

11 From MNIST to ImageNet and Back: Benchmarking Continual Curriculum Learning **81**

12 WATCH: Wasserstein Change Point Detection for High-Dimensional Time Series Data **111**

13 LIFEWATCH: Lifelong Wasserstein Change Point Detection **123**

14 VLAD: Task-agnostic VAE-based lifelong anomaly detection **133**

15 Active Lifelong Anomaly Detection with Experience Replay **161**

16 Distributed Continual Intrusion Detection: A Collaborative Replay Framework **173**

17 Autoencoder-based IDS for cloud and mobile devices **183**

List of Figures **193**

List of Tables **194**

Bibliography **195**

Part I

Part I: Extended summary

Chapter 1

Introduction

1.1 Motivation of the research

Machine learning (ML) describes building models that can learn how to solve specific tasks leveraging training data [55]. Machine learning systems operating in the real world are exposed to continuous and evolving information streams. Therefore, to keep their efficacy, they must learn and adapt while keeping the ability to remember and solve multiple tasks learned across many data distributions [87]. This issue is particularly essential in applications where the environment is dynamically evolving, providing the machine learning system with new tasks or evolved versions of already known tasks. For instance, an autonomous self-driving taxi embedded in a dynamic real-world environment must learn from its experience, progressively acquiring knowledge about evolving traffic conditions in different situations (days, weather, special events) while not losing previously learned skills [57]. Another example could be a classification system that needs to learn new classes progressively during its lifetime, without the possibility of doing the full retraining each time new data is presented [61].

The ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences is referred to as continual or lifelong learning [103, 50]. Lifelong learning requires the system to have two crucial capabilities: i) ability to continuously learn and adapt; ii) ability to retain previously learned knowledge. Lifelong learning research usually operates on scenarios built as sequences of tasks, where a single task represents a piece of knowledge to learn (for example, a set of new classes to learn or a new presentation of already-known classes) [105]. The main lifelong learning challenge is related to the natural inclination of machine learning models to forget the previous knowledge when learning new tasks [30]. This phenomenon is known as forgetting, and, in its most harsh form, it is also referred to as catastrophic forgetting [87]. Catastrophic forgetting leads to severe performance decreases in the affected model and, in extreme cases, complete overwriting of the old knowledge by the newly learned tasks [61]. Consequently, lifelong learning methods focus on finding countermeasures to forgetting and balancing adaptation and knowledge retention [87].

We have observed an emerging interest in lifelong learning research in the last few years. However, most lifelong learning works so far have been directed toward image classification, reinforcement learning, and object recognition [87, 1, 88, 13]. One of the missing directions is anomaly detection, even though the phenomenon of dynamic evolving environments is commonly observed there as well. Moreover, anomaly detection brings challenges currently underexplored in lifelong learning research. First, current methods usually assume the availability of information about changing tasks or the identity of the currently processed task, but this assumption is impractical in many anomaly detection applications [26]. Second, the anomaly detection methods often encounter the issue of contamination in the training data [110].

Anomaly detection aims to find data instances that represent a deviation from normal conditions and, as such, are anomalous [94]. Identification of anomalous behavior is essential in many disciplines and real-world applications. For example, identifying faults and malfunctions in cyber-physical systems, such as water treatment plants or smart grids, allows the operator to take the appropriate actions to keep the system functional [27, 6]. Another example could be detecting anomalies in the results of medical tests, such as an electrocardiogram or magnetic resonance imaging, where identifying deviations from normal may help doctors make more accurate and quick diagnoses [68]. One of the highly relevant and impactful domains that intensively leverage anomaly detection is cybersecurity. Example applications include detecting intrusions via modeling the normal behavior of the system [81], identifying anomalous behavior of malware applications [47], and suspicious patterns indicating bank fraud [52].

Anomaly detection models often have to work in environments that continuously evolve. As a consequence, models become outdated and require updates to keep their detection efficacy [3]. Most anomaly detection methods work either in an offline (batch) or online (stream) manner. Model updates are implemented either as a full retraining stage (batch models) after detecting a concept drift or an online adaptation (stream models) [40]. Both approaches are considered practical and bring much value in specific domains. Offline anomaly detection has demonstrated its potential in various contexts where historical data accumulation and subsequent analysis are feasible. Notable applications include post-incident analysis [48], breast cancer detection [53], and gravitational waves detection [29]. Unfortunately, offline models are insufficient in dynamic and evolving applications, as they do not adapt to changes in the normal class over time. This shortcoming is partially addressed by online anomaly detection methods that are capable of adjusting to the most recent data. This behavior is considered sufficient in many dynamic learning settings such as crowd anomaly detection [39] and fatigue detection [66]. Even though updating the models, either in a retraining stage or in an online manner, allows them to adapt to evolving conditions, it also may have the side effect of gradual forgetting of past knowledge.

Forgetting is a well-known phenomenon in online learning and data streams. In some scenarios, it is regarded as a positive feature that allows models to focus only on the most recent characteristic of a task to handle concept drift [44]. One example is the crowd anomaly detection problem [39], where only the people present in the currently monitored environment (i.e., the most recent data) are considered relevant to predicting anomalies within that environment at the specific time.

Nevertheless, forgetting is a shortcoming in domains where handling multiple recurring tasks is essential [61]. Therefore, adopting lifelong learning may considerably benefit many real-world anomaly detection applications where knowledge retention is as important as adaptation. One example is intrusion detection in a cloud environment, in which the model has to deal with a dynamic environment where multiple cloud instances are added or removed over time. Such instances have diverse characteristics of normal behavior depending, for example, on active services and user interactions. The intrusion detection system must be capable of adjusting and detecting anomalous behavior in new cloud instances while maintaining its performance when analyzing traffic from already known cloud instances. Another example could be monitoring human conditions to detect harmful states, which requires the model to navigate through many human activities, each defining a unique characterization of the normal class. New lifestyle habits introduce new activities that can be considered as new tasks the model has to learn (e.g., jogging, which involves a high heart rate but is not an anomalous behavior). The inability to remember past activities can lead to practical disadvantages and detection inaccuracies, which we explore in further chapters. It is noteworthy that the notion of a task in anomaly detection differs from the conventional understanding present in other lifelong learning domains. In this domain, new challenges and conditions are brought as evolving characterization of the "normal" class, which needs to be learned to detect anomalies. We discuss this problem more in detail in further chapters, and, in Chapter 3, we introduce the term "concept" describing a task in a lifelong anomaly detection setting.

The aforementioned examples are marked by distinct challenges, such as the presence of both: i) evolving emerging conditions that require model adaptation; and ii) recurring conditions that require the ability to preserve and effectively apply knowledge learned before. Adopting lifelong anomaly detection in these domains thus promises to enhance the efficacy and robustness of models in managing this duality.

To sum up, even though lifelong learning gathers a lot of interest, research focuses mainly on image classification, reinforcement learning, and object recognition while not paying much attention to anomaly detection. At the same time, anomaly detection methods found a way to deal with adaptation to new knowledge, but there is very little focus on knowledge retention, even though combining those two capabilities may bring a lot of contribution to many real-world applications. Moreover, the application of lifelong learning to anomaly detection needs to consider the specifics of anomaly detection, such as no clear separation between changing conditions and contamination of training data. These facts create a research gap we attempt to fill in this dissertation.

1.2 Thesis of the dissertation

To address the research gap described above, we formulate the following thesis:

It is possible to leverage lifelong learning in anomaly detection to provide robust detection performance in environments with evolving and recurring conditions in challenging scenarios, typical for real-world domains such as cybersecurity.

Verification of the thesis requires solving three specific problems: i) integration of lifelong learning and anomaly detection; ii) devising methods capable of working in challenging conditions; iii) showcasing the applicability of lifelong anomaly detection to cybersecurity. The rest of this work aims to confirm the formulated research thesis by creating methods to answer the problems mentioned above and experimentally evaluating their quality. The main contributions of this dissertation that support the aforementioned thesis can be summarized as follows:

- **C1**: Bridging the gap between anomaly detection and lifelong learning (see Chapter 3).
  - Showcasing the benefits of lifelong anomaly detection over conventional anomaly detection along with challenges brought by the need to simultaneously adapt and retain knowledge.
  - Introducing the idea of "concept" describing a single task in lifelong anomaly detection settings.
  - A characterization of lifelong anomaly detection scenarios and scenario generation algorithm that can be applied to any anomaly detection dataset, along with its open-source implementation.
  - Describing the evaluation protocol and metrics, aiming to lay the foundation for future lifelong anomaly detection research.
  - Evaluation of anomaly detection methods in lifelong scenarios, emphasizing challenges brought by lifelong learning setting.
- **C2**: Designing lifelong anomaly detection methods suitable for working in challenging conditions along with a supporting lifelong change point detection method.
  - `VLAD`: Variational Autoencoder-based lifelong anomaly detection method providing the desired lifelong learning features in challenging scenarios with no information about changing conditions, where the model has to discover such changes on its own (see Chapter 14).
  - `LIFEWATCH`: A lifelong change point detection that has the ability to model multiple data distributions and recognize whether the change means the appearance of a new task or a recurring of a previously seen task (see Chapter 13). Such ability is essential in methods such as `VLAD` that

address multiple tasks in scenarios with no information about the changes. LIFEWATCH is built on top of WATCH, our change point detection method designed to work with high-dimensional data (see Chapter 12).

  - Active lifelong anomaly detection framework where active learning module improves the performance of replay-based methods by leveraging user feedback to clear replay buffers in scenarios where training data is contaminated (see Chapter 15).
  - Evaluation of existing lifelong image classification methods in new challenging benchmarks (see Chapter 4). While this work is not directly set in anomaly detection settings, it showcases that replay methods provide robust knowledge retention capabilities, outperforming even more complex methods. This conclusion supports our decision to create replay-based methods and frameworks in our other lifelong anomaly detection.

- **C3:** Showcasing that lifelong anomaly detection in cybersecurity represents a promising avenue for enhancing system resilience and adaptability.
  - Evaluating our lifelong anomaly detection methods on intrusion detection datasets as one of the included challenging domains.
  - Proposing a Collaborative Replay Framework that enhances the lifelong anomaly detection capabilities with efficient knowledge sharing, addressing the more challenging settings of lifelong distributed intrusion detection (see Chapter 16). This framework is built on top of our work showcasing the advantages of collaborative data sharing in intrusion detection (see Chapter 17).

1.3 Set of publications

The dissertation consists of a *a set of published and thematically related scientific papers*. We include the following papers:

1. **Kamil Faber**, R. Corizzo, B. Sniezynski and N. Japkowicz, "Lifelong Continual Learning for Anomaly Detection: New Challenges, Perspectives, and Insights", IEEE Access, vol. 12, pp. 41364-41380, 2024; doi: 10.1109/ACCESS.2024.3377690 (IF: 3.9; 100 points; see an overview in Chapter 3 and full text in Chapter 10; primarily supporting Contribution **C1**).

2. **Kamil Faber**, D. Zurek, M. Pietron, N. Japkowicz, A. Vergari, R. Corizzo, "From MNIST to ImageNet and Back: Benchmarking Continual Curriculum Learning", Machine Learning, **accepted, in publication**; (IF 7.5; 140 points; see Chapter 11; primarily supporting Contribution **C2**).

3. **Kamil Faber**, R. Corizzo, B. Sniezynski, M. Baron and N. Japkowicz, "WATCH: Wasserstein Change Point Detection for High-Dimensional Time Series Data", 2021 IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, 2021, pp. 4450-4459, doi: 10.1109/BigData52589.2021.9671962 (CORE B-bank; 70 points; see Chapter 12; primarily supporting Contribution **C2**)).

4. **Kamil Faber**, R. Corizzo, B. Sniezynski, M. Baron and N. Japkowicz, "LIFEWATCH: Lifelong Wasserstein Change Point Detection", 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 2022, pp. 1-8, doi: 10.1109/IJCNN55064.2022.9892891 (CORE B-rank; 140 points; see Chapter 13; primarily supporting Contribution **C2**)).

5. **Kamil Faber**, R. Corizzo, B. Sniezynski and N. Japkowicz, "VLAD: Task-agnostic VAE-based lifelong anomaly detection", Neural Networks, vol. 165, pp. 248-273, 2023, doi:

10.1016/j.neunet.2023.05.032 (IF: 7.8; 200 points; see Chapter 14; primarily supporting Contribution **C2**).

6. **Kamil Faber**, R. Corizzo, B. Sniezynski and N. Japkowicz, "Active Lifelong Anomaly Detection with Experience Replay", 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), Shenzhen, China, 2022, pp. 1-10, doi: 10.1109/DSAA54385.2022.10032405 (CORE A-rank; 140 points; see Chapter 15; primarily supporting Contribution **C2**)).

7. **Kamil Faber**, B. Sniezynski and R. Corizzo, "Distributed Continual Intrusion Detection: A Collaborative Replay Framework", 2023 IEEE International Conference on Big Data (BigData), Sorrento, Italy, 2023 pp. 3255-3263, doi: 10.1109/BigData59044.2023.10386211 (CORE B-bank; 70 points; see Chapter 16; primarily supporting Contribution **C3**)).

8. **Kamil Faber**, L. Faber and B. Sniezynski, "Autoencoder-based IDS for cloud and mobile devices", 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), Melbourne, Australia, 2021, pp. 728-736, doi: 10.1109/CCGrid51090.2021.00088 (CORE A-rank; 140 points; see Chapter 17; primarily supporting Contribution **C3**).

1.4 Structure and contributions

This dissertation is structured into two main parts. We start with an extended summary of the contributions, showcasing the overview of our methodology and experimental evaluation. Chapter 3 focuses on the integration of lifelong learning into anomaly detection, providing foundations in terms of scenarios, protocols, and metrics. Chapter 4 evaluates many existing lifelong image classification methodologies, demonstrating the efficacy of the replay strategy and, therefore, providing the rationale behind incorporating the replay mechanism into our lifelong anomaly detection research. Chapter 5 introduces `WATCH`, our change point detection method capable of detecting changes in high-dimensional data, and its extension `LIFEWATCH`, our lifelong change point detection method with the ability to yield a distinction between new and recurring tasks. Chapter 6 presents `VLAD` – our lifelong anomaly detection method capable of working in challenging scenarios without knowledge of changes requiring model adaptation. Then, Chapter 7 describes our active lifelong anomaly detection framework that effectively tackles the problem of contamination in the training data. Chapter 8 brings the principles of lifelong anomaly detection to the complex problem of distributed intrusion detection. This chapter describes our proposed collaborative lifelong learning framework for enhancing the robustness and efficacy of cybersecurity measures. It also introduces our preliminary work showcasing the advantages of knowledge sharing in intrusion detection. Finally, Chapter 9 concludes the dissertation, summarizing the key findings and contributions. The second part of the dissertation (Chapters 10-17) includes all papers being part of the cycle of publications, accompanied by a description of contributions.

Chapter 2

Background

In this section, we provide a brief introduction to lifelong learning and anomaly detection. We present the requirements for lifelong learning methods and introduce existing lifelong learning scenarios and strategies in already explored domains. Moreover, we provide a background on anomaly detection and its importance in cybersecurity. We also point out research gaps outlining the problems addressed in this thesis.

2.1 Lifelong learning

The concept of "lifelong learning", also known as "continual learning", describes the ability of a machine learning method to persistently acquire new knowledge over time while preserving knowledge learned from previous experiences [103, 50]. A significant obstacle in lifelong learning research is the phenomenon of "catastrophic forgetting." This issue occurs when a model loses previously acquired knowledge upon learning new tasks [61]. To effectively navigate this challenge, a learning system must efficiently manage the trade-off between adaptability and knowledge retention. This balancing act, also known as the "stability-plasticity dilemma", describes the need for a model to be sufficiently adaptable to incorporate new learning (plasticity) while remaining stable enough to preserve existing knowledge (stability) [87]. Addressing this dilemma is pivotal in the development of robust lifelong learning methods capable of functioning effectively in dynamic and unpredictable real-world settings.

We observe emerging interest in lifelong learning with an extensive number of works published in areas such as computer vision (especially image classification) and reinforcement learning [87, 1, 88, 13]. There are also new trends, such as bridging active learning with open world learning [83], applications to robotics [85], as well as learning with access to just a tiny portion of a data stream [77].

Scenario types

Task labels and task boundaries are often utilized to differentiate lifelong learning scenarios [105]. A task label uniquely identifies a specific task being processed during both the training and inference phases. If this information is available, the model can adapt its strategies to the particular task. On the other hand, task boundaries present information about the start of a new task during training but do not present information about which task it is. If only task labels are available, the model has to handle multiple tasks without the guidance of specific task labels, relying solely on the data to provide accurate inferences. Another dimension differentiating types of scenarios is whether new tasks bring new problems to solve (for example, new image classes) or a new representation of already encountered problems (for example, different types of images from already learned classes). Following these three characteristics, the authors of [105] characterize three types of lifelong learning scenarios:

- Task-incremental – A model has to learn to solve new problems while having access to task labels.
- Class-incremental – A model has to learn to solve new problems while being informed about task boundaries.
- Domain-incremental – A model has to learn new representations of already encountered problems while having access to task boundaries.

Let us consider the viral example of handwritten digits' classification better to illustrate the distinctions between these lifelong learning scenarios. MNIST dataset [112] is commonly adopted for all three types of scenarios [105]. In a task-incremental scenario, each task requires classifying between two specific digits. The model is aware of the pair of digits it needs to differentiate for each task, thus focusing its classification efforts on these two digits exclusively. In contrast, in a class-incremental scenario, the model is tasked with classifying among all the digits it has encountered up to that point. Each task also contains two digits, but since task labels are unavailable, the model cannot rely on them to determine which task is being addressed. Instead, the model must continually expand its classification capability to include new digits. Figure 2.1 presents an illustrative example of an example scenario built on top of MNIST dataset. Finally, in a domain-incremental scenario, the model initially learns to classify all digits. Subsequent tasks introduce new representations of these classes, such as rotated images of the same digits. The challenge here lies not in learning new digits but in adapting to these new presentations of already learned classes. Notably, recent years brought a few emerging trends in terms of lifelong learning scenarios, such as simultaneous adaptation to new classes and new representations of the same class [75], and repetition of already encountered tasks [31].

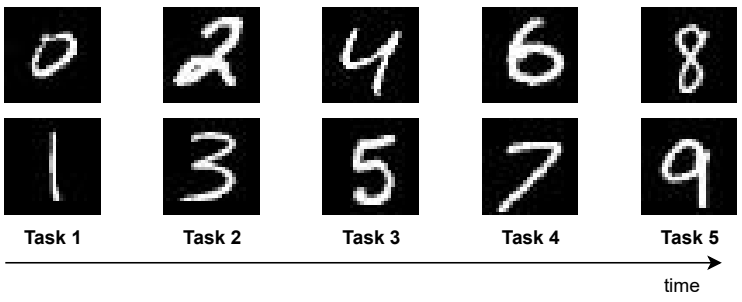


Figure 2.1: An example scenario built on top of MNIST dataset. It consists of 5 tasks presented over time.

Most existing lifelong learning works concentrate on one of the three previously mentioned scenarios (class-incremental, task-incremental, domain-incremental). However, a more detailed classification of lifelong learning is presented in [74], offering the possibility to develop even more complex lifelong learning scenarios. Moreover, recent years brought a few emerging trends in terms of lifelong learning scenarios, such as simultaneous adaptation to new classes and new representations of the same class [75] and repeating training of already encountered tasks, allowing the model to refine its performance [31]. Particularly noteworthy is the task-agnostic scenario, which deserves special attention due to its closer resemblance to many real-world applications. Unlike the other scenarios, task-agnostic learning operates under the assumption of a complete absence of task labels and task boundaries [74]. This setting presents a realistic and challenging environment where the model has to autonomously identify and adapt to new tasks. Such a scenario closely mirrors real-world conditions where the boundaries and nature of tasks are not explicitly defined and must be inferred from the data [26].

Strategies

Lifelong learning research has experienced significant growth of interest in recent years, resulting in the development of various strategies aimed at facilitating continuous adaptation while addressing the challenge of catastrophic forgetting. The strategies offer mechanisms to balance the acquisition of new knowledge with the preservation of previously learned information. These strategies are usually categorized into three types: regularization-based, architecture-based, and rehearsal-based (or replay-based) approaches.

- **Regularization-based approaches** focus on preserving the weights essential for previously learned tasks while learning new tasks by imposing constraints during the learning process. Notable examples include Elastic Weight Consolidation (EWC) [61] and Learning Without Forgetting (LWF) [71], which modify the regularization loss by including knowledge distillation techniques to mitigate significant changes in weights that are important for past tasks.
- **Architecture-based approaches** adapt the model's architecture in response to new learning experiences. Many methods in this category include expanding the model by adding new neurons or layers upon encountering new tasks [35]. Other techniques incorporate dynamic adaptation through pruning less important weights or dividing the model into separate sub-networks specialized in different tasks [56].
- **Replay-based methods**, also known as rehearsal-based, focus on retaining knowledge from previous tasks by periodically integrating a summarized version of past data (stored in a replay buffer) into the model's updates [87]. The key challenge lies in selecting the most representative data samples for inclusion in the replay buffer [19]. Moreover, some methods leverage generative models to generate artificial data from previous tasks each time the model is updated [99]. Despite their relative simplicity, replay-based methods can yield high results, even when the memory capacity allocated for the replay buffer is limited. Due to this, we focus our efforts on designing replay-based methods.

Gaps

As we mentioned in Chapter 1, while lifelong learning is an emerging topic with numerous methods proposed in recent years, a predominant focus has been observed in areas such as image classification and reinforcement learning with a secondary focus on domains like object recognition [87, 1, 88, 13]. However, there is a notable gap in the integration of lifelong learning principles with anomaly detection, even though lifelong learning could benefit the performance of anomaly detection models in evolving environments with recurring tasks.

Moreover, most existing lifelong learning methods are designed for scenarios where either task labels or boundaries are available. This design limits their applicability in task-agnostic settings, as they are unable to identify a change and control their learning process. The relative scarcity of approaches capable of operating in task-agnostic environments highlights a significant area for future research.

2.2 Anomaly detection

Anomaly detection, also called outlier and novelty detection, aims to identify anomalous data samples that exhibit significant deviations from the typical patterns and conditions of analyzed data [3, 94]. The authors of [51] characterize an outlier (anomaly) as: "An observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism". This concept is essential in multiple domains due to its ability to flag data points that are potentially indicative of errors, novel phenomena, or system irregularities [22]. The implications of anomaly occurrence may be far-reaching and vary depending

on the specific field of application. For example, in industrial settings, recognizing anomalies may help detect malfunctions in the manufacturing process [6, 28]. Due to its significant practicality in multiple domains, anomaly detection has been widely researched in the last few decades [15, 86].

One of the domains in which anomaly detection plays a pivotal role is cybersecurity [36]. Traditional security measures often are unable to detect complex and evolving cyber threats effectively. Therefore, the ability to identify deviations from normal patterns is crucial in recognizing indicators of attacks and security breaches [58]. Anomaly detection has been successfully leveraged in multiple cybersecurity applications, showcasing its versatility and critical importance in this field. One notable example is intrusion detection, in which anomaly detection methods can analyze network traffic and systems logs to identify any deviations that could mean an intrusion [58]. Other prominent examples include malware detection [47] and phishing detection [18].

As we explained in Chapter 1, anomaly detection algorithms often work in an evolving environment, where they adapt by utilizing either a full retraining stage or following an online adaptation [40]. Both types of approaches are valuable depending on the specific characteristics of the domain. Updating models may lead to forgetting, which is an acceptable characteristic in some domains, such as crowd anomaly detection [39] and real-time fatigue detection [66]. However, as we learn from lifelong learning, forgetting may also be a significant limitation for a model [87], especially in applications where previous and current data are vital for accurate anomaly detection. Therefore, addressing catastrophic forgetting is essential in developing robust and reliable anomaly detection systems that can maintain a comprehensive understanding over time and multiple tasks.

In addition to the distinction between the online and offline approaches discussed above, anomaly detection methods can further be categorized into supervised, semi-supervised, and unsupervised based on the availability of labels [22, 86, 48]. Supervised anomaly detection methods require labeled data for both normal and anomalous classes. A critical challenge these methods face is the issue of class imbalance, as anomalies usually have much smaller availability than normal data [3, 59]. Moreover, their efficacy is constrained to identifying known types of anomalies, thereby limiting their capacity to uncover new, previously unseen anomalies [5]. Semi-supervised methods, in contrast, are trained on normal data and do not require prior knowledge of anomalies. These methods attempt to identify anomalies in unseen data by detecting deviations from the learned distribution of the normal class [22]. Finally, unsupervised methods differ in that they do not rely on labeled training data, as they are purely data-driven and identify anomalies based on the intrinsic characteristics of the available unlabeled data [22]. Works in literature [48] suggests leveraging semi-supervised methods when sufficient labeled normal data is available, as this can lead to the development of more robust models.

The distinctive characteristics of semi-supervised and unsupervised learning settings often entail adopting one-class learning models. Such models are designed to identify anomalies by learning primarily from normal data and assessing deviations in new data. Relevant examples of one-class learning anomaly detection methods, which we also implement on our study, are: *i)* Variational Autoencoder (VAE) [60], a neural-network reconstruction-based model trained to minimize the reconstruction error on the training data. Anomalies are then identified based on the magnitude of this reconstruction error, effectively using it as an anomaly score. *ii)* One-Class Support Vector Machine (OC-SVM) [95], which provides anomaly scores by comparing new data against a decision boundary formed by a hyperplane learned during the training stage. *iii)* Local Outlier Factor (LOF) [17], which calculates an anomaly score based on the local density contrast of a new data sample with respect to the average local density of its nearest neighbors; *iv)* Isolation Forest (IF) [73], which employs tree ensembles where the length from the root to the leaf is used as an anomaly score, with shorter paths typically indicating an isolated sample and, therefore, an anomaly. *v)* Copula-based anomaly detection (COPOD) [70], which assesses the degree of "extremeness" of data samples based on tail probabilities of an empirical multivariate cumulative copula distribution function.

2.3 Lifelong anomaly detection

The critical gaps identified in the above discussion can be summarized as follows: i) Most lifelong learning research primarily focuses on image classification and reinforcement learning. The strategies and scenarios developed are often tailored specifically for these domains. This specialization leaves a research void in applying lifelong learning principles to domains such as anomaly detection. ii) Similarly, anomaly detection research is primarily directed towards offline and online methods, with a notable lack of focus on knowledge retention and limiting forgetting. iii) Most existing methods in lifelong learning rely heavily on the availability of task labels or clear task boundaries. This reliance significantly constrains the possibility of leveraging them in task-agnostic scenarios that closely resemble real-life conditions.

These gaps indicate a severe need for research in **lifelong anomaly detection**, a combination of lifelong learning and anomaly detection. The aim of lifelong anomaly detection is to create more robust anomaly detection methods with the capability to adapt without forgetting the previous knowledge and fit to work in challenging anomaly detection applications.

It is noteworthy that during our work on this thesis, a few papers have been published attempting to explore the combination of lifelong learning and anomaly detection. While very scarce and varied in their approaches, these recent contributions signify the growing relevance and potential of lifelong anomaly detection. One observable trend is the extension of lifelong learning in image vision to the realm of anomaly detection. [43] introduced ARCADe, a method utilizing meta-learning to identify optimal parameter values for each task in one-class image anomaly detection. Similarly, [37] proposed a transfer learning approach for anomaly detection in surveillance videos. Another approach leveraging lessons learned in lifelong image classification is [8], where authors implement binary classification for anomaly detection under supervised learning settings. Similarly, [109] extend the idea of generative experience replay to anomaly detection in cybersecurity. We can also observe a trend of implementing lifelong learning for domain-specific anomaly detection. In [25], the authors explored the application of a simple upper-bound strategy gathering all data observed so far in Software Defined Networks. The other approach is presented in [79], which implements regularization techniques to enhance the performance of models in manufacturing anomaly detection, adhering to a domain-incremental setting with available task boundaries.

These attempts confirm that a combination of lifelong learning and anomaly detection is a relevant topic significant for multiple real-world domains. However, we can observe that existing works miss common ground in terms of scenarios, metrics, and evaluation protocol. Moreover, a common limitation of these studies is their focus on scenarios with available task boundaries, hence not addressing the more complex and realistic task-agnostic conditions. To the best of our knowledge, the only study that approaches anomaly detection task-agnostic scenarios is CPDGA [26], but it leverages scenarios without recurring tasks, focusing on the detection of task changes and adaptation without measuring forgetting. None of the existing studies reference the problem of contaminated training data, which is a widespread issue in anomaly detection applications. Finally, the research on lifelong learning for anomaly detection is minimal and sporadic in comparison to lifelong learning in domains such as computer vision and reinforcement learning. One of the reasons behind this is the subject's novel and emerging nature and the lack of established practices, protocols, and guidelines.

Our work seeks to fill these gaps by thoroughly exploring, motivating, and explaining the concept of lifelong anomaly detection. We aim to establish foundational principles regarding scenarios, strategies, and metrics pertinent to this field. Furthermore, we propose methods specifically designed to operate under challenging conditions, such as task-agnostic scenarios and environments with contaminated data. This effort positions our research at the forefront of advancing the application of lifelong learning principles in the complex and evolving landscape of anomaly detection.

Chapter 3

Lifelong anomaly detection: challenges, perspectives and insights

In this chapter, we present an overview of our paper "Lifelong Continual Learning for Anomaly Detection: New Challenges, Perspectives, and Insights". While this paper is one of the last chronologically, we present it at the beginning of this dissertation because it summarizes many lessons learned during previous research. Moreover, this work highlights the potential of lifelong anomaly detection, explains the benefits of its adoption, and provides foundations for scenarios, strategies, and metrics.

In this paper, we aim to bridge the gap between anomaly detection and lifelong learning (Contribution **C1**) by:

- Presenting the advantages of lifelong anomaly detection, and putting a foundation for new methods more adjusted to evolving real-world conditions;
- Describing a characterization of lifelong anomaly detection scenarios, showcasing how to create more challenging benchmarks for anomaly detection;
- Proposing a scenario generation method allowing transforming traditional anomaly detection datasets to lifelong learning scenarios, helping in wider adoption of lifelong anomaly detection;
- Describing the evaluation protocol and metrics, aiming to lay the foundation for future lifelong anomaly detection research.
- Emphasizing challenges and limitations that conventional anomaly detection methods encounter in our proposed lifelong learning scenarios.

We present a brief overview of these contributions below, while the full text is available in Chapter 10.

3.1 Challenges of lifelong anomaly detection

In Chapters 1 and 2, we explained the need for anomaly detection methods that can simultaneously adapt to new knowledge and retain previously learned knowledge. In this section, we focus on the more detailed, intuitive analysis of the advantages that can be brought by introducing lifelong learning to anomaly detection.

Figure 3.1 demonstrates a comparison between conventional anomaly detection with model updates and lifelong anomaly detection in a representative lifelong scenario containing 4 tasks repeating multiple times. As a reminder, a lifelong scenario is a sequence of tasks that the model has to learn one by one while preserving its ability to solve all previously presented tasks. As we can see in Figure 3.1, the critical

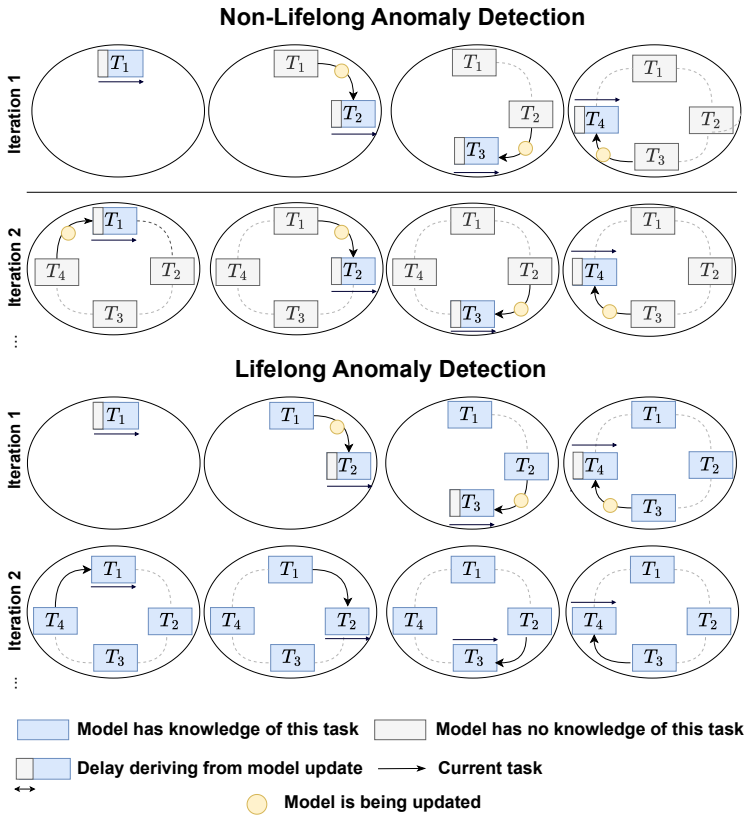


Figure 3.1: A scenario with four recurring tasks (T_1, T_2, T_3, T_4). Conventional anomaly detection requires constant model updates and results in detection delays. Lifelong learning mitigates this burden by retaining knowledge of tasks.

distinction materializes during the second iteration of a recurring task when lifelong anomaly detection demonstrates a significant advantage by eliminating the need for model updates with each task recurrence. In contrast, conventional anomaly detection models have to be updated each time a task recurs, leading to an inherent delay in detection and constant resource usage required for training.

Figure 3.2 illustrates another comparison between conventional and lifelong anomaly detection approaches. This comparison showcases scenarios where the normal class data evolves over time, requiring the model to be updated as new tasks emerge. The conventional approach leveraging model updates tends to prioritize recent data, leading to forgetting previous tasks and misclassification. Contrastingly, the lifelong learning method adapts to new tasks while retaining knowledge of previous ones, creating a more robust and comprehensive anomaly detection model that can handle all tasks.

Overall, while the conventional anomaly detection with the model update approach may be able to relearn previously forgotten tasks, this approach presents significant disadvantages and limitations, such as delays, lower detection accuracy, and extensive use of computational resources.

By analyzing and generalizing the scenarios depicted in Figures 3.1 and 3.2, we identify several significant limitations of conventional anomaly detection methodologies that rely on constant model updates:

- The tendency to forget past knowledge and focus only on adapting to new normal class distributions can lead to a significant decrease in performance before the model can be updated to the recurring task;

- Failure to leverage previously learned skills to provide a more comprehensive solution to all tasks;

- Too simplistic experimental settings, not reflecting the complexity of many real-world challenges, such as the appearance of new situations and the recurrence of old situations;

- The continuous model retraining to accommodate recurring tasks requires significant computational resources;

- From a theoretical perspective, the model will lack a comprehensive view of the environment and will not be able to leverage task similarity and knowledge transfer from already learned tasks to new ones.

Due to these limitations, current anomaly detection models are somewhat simplistic compared to human-level intelligence capable of constantly learning over time and building up knowledge on previous experiences. Extending models with similar abilities should allow them to exhibit more sophisticated behavior, resulting in more robust and accurate models [64].

In essence, adopting lifelong learning in anomaly detection will bring several advantages that will allow models to align closer with the complex, dynamic nature of real-world scenarios. To sum up, the minimal set of benefits includes capabilities such as: *i)* simultaneous adaptation and knowledge preservation; *ii)* leveraging a more comprehensive overview of the environment; *iii)* less resource-demanding model updates than conventional anomaly detection methods with constant model updates; *iv)* more realistic experimental evaluation considering all tasks encountered throughout the models' lifetime.

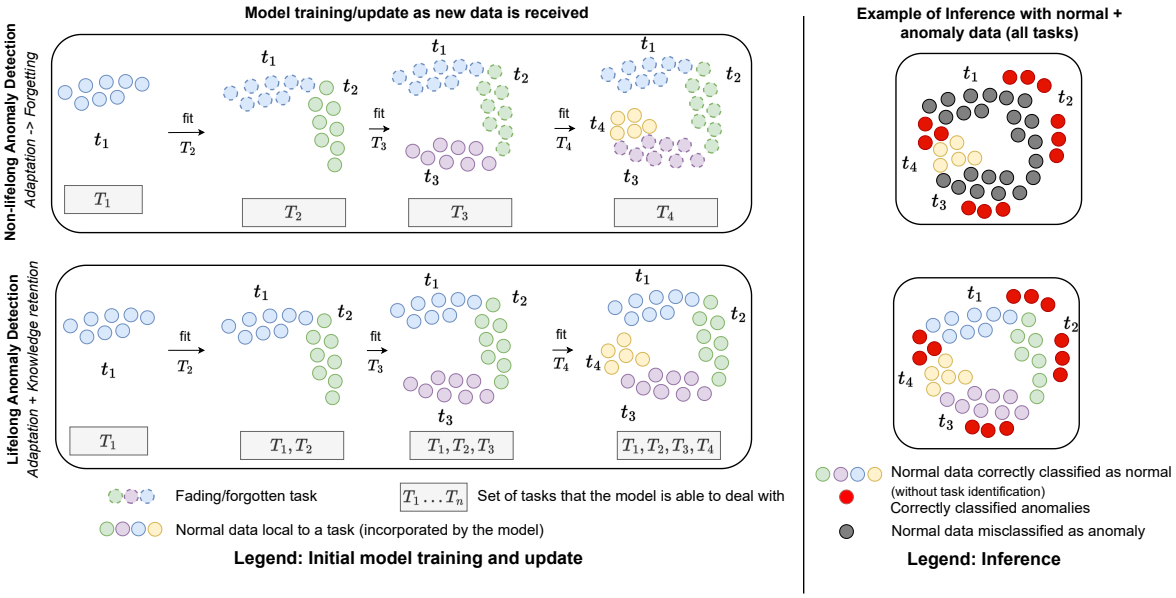


Figure 3.2: Comparison of training/update and inference for non-lifelong and lifelong anomaly detection in the scenario with four tasks (T_1, T_2, T_3, T_4). In non-lifelong anomaly detection, the model forgets the previous tasks as soon as a new task is learned (left – top). In contrast, the lifelong anomaly detection model aims to retain knowledge of all tasks (left – bottom). This characteristic has a serious impact on the model's behavior during inference (right). In non-lifelong anomaly detection, after learning task T_4, the model misclassifies data from previous tasks as anomalous since it considers only data from the current task as normal behavior (right – top). On the other hand, the ideal lifelong anomaly detection model retains the knowledge of all tasks, preventing the misclassification of normal data from previous tasks as anomalous (right – bottom). This difference in behavior between non-lifelong and lifelong anomaly detection may lead to a discrepancy in their performance scores.

3.2 Lifelong learning scenarios

Considering the novelty of lifelong anomaly detection and the scarcity of research in this field, establishing standardized procedures and guidelines is crucial. Therefore, we propose a categorization of lifelong learning scenarios, propose a scenario creation algorithm, and describe the evaluation protocol.

Lifelong learning scenarios from an anomaly detection perspective

Current lifelong learning methods predominantly concentrate on classification tasks. In this context, *tasks* are defined as sets of classes. For instance, in the MNIST dataset, a task may comprise any combination of two classes. The learning workflow is usually a sequence of n tasks $T = t_1, t_2, \ldots, t_n$ where the model is charged with learning new tasks without forgetting previous tasks. As we mentioned in Chapter 2, three primary scenarios are: task-incremental, class-incremental, and domain-incremental.

However, the notion of a task in anomaly detection notably diverges from the conventional understanding present in image classification. Anomaly detection typically operates on two classes: normal and anomalous. In this domain, new challenges and conditions are brought as evolving characterization of the "normal" class, which needs to be learned to detect anomalies. Moreover, what is normal in one context can be anomalous in another, further increasing the complexity of the problem. To differentiate lifelong anomaly detection settings, we define *a self-consistent behavior of the normal class, alongside the specific anomalies occurring with it*, as a **concept**. This behavior could correspond to a single distribution, one of many performed activities, or a distinct state of the environment, depending on the specific analytical context considered. From now on, we will use the term **concept** instead of **task** in our extended summary whenever referencing lifelong anomaly detection. We note that the term "task" is used in part of our publications that were prepared before introducing the idea of "concept".

In lifelong anomaly detection, we encounter multiple consistent behaviors or concepts of the normal class over time, in contrast to introducing new classes as seen in class and task-incremental scenarios. Moreover, this approach focuses on the evolution of a singular 'normal' class rather than the evolution of all classes as observed in domain-incremental scenarios. This differentiation highlights the necessity for scenarios tailored specifically for lifelong anomaly detection settings.

Following the example of anomaly detection in human conditions, *concept identifiers* inform about the unique id of a single concept (a specific activity), while *concept boundaries* represent exact knowledge of whether the presently analyzed concept (a specific activity) has changed. Concept identifiers and concept boundaries may correspond to task labels and task boundaries known in lifelong image classification.

Based on this deliberation, we specify the following learning scenarios in increasing order of complexity:

- *Concept-aware*: Known concept identifier and concept boundaries.
- *Concept-incremental*: Unknown concept identifier but known concept boundaries.
- *Concept-agnostic*: Unknown concept identifier and concept boundaries.

In our example of anomaly detection in human conditions, a concept-aware scenario means that the model is fully informed about the current activity and its duration (at both training and inference time). In a concept-incremental scenario, the model only knows that the activity has changed but does not know which activity is currently being processed. Ultimately, in the most challenging concept-agnostic scenario, the model has no information about the current activity and its duration. These notions are general and can be adopted in any domain.

Algorithm for scenarios creation

A substantial challenge in advancing the research on lifelong anomaly detection is the need for datasets tailored explicitly for this purpose. To address this gap, we propose a scenario creation procedure that can be applied to most anomaly detection datasets and enable researchers to adapt their current scenarios toward lifelong anomaly detection.

We present a general pseudocode for scenario adaption procedure in Algorithm 1. The algorithm is customizable and allows users to define the number of desired concepts $\mathbf{c}$ and three functions: ϕ, γ, and λ. ϕ and γ are concept creation functions that transform the original dataset into self-consistent sets of data samples with shared characteristics. ϕ is responsible for creating concepts from normal data, while γ focuses on creating concepts of anomalies. An example implementation for ϕ and γ is any clustering algorithm. λ is an assignment function that creates a combined concept by matching each normal concept with an anomaly concept. The sequence of these combined concepts builds the complete scenario. An example implementation for λ is matching an anomaly cluster with the closest normal cluster.

Algorithm 1 starts with the creation of concepts for the normal class leveraging a concept creation function ϕ (Line 1) and the creation of concepts for the anomaly class leveraging the anomalous concepts creation function γ (Line 2). Then, the algorithm uses a function λ to select an anomaly concept C_{A_j} for each normal concept C_{N_i} (Lines 3-5). Then, the combination of C_{N_i} and C_{A_j} is added to the scenario (Line 6). Every time a new concept is created, the anomaly concept C_{A_j} is excluded from the set of available anomaly concepts C_A to avoid repetition in the scenario (Line 7). The algorithm yields the lifelong scenario as a sequence of concepts (Line 9) that may need to be further split into training and evaluation data depending on the learning settings, e.g., unsupervised or semi-supervised.

Input: c – Number of desired concepts
Input: $\mathbf{N}$, $\mathbf{A}$ – Normal/Anomaly data
Input: ϕ – Concepts creation function for normal data
Input: γ – Concepts creation function for anomalies
Input: λ – Assignment function

1 $C_N \leftarrow \phi(\mathbf{N}, c)$ // Create normal concepts $\{C_{N_0}, C_{N_1}, \ldots, C_{N_c}\}$
2 $C_A \leftarrow \gamma(\mathbf{A}, c)$ // Create anomaly concepts $\{C_{A_0}, C_{A_1}, \ldots, C_{A_c}\}$
3 $T \leftarrow [\ ]$ // Initialize result scenario as an empty sequence
4 **for** $C_{N_i} \in C_N$ **do**
5 $\quad j \leftarrow \lambda(C_A, C_{N_i})$ // Match anomaly-normal concepts
6 $\quad T \leftarrow T \cup (C_{N_i}, C_{A_j})$ // Add concepts to scenario
7 $\quad C_A \leftarrow C_A - C_{A_j}$ // Remove used anomaly concept
8 **end**
9 **return** T

Algorithm 1: Scenario creation algorithm

3.3 Evaluation protocol and metrics

In lifelong anomaly detection, a crucial requirement for the assessment of lifelong capabilities of the models is the continuous evaluation of the model across all encountered concepts. To this aim, we have implemented a comprehensive evaluation protocol that systematically assesses the model's performance.

Algorithm 2 describes a protocol offering a general overview applicable across a broad spectrum of use cases. This protocol is flexible and can be adapted to specific requirements. For example, in Chapter 6, we adapt it to concept-agnostic scenarios.

The evaluation algorithm starts by initializing a matrix R that will gather detection results for specific concepts (Line 1). Then, the protocol iterates over training sets for all concepts (Line 2). For each concept, the model is trained/updated (Line 3) and evaluated on all testing sets for all concepts (Lines 4-5), i.e., previous, current, and future. Our protocol yields a matrix R, where entries $R_{i,j}$ define the ROC-AUC metric of the model evaluated on concept j after learning concept i. We select ROC-AUC [16] (Area Under Receiver Operating Characteristic Curve) as our base metric, as it is one of the best-suited metrics for assessing anomaly detection performance. An ROC plot shows the trade-off between the true negative rate and the true positive rate.

Input: $\mathbf{T}$ – Sequence of N training sets
Input: $\mathbf{E}$ – Sequence of N testing sets
Input: L – Learning model
Input: ρ – Evaluation function computing ROC–AUC
1 Initialize an empty results matrix $R_{N \times N}$
2 **for** $T_i \in \mathbf{T}$ **do**
3 $\quad L \leftarrow$ update L with T_i `// Train/update model`
4 $\quad$ **for** $E_j \in \boldsymbol{E}$ **do**
5 $\quad\quad R_{i,j} \leftarrow \rho(L, E_j)$ `// Evaluate` L `on` E_j
6 $\quad$ **end**
7 **end**
8 **return** R

Algorithm 2: Pseudo-code of the evaluation protocol

The matrix R can be used to compute lifelong learning metrics directly, such as backward and forward transfer. These metrics allow us to assess model behavior more extensively than standard performance metrics by taking into account the model's performance on different concepts (previous, current, and future).

Inspired by [38], we propose a **Lifelong ROC-AUC** – a lifelong variant of ROC-AUC that can adequately assess models' performance on all concepts after learning every new concept, instead of models' performance on just a single concept. It is defined as:

$$\text{Lifelong ROC-AUC} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{i} R_{i,j}}{\frac{N(N+1)}{2}} \tag{3.1}$$

Lifelong ROC-AUC considers the model's performance on all previously learned concepts, including the current concept, which equals to averaging over $\frac{N(N+1)}{2}$ entries from lower triangular.

We favor ROC-AUC over threshold-dependent measures such as Precision, Recall, and F–Score because ROC-AUC provides a more comprehensive evaluation of the model's performance. Moreover, it is possible to swap ROC–AUC with other metrics of choice, keeping the validity of the protocol.

Backward Transfer for ROC-AUC (BWT) quantifies the effect of learning new concepts on the model's performance on all previously learned concepts. Negative backward transfer indicates the occurrence of forgetting, which means that the model's performance on previous concepts deteriorates as it learns new ones. On the other hand, positive backward transfer implies that introducing new concepts improves models' performance on previously learned concepts. Backward transfer is computed over all concepts as:

$$BWT = \frac{\sum_{i=2}^{N} \sum_{j=1}^{i-1} (R_{i,j} - R_{j,j})}{\frac{N(N-1)}{2}} \tag{3.2}$$

Forward Transfer for ROC-AUC (FWT) is a metric that assesses the influence of learning new concepts on the model's performance on future concepts Forward transfer can also be considered the zero-shot

model performance on future concepts since it assesses model performance on unknown concepts. It partially depends on concept similarity (task similarity) and the model's knowledge transfer ability. It is defined as:

$$FWT = \frac{\sum_{i=1}^{N}\sum_{j=i}^{N} R_{i,j}}{\frac{N(N-1)}{2}} \tag{3.3}$$

3.4 Experimental results overview

Our experiments aim to answer two research questions: *i)* Do lifelong scenarios impact the performance of non-lifelong anomaly detection models?; ii) Does adopting lifelong learning provide a valuable improvement for anomaly detection models in lifelong scenarios?

In our experiments, we employ three distinct strategies to evaluate lifelong learning models:

- Naive Lifelong Strategy: In this strategy, the model is continually retrained solely based on newly incoming data, without employing any mechanisms for balancing adaptation and knowledge retention. A notable consequence of this strategy is forgetting previously learned information. The naive lifelong strategy can be considered a lower bound and provides a reference point to understand how the non-lifelong anomaly detection method behaves in lifelong scenarios.

- Multiple Single-Task Experts (MSTE): This approach creates a pool or ensemble of models, where each model is trained to be an expert on a single concept. Whenever a new concept is discovered, a new model is trained on the data from this concept and then added to the ensemble. It is important to note that MSTE is not a feasible strategy in real-world scenarios, as it would require infinite computational resources to accommodate the ever-increasing number of models. However, the MSTE strategy can be considered an upper bound and provide an insightful comparison to viable lifelong strategies. Moreover, MSTE requires knowledge of concept identifiers to select the appropriate model from the ensemble. As a result, it must be evaluated in the concept-aware setting.

- Replay Strategy: This strategy maintains a replay buffer that stores selected data samples from previous concepts. The buffer has a limited size and is revised to include new data whenever the model discovers a new concept. The reasoning behind replay-based strategies is that the replay buffer enables the model to mitigate forgetting by providing a representation of all previously learned concepts. In our experiments, we leverage a balanced replay buffer with a small budget of 3,000 data samples.

We conduct experiments leveraging a diversified set of datasets, such as intrusion detection datasets (NSL-KDD [102] and UNSW-NB15 [82]) and power production plants (Energy [27] and Wind [28]). The selected datasets present a variety of feature representations, types of anomalies, and complexity levels. We build three variants of lifelong learning scenarios based on different choices of γ, ϕ, and λ functions:

- **CC**: clustered anomaly concepts assigned to the closest normal concept (γ and ϕ are a clustering function, and λ maps a normal concept to the closest available anomaly concept).

- **CR**: clustered anomaly concepts assigned randomly to normal concepts (γ and ϕ are a clustering function, and λ assigns a random anomaly concept to a given normal concept).

- **R**: anomalies randomly assigned to normal concepts (γ is a clustering function, ϕ creates anomaly concepts using random sampling, and λ assigns a random anomaly concept to a given normal concept).

These scenarios are conceptually represented as two-dimensional plots in Figure 3.3. We implement k-Means as the clustering function in all scenarios. All scenarios are set in concept-aware settings, which means that the lifelong learning strategies have knowledge of when the concept changes (concept boundaries) and which concept is currently being processed (concept identifiers). We include five anomaly detection methods: IF, LOF, COPOD, OC-SVM, and VAE (see Chapter 2) .

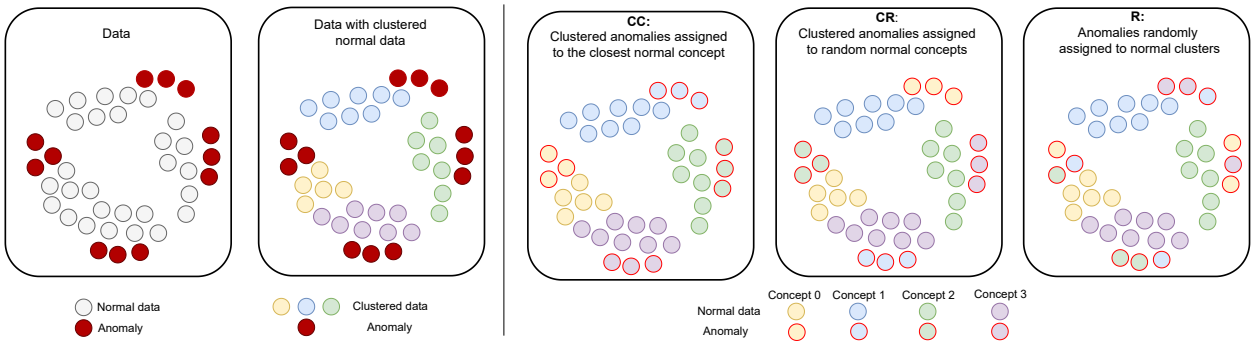


Figure 3.3: Lifelong scenarios variants based on different choices of functions γ, λ and ϕ: *i)* clustered anomaly concepts assigned to the closest normal concept (**CC**), *ii)* clustered anomaly concepts assigned randomly to normal concepts (**CR**), and *iii)* anomalies randomly assigned to normal concepts (**R**).

We start by assessing lifelong scenarios' influence on non-lifelong anomaly detection methods. We utilize the previously mentioned strategies, Naive Lifelong and Multiple Single-Task Experts (MSTE), to compare model performance in lifelong and non-lifelong scenarios. In this case, MSTE can be considered an upper bound for the base model simulating non-lifelong scenarios with one model created for every concept. At the same time, the Naive strategy showcases the behavior of baseline models challenged with lifelong scenarios without any smart knowledge retention mechanism. We present the visual representation of the results in Figure 3.4

We can observe a notable performance gap in ROC-AUC between conventional anomaly detection methods and the hypothetical upper bound set by MSTE across all methods and datasets. Looking at the Energy dataset, the MSTE approach significantly outperforms the Naive strategy (for instance, for Isolation Forest, CC: 0.88 vs. 0.64; CR: 0.97 vs. 0.65; 0.97 vs. R: 0.59). This pattern is consistent across other datasets as well.

Notably, the MSTE strategy consistently achieves high performance across most scenarios. This finding indicates that our proposed scenario generation algorithm effectively decomposes the complexities of each dataset into simpler sub-complexities or concepts. While easier to learn in isolation, these concepts present a significant challenge in a lifelong learning context. This phenomenon aligns with observations made in non-lifelong one-class classification [98].

The lower ROC-AUC values achieved by non-lifelong methods in lifelong scenarios suggest that these methods are penalized by the evolving nature of such scenarios, leading to suboptimal performance. In summary, the observed gap in ROC-AUC performance between Naive lifelong and MSTE strategies confirms that created lifelong learning scenarios are challenging for non-lifelong anomaly detection methods.

The second part of our analysis aims to examine the potential benefits that can be brought by the adaptation of lifelong learning strategies on top of base anomaly detection models. To this aim, we analyze the results obtained with the Replay strategy in comparison to the results of Naive and MSTE strategies.

Compared to the Naive strategy, Replay leads to performance improvement to varying degrees, depending on the scenario. This pattern of improvement is consistent across all base models and datasets, as Replay outperforms the Naive strategy in 54 out of 60 cases. The exceptions are primarily in the UNSW (R) scenario and can be attributed to the complexity of concepts in the UNSW dataset and the Replay strategy's

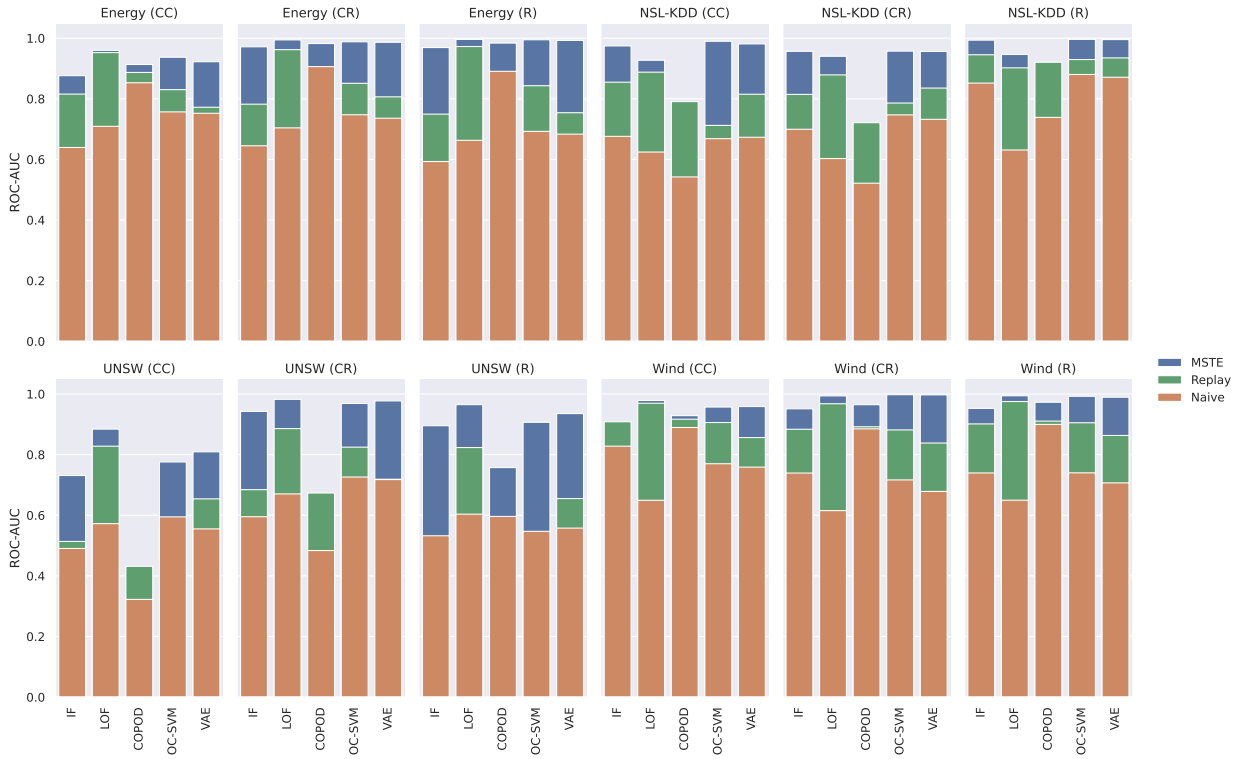


Figure 3.4: Summary of experimental results for all datasets (Energy, UNSW, Wind) in three scenario types (CC, CR, R) comparing strategy without any knowledge retention mechanism (Naive), lifelong (Replay), and upper bound (MSTE) learning strategies. This figure illustrates the performance gap between strategies without and strategies with knowledge retention mechanisms in lifelong anomaly detection scenarios.

relative simplicity. It suggests the need for more sophisticated lifelong learning approaches in such complex scenarios.

Moreover, we can observe a performance gap between the Replay strategy and the simulated upper-bound results achieved by MSTE. In most instances (56 out of 60), MSTE demonstrates superior performance compared to Replay. However, there are four cases where Replay surpasses MSTE. The possible explanation is that while MSTE excels in building specialized models for each concept, it lacks the ability to leverage and transfer knowledge across multiple tasks, a capability inherently supported by the Replay strategy.

In conclusion, the performance improvements brought by the Replay strategy underscore the benefits of integrating lifelong learning strategies and techniques for anomaly detection. Moreover, there is a need for designing new and more robust strategies to effectively tackle the complexities inherent in lifelong anomaly detection scenarios, as indicated by the performance gap between Replay and MSTE. A more detailed experimental analysis is presented in Chapter 10.

Chapter 4

Benchmarking lifelong learning methods

As we mentioned in Chapter 2, there are three main types of approaches to designing lifelong learning methods for image classification and reinforcement learning: i) replay-based, ii) regularization-based, and iii) architectural-based. The landscape of existing lifelong image classification works is characterized by diverse learning datasets, methods, and evaluation metrics. This diversity complicates the comparative analysis and assessment of models and strategies. In this work, we aim to fairly evaluate current state-of-the-art lifelong learning strategies on a common ground closer to a complex real-world scenario. To this end, we propose two novel lifelong image classification benchmarks and evaluate 12 existing lifelong image classification methods.

While this work is focused on image classification, it supports our lifelong anomaly detection research presented in the next chapters by: i) confirming the validity of our choice to integrate replay-based methods as the first ones into lifelong anomaly detection in challenging scenarios; ii) indicating a need to create more robust lifelong learning methods; iii) showcasing which other methods may be worthy of adopting to lifelong anomaly detection. Our work was published in the paper "From MNIST to ImageNet and Back: Benchmarking Continual Curriculum Learning". We present a brief overview of the contributions below and full text in Chapter 11.

4.1 M2I and I2M benchmarks

In this work, our objective is to evaluate current state-of-the-art lifelong image classification strategies within a unified framework that more accurately reflects the complexities of real-world situations. Therefore, we introduce two new lifelong image classification benchmarks: M2I (Mnist to Imagenet) and I2M (Imagenet to Mnist). They aim to assess lifelong image classification methods in conditions that more closely resemble real-world challenges. To this aim, these benchmarks incorporate multiple heterogeneous tasks derived from six image datasets to provide new and unprecedented situations significantly differing from the ones encountered previously. Moreover, each task exhibits different levels of complexity and image quality (for example, black and white vs. color) since, in the real world, the model should present generalization capabilities to deal with data from sources with varying quality. Finally, to assess how the models are dependent on the task order, we organize the datasets into curriculum learning (increasing level of complexity; M2I) and reverse curriculum learning (decreasing order of complexity; I2M). It also allows us to assess whether lifelong methods can effectively exploit this structure across tasks and datasets. Figure 4.1 showcases our proposed benchmarks.

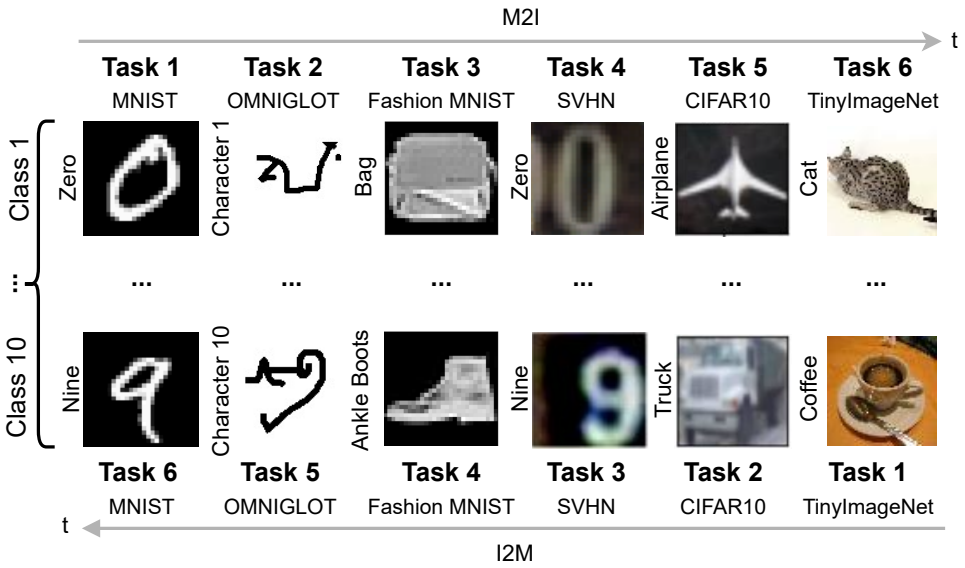


Figure 4.1: The Proposed M2I and I2M lifelong learning benchmarks. There are 6 different tasks, each sampled from a different dataset: MNIST [112], OMNIGLOT [65], Fashion MNIST [111], SVHN [84], CIFAR10 [63] and TinyImageNet [67]. Tasks are organized in two curriculum ordering, from simple to harder (left to right) and backward (right to left). Every task contains 10 classes.

4.2 Experimental results overview

In our experiments, we evaluate 12 image classification lifelong learning methods representing all three major types of approaches: regularization-based (EWC [61], LwF [71], MAS [7], SI [113]), replay-based (AGEM [23], GEM [33], GenerativeReplay [99], Replay [92]), and architectural-based (CWRStar [75], PNN [93]). We also include two unrealistic upper bounds: i) Multiple-Single Task Expert strategy that assumes having an independent model for each task; ii) Cumulative strategy that assumes unlimited data storage to keep all data encountered in the scenario. Moreover, we evaluate two baselines: i) Naive strategy that simply updates the model with the most recent data; ii) GDumb [9] strategy that greedily stores samples in a buffer and uses them to retrain a model from scratch. The methods are evaluated in task-incremental and class-incremental scenarios (whenever possible, as some methods work only in task-incremental settings). Moreover, we repeat experiments with three different model architectures to assess if the model size impacts the performance of lifelong learning methods. While the full results are available in Chapter 11, we present the results for the EfficientNet [101] model architecture and M2I scenario in Table 4.1.

Our comprehensive experimental evaluation across all scenarios and models reveals that widely recognized lifelong image classification methods do not maintain satisfactory performance levels within our proposed benchmarks. Furthermore, these methods exhibit significant limitations related to forgetting. Except for the two upper bounds (MSTE and Cumulative), Replay achieves the best results, consistently maintaining the capability to retain knowledge while adapting to new tasks. Moreover, the results suggest that existing methods do not leverage the structured curriculum task ordering designed to enhance performance and robustness. This finding underscores the need to develop more advanced lifelong learning strategies to effectively integrate and retain knowledge across varied learning contexts. A more detailed experimental analysis is presented in Chapter 11.

Table 4.1: Experimental results (EfficientNet – M2I) in terms of average performance (and rank) for all lifelong learning strategies grouped by type (from top to bottom: regularization, replay-based, architectural, and baseline) in two learning settings (class-incremental, task-incremental)

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.180 (9)	-0.222	0.272 (11)	-0.199	0.106
LwF	0.165 (11)	-0.205	0.240 (12)	-0.124	0.096
MAS	0.187 (7)	-0.190	0.276 (10)	-0.198	0.104
SI	0.184 (8)	-0.231	0.277 (9)	-0.205	0.099
AGEM	0.179 (10)	-0.224	0.328 (7)	-0.171	0.091
GEM	0.312 (5)	-0.056	0.420 (5)	-0.043	0.096
GenerativeReplay	0.517 (3)	-0.143	0.551 (4)	-0.102	0.097
Replay	0.655 (2)	-0.041	0.605 (3)	-0.098	0.102
CWRStar	0.326 (4)	-0.014	0.367 (6)	0.001	0.092
PNN			0.091 (14)	0.000	0.100
Naive	0.205 (6)	-0.257	0.290 (8)	-0.220	0.103
GDumb	0.029 (12)	-0.007	0.099 (13)	0.000	0.100
Cumulative	0.834 (1)	0.003	0.656 (2)	0.025	0.101
MSTE			0.784 (1)	0.000	0.100

Chapter 5

Change point detection for lifelong learning

As we mentioned in Chapters 1 and 2, one of the unexplored challenges relevant to lifelong anomaly detection is learning without the explicit knowledge of when the concept changes. In such concept-agnostic scenarios, the lifelong learning method has to discover change on its own to be able to adapt to new data. To this aim, in this chapter, we describe our two methods: WATCH, a change point detection method capable of working with high-dimensional data common in real-life anomaly detection applications; and LIFEWATCH, a change point detection explicitly designed for lifelong scenarios and capable of recognizing new and recurrent concepts. LIFEWATCH aims to support lifelong anomaly detection methods working in concept-agnostic scenarios, instrumental to our Contribution **C2**. We leverage a modified LIFEWATCH version in our work on the concept-agnostic lifelong anomaly detection method presented in Chapter 6.

Change point detection is the process of identifying time points when the data distribution changes [12]. It provides a significant grant to data analysis in domains characterized by a dynamic and time-evolving nature, such as human activity recognition, monitoring in smart grids, and intrusion detection. Most change point detection methods described in the literature are limited to yielding good performance on univariate data or a restricted number of dimensions in multivariate data [89]. However, when data dimensionality increases, these methods either experience a decline in accuracy or a notable drop in execution time. As anomaly detection methods often have to work with high-dimensional data, we propose WATCH, our change point detection method that addresses the problem of high-dimensionality by introducing a Wasserstein-distance-based method tailored specifically for high-dimensional time series data. WATCH maintains a representation of the last detected distribution and tracks its evolution while identifying change points by calculating the Wasserstein distance between new data points and the distribution. WATCH was published in the paper "WATCH: Wasserstein Change Point Detection for High-Dimensional Time Series Data". We present the overview of contributions and experimental results in this chapter, while the full text is available in Chapter 12.

Change point detection techniques are often integrated into more complex machine learning methods to signal shifts in data distribution, prompting necessary actions [100, 62]. From the lifelong learning perspective, change point detection methods may be instrumental in concept-agnostic scenarios where the lifelong learner is not informed about the change of concept being currently processed. In such scenarios, the learner may need to detect the change of concept on its own to be able to adapt. However, existing change point detection methods still need to be improved for the requirements of lifelong learning, as they are unable to distinguish between changes that indicate a new concept and changes that indicate a recurring concept. Consequently, this limitation may affect lifelong learning methods in terms of robustness and accuracy.

In our work built on top of WATCH, we propose LIFEWATCH – lifelong change point detection method that leverages a memory component to offer lifelong capabilities and is specifically tailored for the analysis

of multivariate time series data in lifelong learning scenarios. Our method efficiently identifies changes in multivariate time series in an unsupervised manner, distinguishing between new concept appearances and recurring concepts. We conduct an extensive experimental evaluation on real-world multivariate time series datasets, showcasing results in traditional change point detection and lifelong consolidation. LIFEWATCH was published in the paper "LIFEWATCH: Lifelong Wasserstein Change Point Detection". We present the overview of contributions and experimental results in this chapter, while the full text is available in Chapter 13.

5.1 WATCH

The foundation of the WATCH method is the use of the Wasserstein distance to enable high-quality change point detection in high-dimensional data. The Wasserstein metric is also known as the earth mover's distance due to its intuitive explanation through an analogy, in which the distribution is seen as an amount of earth (soil) piled on the metric space, and the metric represents the minimum "cost" of turning one pile into the other. The cost is visualized as the amount of earth to be moved multiplied by the mean distance it has to be moved. Recent advancements in machine learning [45] successfully incorporate this metric into various model architectures, highlighting its effectiveness.

The rationale behind adopting the Wasserstein distance in our method lies in its ability to measure the work required to transport the probability mass from one distribution to another, distinguishing it from distances such as the Kullback-Leibler divergence. This characteristic makes the Wasserstein distance a good candidate in domains where analyzing the similarity of outcomes is more relevant than an exact likelihood matching. Figure 5.1 demonstrates that while measures like the Kullback-Leibler divergence may suggest identical distances between the red and blue distributions, the Wasserstein distance is able to capture "the work" involved in transitioning the probability mass from the red to the blue state.

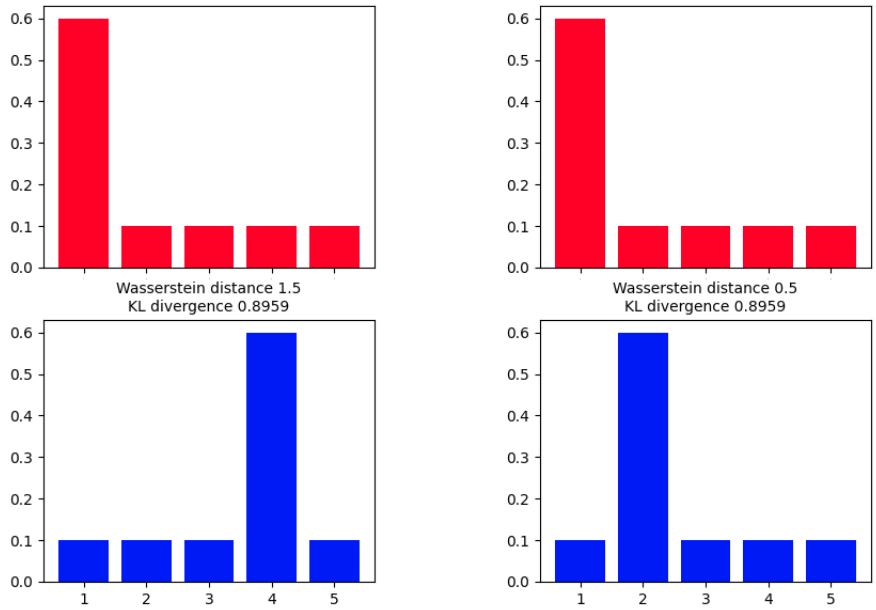


Figure 5.1: Difference between Wasserstein distance and Kullbach-Leibler (KL) divergence.

Furthermore, a significant advantage of the WATCH method, enabled by its use of the Wasserstein distance, is the capacity to compare two sets of data points directly rather than relying on the parameters of

specific distribution types. This feature enables comparison between distributions of unknown types without restricting the scope to a small subset of possible distribution types.

Intuitively, WATCH learns new distributions in an unsupervised manner, storing newly available samples in a buffer until the buffer reaches its pre-defined capacity. At this point, the change detection process starts. For each mini-batch of incoming data, the method calculates the Wasserstein distance between the mini-batch and the current distribution. The distance is compared to a threshold calculated according to the distribution's internal structure and multiplied by the sensitivity parameter tuned by the user. If the Wasserstein distance is lower than a threshold, the new data is consistent with the characteristics of the current distribution and classified as part of it. Contrarily, if the distance exceeds the threshold, it suggests a notable divergence from the current distribution. Therefore, the current mini-batch is considered to belong to a different distribution. Furthermore, this mini-batch is marked as a change point, signaling a shift in the data stream and starting the creation of a new distribution.

Our experimental study aims to benchmark the performance of WATCH against state-of-the-art algorithms, particularly emphasizing high-dimensional datasets. To this aim, we utilize the Turing Change Point Detection (TCPD) benchmark, which enables us to assess WATCH alongside 13 competitor algorithms across 42 time-series datasets described in [106]. Additionally, to ensure the robustness and relevance of our evaluation, we have extended the TCPD benchmark with four high-dimensional datasets described in 12. Our experimental evaluation reveals the potential of WATCH, which outperforms all considered competitors in the majority of the cases and manages to provide consistent and robust detection with the datasets presenting the highest dimensionality. We present the detailed algorithm and the comprehensive experimental study in Chapter 12.

5.2 LIFEWATCH – lifelong change point detection

The crucial lifelong capability of LIFEWATCH is distinguishing between changes that mean the occurrence of a new concept and changes meaning the recurrence of the already observed concept. To do this, LIFEWATCH keeps the pool of already discovered distributions P, and, whenever a change is detected, it compares new data to known distributions to identify to which one of the new data belongs.

Algorithm 3 presents details on how our method works. The incoming data is processed in the form of mini-batches B_i. LIFEWATCH acquires an initial understanding of an unknown distribution by collecting data points in a buffer for the current distribution D_C until reaching minimum distribution size κ, at which point D_C is included in the distribution pool P. To identify whether new data belong to one of the distributions in P, we maintain a threshold $E[D_j]$ for each distribution D_j. In LIFEWATCH, each threshold $E[D_j]$ is updated to reflect the largest distance across mini-batches B_d assigned so far to a given distribution D_j, scaled by a factor ϵ, as defined below:

$$E[D_j] = \epsilon \max_{B_d \in D_j} W_A(B_d, D_j). \tag{5.1}$$

The scaling factor ϵ can be adjusted to suit various domain-specific characteristics, such as concept volatility.

LIFEWATCH starts the change point detection process by evaluating whether B_i conforms to the characteristics of the current distribution D_C. If the Wasserstein distance between B_i and D_C is lower than the established threshold $E[D_C]$, B_i is considered a part of D_C. On the other hand, a distance exceeding $E[D_C]$ indicated the emergence of a new, unknown distribution.

Upon detecting a change point, LIFEWATCH categorizes it into one of two types: new concept changes, represented as N, and recurring concept changes, denoted as R. This determination involves calculating the closest distribution, D_k, by assessing the lowest ratio of the Wasserstein distance $W_A(B_i, D_j)$ to the specific threshold $E[D_j]$ for each distribution. If the distance to the closest distribution D_k is less than the threshold

Input: $\mathbf{I} = (I_1, I_2, \ldots, I_T)\,; I_i \in R^N$ Time series data

Result: N Change points (New concepts),
R Change points (Recurring concepts).

Parameters: κ Min points required in a new distribution before starting detection,
μ Max points stored for a distribution,
ϵ Threshold ratio for the distance value,
ω Mini-batch size.

Initialize pool of distributions $P \leftarrow \emptyset$
Initialize pool of distribution thresholds $E \leftarrow \emptyset$
Initialize current distribution $D_C \leftarrow \emptyset$
Initialize results variables $N \leftarrow \emptyset, \;\; R \leftarrow \emptyset$

$\mathbf{B} \leftarrow$ Process $\mathbf{I}$ into sequence $(B_1, B_2, \ldots, B_{\frac{T}{\omega}})$ of non-overlapping mini-batches B_i of size ω
for $B_i \in \mathbf{B}$ **do**
  if $|D_C| < \kappa$ **then**
    $D_C \leftarrow D_C \cup B_i$
    if $|D_C| >= \kappa$ **then**
      Determine $E[D_C]$ from Eq. (5.1) with ϵ ; `// Save threshold for` D_C
      $P \leftarrow P \cup D_C$; `// Add` D_C `to the distribution pool`
    end
  else
    $\upsilon \leftarrow W_A(B_i, D)$; `// Compute Wasserstein Distance`
    if $\upsilon > E[D_C]$ **then**
      $$D_k = \begin{cases} \underset{D_j \in P}{\arg\min} \dfrac{E[D_j]}{W_A(B_i, D_j)}, & \text{if } \underset{D_j \in P}{\exists} W_A(B_i, D_j) < E[D_j] \\ \emptyset, & \text{otherwise} \end{cases}$$
      Set change point $c \leftarrow i \cdot \omega$
      if $|D_k| = 0$ **then**
        $N \leftarrow N \cup \{c\}$; `// Change point (New concept)`
      else
        $R \leftarrow R \cup \{c\}$; `// Change point (Recurring concept)`
      end
      $D_C \leftarrow D_k$; `// Set` D_k `as a current distribution`
    end
    if $|D_C| < \mu$ **then**
      $D_C \leftarrow D_C \cup B_i$
      Determine $E[D_C]$ from Eq. 5.1 with ϵ ; `// Update threshold for` D_C
    end
  end
end
return N, R

Algorithm 3: Pseudo-code of the proposed `LIFEWATCH` method

$E[D_k]$, B_i is considered as part of D_k, prompting a shift in the current distribution from D_C to D_k. The identified change point is recorded in the set R for recurring concepts. Alternatively, if the distance to D_k exceeds $E[D_k]$, this suggests that B_i does not belong to any known distribution, leading to the creation of a new distribution. In this case, the method initializes a new distribution D_C, and the change point is added to the set N.

When LIFEWATCH identifies the appropriate distribution for a mini-batch B_i, it integrates B_i into the current distribution D_C, allowing it to reflect new data points. Alongside this integration, the corresponding threshold $E[D_C]$ is also updated, ensuring that D_C remains accurately representative of ongoing data trends. Additionally, the method limits the size of D_C according to a storage capacity parameter μ. This strategy allows LIFEWATCH to balance detailed data representation with a small memory footprint, making it efficient and adaptable for diverse data stream environments

There are a few advantages of LIFEWATCH design when compared to traditional change point detection techniques:

- Unlike methods that rely on the parameters of predefined distributions, LIFEWATCH compares sets of data points directly. This approach circumvents the need to make assumptions about the type of distribution, thereby enhancing its applicability across varied data scenarios.
- LIFEWATCH distinctively categorizes change points into two types: new concepts and recurring concepts.
- The method autonomously updates thresholds for each learned distribution. This automation eliminates the need for manual intervention, ensuring a consistent and unbiased adjustment to changing data patterns.
- The method efficiently avoids unnecessary comparisons with all pool distributions if new data aligns with the current distribution.

Experimental results overview

We design lifelong change point detection as a support tool for lifelong methods working in scenarios without explicit concept boundaries. For this reason, our experiments aim to assess LIFEWATCH's capabilities of detecting changes in data distribution and identifying the number of concepts present in data.

To evaluate LIFEWATCH in terms of change point detection, we adopt the Turing Change Point Detection (TCPD) benchmark [106], which allows us to evaluate 13 existing algorithms on 42 time series datasets. Moreover, we add four multivariate time series datasets containing challenging data representing real-world scenarios: **i) Human Activity Recognition** (HAR) dataset [10] containing six types of human activities; **ii) Movement** dataset [34] containing sequences of 15 hand movement types in Brazilian signal language; **iii) Traffic** dataset [41] containing sequential urban traffic measurement; **iv) Eye** dataset [91] containing records of EEG measurements while observing the eye of the subject.

We adopt the metrics leveraged by the authors of the TCPD benchmark [106] to maintain consistency and allow an easy comparison for future works:

- **F1** – Harmonic mean of Precision and Recall, where Recall represents the method's capacity to find all relevant change points, and Precision measures the method's ability to identify only relevant change points.
- **Cover** – Describes how well the predicted segments cover the ground truth segments, where a single segment corresponds to a subsequence of contiguous points.

Moreover, as we aim to simulate a lifelong learning environment characterized by the emergence of new concepts over time, we adopt existing datasets so that they present every concept a few times. To do this, we enumerate the set of potential concepts corresponding to the k distinct distributions in each dataset. Then, we generate up to ten random permutations of identified concepts. In order to further increase the complexity and realism of the scenario, we generate a sequence of concepts with a three-time recurrence, resulting in sequential scenarios of $(k \times 3)$ concepts. This adaptation increases the difficulty of the experimental conditions for each considered method because the methods must determine whether changes in data distribution signify a completely new concept or a recurrence of a previously encountered one.

Table 5.1 presents the results of our experimental evaluation. We carry out experiments with multiple parameter values for each method. The results demonstrate that our LIFEWATCH method outperforms others in terms of the average Cover and F1 metrics across both univariate and multivariate datasets. These findings validate the efficacy of the LIFEWATCH method in standard change point detection concepts. Even though LIFEWATCH may not be significantly better than all other methods, it proves to be accurate and efficient in identifying changes in data streams.

Table 5.1: Change point detection: results in terms of Cover and F1 metrics averaged across multiple parameter configurations. Blank values indicate that the method does not support multivariate data. Results for the best-performing method are marked in bold.

	Univariate		Multivariate	
	Cover	F1	Cover	F1
AMOC	0.726	0.778		
BINSEG	0.760	0.833		
BOCPD	0.769	0.857	0.705	0.785
BOCPDMS	0.737	0.620	0.285	0.263
CPNP	0.538	0.648		
ECP	0.711	0.789	0.616	0.665
KCPA	0.619	0.679	0.583	0.671
PELT	0.706	0.766		
PROPHET	0.574	0.537		
RBOCPDMS	0.422	0.349	0.069	0.092
RFPOP	0.403	0.517		
SEGNEIGH	0.763	0.832		
WBS	0.417	0.519		
ZERO	0.573	0.658	0.268	0.332
LIFEWATCH	**0.798**	**0.908**	**0.713**	**0.800**

We also assess LIFEWATCH ability to identify the number of concepts present in data. To this aim, we put LIFEWATCH in a consolidation experiment, leveraging its ability to differentiate between two types of change points to adapt it to consolidation setup. We compare our results with three classes of methods: i) Change point detectors, where a new cluster is created every time a change point is detected: BOCPD [2] and ECP [80]; ii) Conventional clustering methods, which require knowledge of the entire dataset from the start, thus giving them a strong advantage: BIRCH [114]), OPTICS [11] and MEAN-SHIFT [24]); iii) Incremental stream clustering methods: DENSTREAM [21], HDBSCAN [20], and TRESTLE [76]. We perform experiments on four high-dimensional datasets mentioned above: HAR, Movement, Traffic, and Eye.

Figure 5.2 visually presents the lifelong consolidation outcomes regarding the number of identified clusters (concepts). This graphical depiction allows for an immediate comparison of the performance of various methods against the actual number of concepts, indicated by a dashed line. We can observe that

LIFEWATCH demonstrates superior results for all datasets except the Eye dataset. For the Eye dataset, an initial comparison indicates that the MEAN-SHIFT algorithm, on average, extracts a number of clusters closer to the ground truth than LIFEWATCH, with a marginal difference of 0.5. However, a more detailed analysis of results for all datasets reveals that MEAN-SHIFT tends to produce a consistent number of clusters across different datasets, suggesting a lack of adaptability. Notably, LIFEWATCH operates incrementally, identifying clusters without prior knowledge of the data distribution, while traditional clustering algorithms have the advantage of analyzing the whole dataset. Despite this, the average number of clusters identified by LIFEWATCH is remarkably close to the ground truth, underscoring its effectiveness. We present the detailed algorithm and the comprehensive experimental study in Chapter 13.

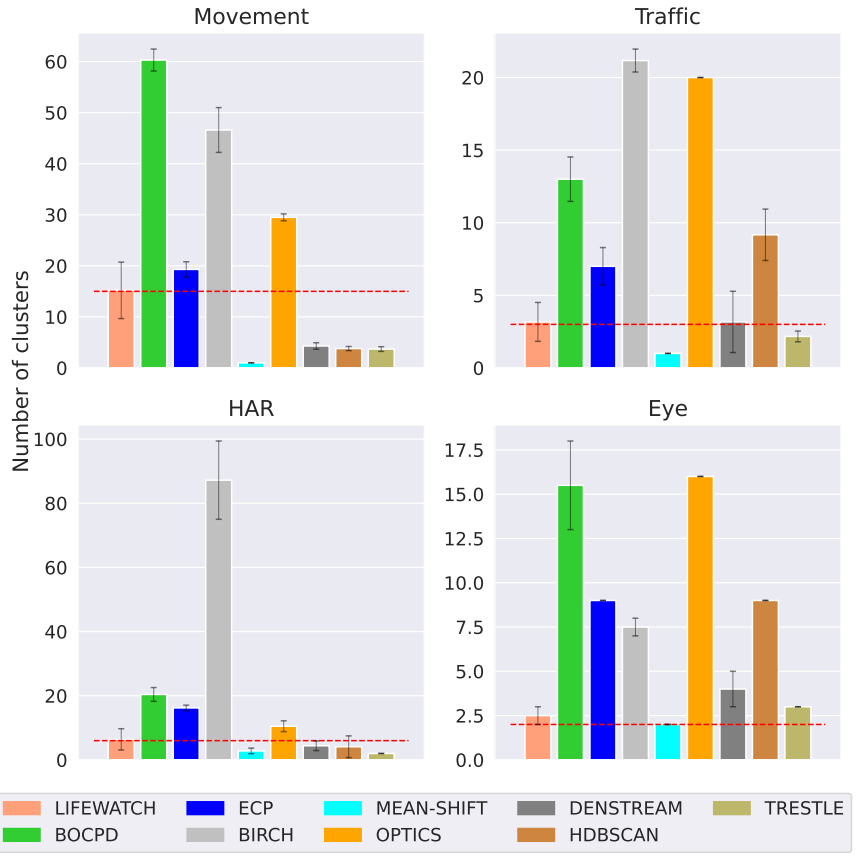


Figure 5.2: Lifelong consolidation: Number of clusters extracted (average and standard deviation) for all methods and datasets considered. The dashed line is the ground truth, i.e. the number of actual concepts in each dataset.

Chapter 6

VLAD – concept-agnostic lifelong anomaly detection

As we mentioned in Chapter 1 and Chapter 2, a significant shortcoming of currently existing lifelong learning methods is that they are designed to work in class-incremental and task-incremental scenarios that require the availability of task (concept) boundaries and labels. While this assumption may be suitable for multiple image classification and reinforcement learning cases, it is infeasible for many anomaly detection applications. In such scenarios, it is often impossible to have external information about the occurrence of a new concept delivered from the environment. Therefore, a robust lifelong anomaly detection method working in challenging concept-agnostic scenarios must be able to perform its duties while detecting changes and providing an effective model update strategy (Contribution **C2**).

To this aim, we designed `VLAD` – a novel **V**AE-based **L**ifelong **A**nomaly **D**etection method. `VLAD` tackles concept-agnostic lifelong anomaly detection through a comprehensive strategy combining lifelong change point detection with actively maintained memory and an effective VAE model update strategy.

Specifically, `VLAD` utilizes the Lifelong Change Point Detection component to discover changes and recognize whether the change brings a recurring or new concept. This recognition supports accumulating and organizing knowledge in organized short and long-term memories. The Consolidation component manages a hybrid of short and long-term memories, hierarchically structuring experiences in concepts and subconcepts. Further memory refinement is carried out via a *Memory Summarization* component, preserving relevant experiences within a compact memory-constrained representation. Finally, an *Experience Replay* component triggers model updates, leveraging the memorized experiences. Our experimental evaluation includes four cybersecurity datasets, therefore supporting our Contribution **C3**.

`VLAD` was published in the paper "VLAD: Task-agnostic VAE-based lifelong anomaly detection". We present the overview of the method and experimental results below, while the full text is available in Chapter 14.

6.1 Method overview

Figure 6.1 presents the overview of the learning stage of `VLAD` and its components. `VLAD` process input data in batches S_k, which are used in model initial training and updates. Moreover, input data is processed by Lifelong Change Point Detection, which is designed to identify potential changes in the data presented by the environment. The Lifelong Change Point Detection component can distinguish between changes signifying recurrent distributions (concept) and changes signaling an entirely new distribution (concept). Lifelong Change Point Detection leverages this information to decide where the new data should be integrated. If the distribution is new, it is added to the Emerging Pool, which conceptually functions as a short-term memory.

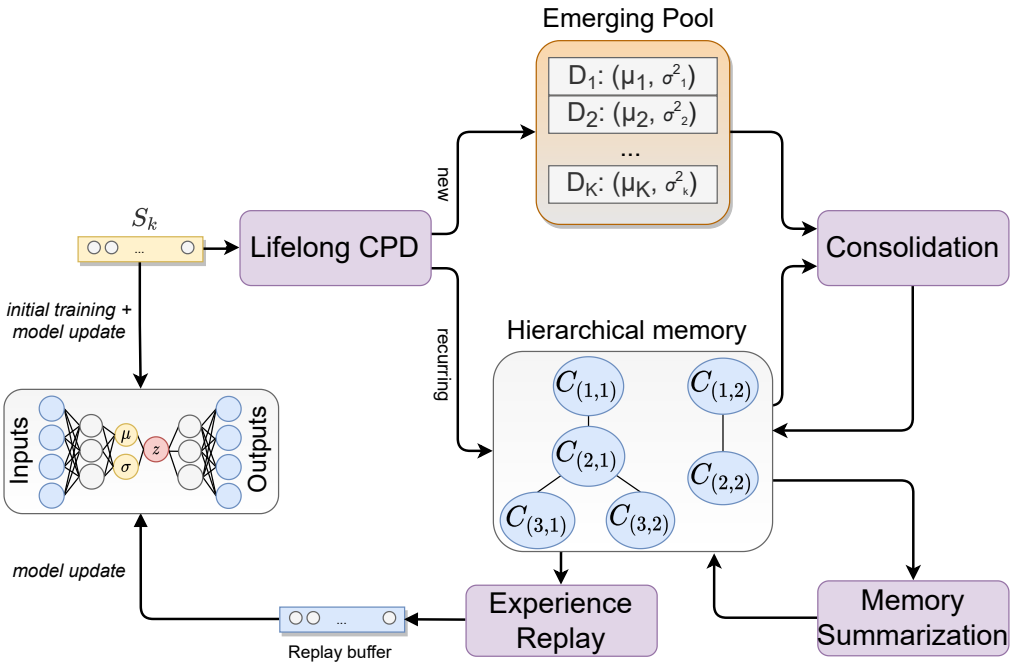


Figure 6.1: VLAD: Learning stage, featuring the combined effort of Lifelong CPD, Memory Summarization, Consolidation, and Experience Replay. The VAE model is trained and updated with learning data in combination with replay buffers provided by the Experience Replay component.

On the other hand, data representing recurring distributions is utilized to update the corresponding concept within the Hierarchical Memory, serving as a long-term memory. The new distributions existing temporarily in the Emerging Pool are subsequently integrated into the hierarchical memory through the Consolidation component. Over time, the Experience Replay component constructs a replay buffer by recalling past experiences from the memory. The data from the buffer is combined with newly available data to mitigate forgetting during updating the Variational Auto-Encoder model. When the designated memory capacity is surpassed, the Memory Summarization component is activated to ensure the stability of memory occupation by employing Pyramidal Summarization, followed by the within-concept summarization technique.

Lifelong Change Point Detection

Our method's ability to detect changes in a lifelong learning context endows it with concept-agnostic learning capabilities, as it does not rely on predefined concept boundaries. However, conventional change point detection methods only indicate a change has occurred without specifying the type (new vs. recurring concept). If these methods were to build a replay memory structure, the memory representations might not accurately reflect the concept taxonomy, leading to some concepts being underrepresented and others overrepresented. This imbalance could cause the model to forget concepts that are not frequently observed or are overshadowed by concepts that are recurring more often. In this work, we adopt an enhanced version of LIFEWATCH (introduced in the previous Chapter 5) as a lifelong change point detection method capable of discriminating between changes that describe transitions to new concepts and to already observed concepts. Furthermore, it is crucial to use this feedback to identify new data points as parts of distributions seen previously. This ability should, in principle, help resolve the imbalanced memory representation issue mentioned before, thereby reducing the risks of model degradation and forgetting. Figure 6.2 presents an illustrative comparison between the two change point detection methods, focusing on their effectiveness in this context.

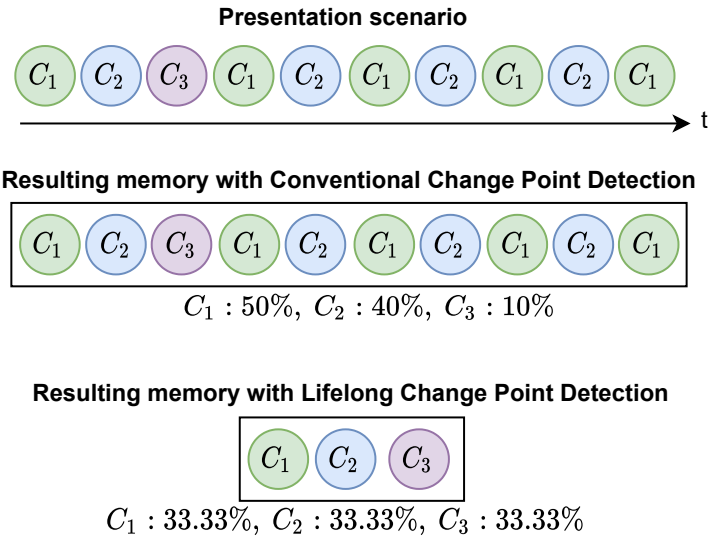


Figure 6.2: Comparison of conventional change point detection and lifelong anomaly detection in terms of memory storage. The ability to characterize different types of changes in our lifelong change detection approach yields a balanced taxonomy for concepts observed over time. For each concept, we report their percentage of representation in memory.

For the conciseness of this overview, we do not present the algorithm and its full discussion. They can be found in Chapter 14.

Memory Consolidation

The Consolidation component is crucial for building, organizing, and preserving knowledge, a fundamental aspect of lifelong learning. This ability is necessary to retain past knowledge (stability) while assimilating new information (plasticity). In VLAD, knowledge is stored in a long-term Hierarchical Memory $\mathcal{M}$, mod-

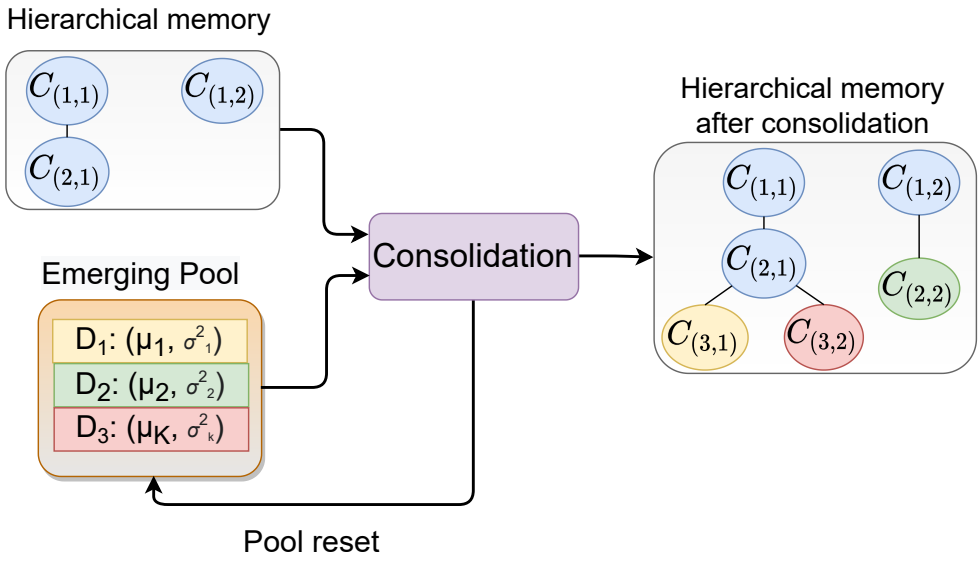


Figure 6.3: The *Consolidation* component is responsible for organizing concepts and sub-concepts in the Hierarchical Memory, and for triggering the Pool Reset mechanism. It takes the Emerging Pool and the Hierarchical Memory in their current state and returns an updated version of the Hierarchical Memory.

eled after taxonomic hierarchies present in various real-world scenarios. This memory consists of a forest structure encompassing root-level concepts and their sub-concepts. Conceptually, the definition of concepts and sub-concepts is associated with domain-specific characteristics of the analyzed data. For example, in network traffic analysis, concepts could differentiate between different communication protocols, with sub-concepts further detailing specific activities performed within this protocol. We leverage Wasserstein distance to compute the distance between distributions during the Consolidation process. New sub-concepts are recognized through notable, yet relatively minor, deviations from the initial known distribution. In contrast, the identification of entirely new concepts is marked by larger Wasserstein distances. Figure 6.3 showcases the general overview of how the Consolidation process leverages the current version of Hierarchical Memory and the Emerging Pool to build a new version of Hierarchical Memory.

Memory Summarization

Accumulating all data over time can quickly exhaust memory resources due to the extensive number of experiences collected and retained in the hierarchical memory. Continuously storing every experience is impractical, highlighting the necessity for memory summarization mechanisms to keep data manageable and easily updatable. Additionally, the ability to summarize and generalize is vital for ensuring the model's longevity in lifelong learning environments. This component is responsible for summarizing experiences stored in memory and works in synergy with the *Consolidation* component to maintain a lightweight and accurate knowledge representation.

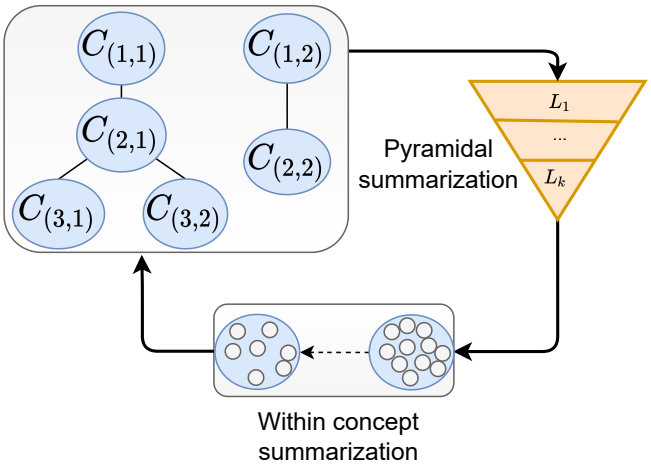


Figure 6.4: The *Memory Summarization* component balances the granularity level of concepts through pyramidal summarization. This granularity is exploited during the within-concept summarization process, which generalizes experiences in each concept. The resulting memory maintains the same hierarchy but balances the number of samples of each concept based on their level in the hierarchy.

In our approach, the memory summarization process is initiated when the memory's capacity surpasses the predefined upper limit set by the user. We present a schematic representation of this component in Figure 6.4. Inspired by the effectiveness of pyramid-based networks in [72], we propose a novel Pyramidal Summarization strategy for generalizing experiences stored in the hierarchical memory. Due to the hierarchical structure of our memory, the Pyramidal Summarization strategy gives more importance to root-level concepts, which are represented with a broader set of samples compared to sub-concepts. This strategy is designed to maintain the number of stored experiences within the upper limit M_B.

Moreover, every concept in hierarchical memory is summarized according to our devised within-concept summarization, which aims to identify sub-groups of similar experiences within each concept and, conse-

quently, store a limited number of representative samples for each sub-group. Within-concept summarization helps to avoid storing experiences that are too similar and ensures the diversity of experiences in each concept.

Experience replay

While periodic updates are essential for the Variational Autoencoder (VAE) model to adapt to new data, retaining already acquired knowledge is another necessary capability. A common challenge with model updates is the risk of forgetting, where the model's performance on previously learned concepts declines as new information is assimilated. `VLAD` aims to prevent this phenomenon by leveraging an Experience Replay component, which allows to replay previous experiences while learning new concepts. In our approach, we move beyond the standard flat replay buffer typically described in the literature. Instead, we build an experience replay from a hierarchical memory structure with Pyramidal Summarization described above. This approach ensures that as new concepts are introduced, the significance of past experiences is recalibrated and accurately represented by the model, as depicted in Figure 6.5. This concept draws inspiration from established findings in neuroscience and biology, which underscore the hierarchical organization of neural systems, as discussed in works like [104, 14].

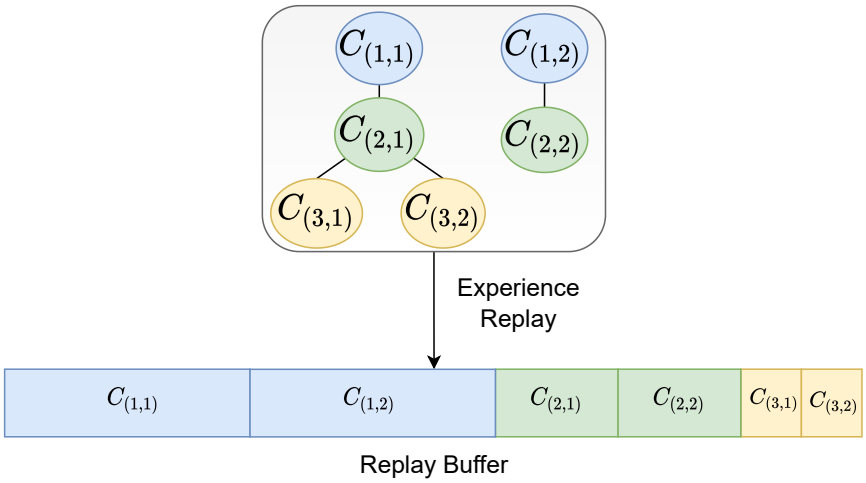


Figure 6.5: The *Experience Replay* component takes the most updated state of the memory $\mathcal{M}$ and creates a new replay buffer R each time it is required. The memory structure is built upon consolidation, and concepts at different levels are represented with a different number of samples according to the outcome of pyramidal summarization, which provides a proper balance of concepts in the replay buffer.

We devised a model update strategy that triggers the update in the following three scenarios:

- *Detection of change*: When the Lifelong CPD component detects the change and the method acquires enough data points (C_κ), the model is retrained.

- *Extended Period without Update*: There are two possibilities in this scenario: either there has been no actual change in the data distribution, or a change went undetected. In both cases, a model update is initiated after processing a predetermined number of data points since the last update (R_ψ).

- *Execution of Memory Summarization*: This occurs after the memory summarization process is triggered when the memory's hard upper limit is reached.

6.2 Experimental results overview

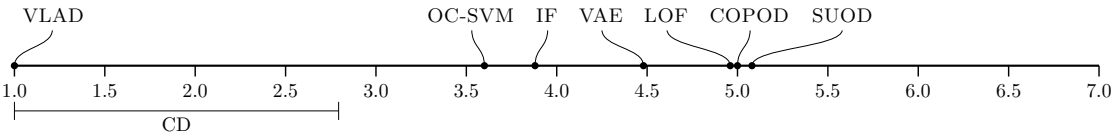
Two main goals of our experiments are to i) assess the anomaly detection performance of VLAD in comparison to non-lifelong anomaly detection models and ii) assess how well VLAD manages to mitigate forgetting. In Chapter 14, we provide a more detailed analysis of the mentioned problems, as well as the assessment of memory representation accuracy, the impact of memory size on detection performance, and a study of how different components impact the results.

Table 6.1 shows the performance in terms of ROC-AUC, BWT, and FWT (average and standard deviation) with all one class learners described in Chapter 2 (IF [73], LOF [17], OC-SVM [95], SUOD [115], COPOD [70], VAE [60], VLAD) and all analyzed scenarios (NGIDS [115], NSL-KDD [102], UNSW [82], WIND [28], and 3IDS built upon three datasets: NGIDS-IDS [49], WWW2019 [69], and ADFA-LD [32]).

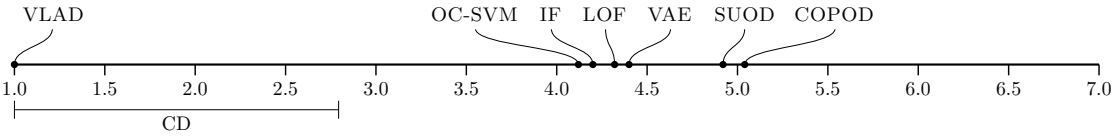
Anomaly detection performance

We compare VLAD against its baseline model (VAE) to demonstrate how the lifelong learning enhancements integrated into VLAD significantly boost anomaly detection performance. This comparison aims to validate VLAD's effectiveness in handling the complexities of a concept-agnostic lifelong anomaly detection scenario. Additionally, VLAD is compared with the leading competitor method in each scenario under study, to measure the minimum performance gain VLAD achieves over other methods in our experimental setup. For example, for *3IDS*: VLAD demonstrates an anomaly detection score of 0.779, surpassing VAE (0.597) by 30.49%, and the best competitor, IF (0.628), by 24.04%. We believe that this scenario is the most challenging, as it entails complex concepts from multiple datasets.

We conduct two different analyses to verify the statistical significance of our results. We applied the Signed-Rank Wilcoxon Test with Holm-Bonferroni Correction across all datasets to compare VLAD with other methods. The rank plot, as illustrated in Figure 6.6, underscores VLAD's superior performance according to the ROC-AUC metric in both experimental conditions: across different folds and various randomly sampled concept orderings. The visual representation in the rank plot distinctly shows VLAD outperforming other methods. We also confirm the superiority of VLAD by a pairwise analysis of p-values, as presented in Chapter 14.



(a) *K-Folds*



(b) *Orderings*

Figure 6.6: Rank plot for all datasets (ROC-AUC criterion). The algorithms positioned at the leftmost side are the best performing

Measuring forgetting

Moreover, we evaluate VLAD's effectiveness in mitigating forgetting in concept-agnostic scenarios while still maintaining robust anomaly detection performance on previously learned concepts. An analysis of the

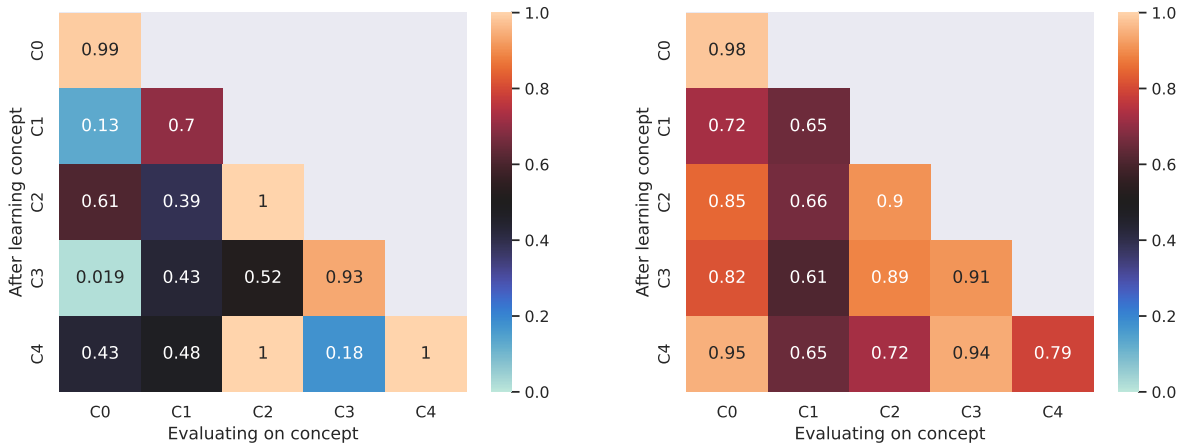
Table 6.1: Anomaly detection results in terms of Lifelong ROC-AUC, Backward Transfer (BWT), and Forward Transfer (FWT). The results are averaged across different folds (*K-Folds*).

	NGIDS			3IDS		
	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT
IF	0.615±0.032	-0.489±0.048	0.389±0.025	0.628±0.010	-0.433±0.012	0.469±0.038
LOF	0.634±0.015	-0.417±0.022	0.367±0.021	0.530±0.019	-0.483±0.035	0.451±0.158
OC-SVM	0.599±0.004	-0.526±0.006	0.227±0.007	0.594±0.003	-0.350±0.004	0.158±0.008
SUOD	0.583±0.026	-0.541±0.037	0.373±0.030	0.618±0.010	-0.426±0.019	0.416±0.080
COPOD	0.685±0.004	-0.171±0.017	0.432±0.002	0.337±0.001	-0.103±0.001	0.390±0.002
VAE	0.583±0.003	-0.558±0.010	0.231±0.010	0.597±0.007	-0.464±0.011	0.312±0.014
VLAD	0.777±0.009	-0.163±0.006	0.179±0.007	0.779±0.024	-0.108±0.021	0.426±0.021
	NSL-KDD			UNSW		
	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT
IF	0.745±0.008	-0.260±0.008	0.809±0.033	0.533±0.020	-0.423±0.021	0.536±0.035
LOF	0.610±0.010	-0.323±0.032	0.545±0.058	0.719±0.048	-0.292±0.060	0.678±0.016
OC-SVM	0.722±0.002	-0.265±0.004	0.858±0.008	0.740±0.016	-0.288±0.019	0.722±0.004
SUOD	0.683±0.020	-0.341±0.029	0.701±0.032	0.581±0.016	-0.403±0.022	0.479±0.013
COPOD	0.452±0.002	-0.128±0.001	0.268±0.001	0.361±0.001	-0.115±0.002	0.309±0.004
VAE	0.700±0.016	-0.311±0.031	0.877±0.023	0.668±0.018	-0.359±0.022	0.689±0.010
VLAD	0.880±0.013	-0.056±0.010	0.915±0.017	0.910±0.015	-0.037±0.011	0.539±0.028
	WIND					
	ROC-AUC	BWT	FWT			
IF	0.842±0.006	-0.147±0.009	0.748±0.010			
LOF	0.802±0.007	-0.268±0.012	0.808±0.019			
OC-SVM	0.876±0.000	-0.161±0.001	0.874±0.000			
SUOD	0.802±0.015	-0.239±0.024	0.696±0.014			
COPOD	0.927±0.001	-0.012±0.001	0.935±0.002			
VAE	0.858±0.008	-0.167±0.011	0.847±0.009			
VLAD	0.959±0.001	-0.014±0.003	0.843±0.008			

Backward Transfer (BWT) average values in Table 6.1 reveals that all methods experience some degree of forgetting across various scenarios. However, VLAD consistently shows the least amount of forgetting, maintaining a very low memory footprint and preserving ROC-AUC performance. For the purpose of concise comparison in this section, we focus on contrasting VLAD with the base model VAE. VLAD significantly reduces the forgetting experienced by VAE across all scenarios, achieving a 70.78% reduction in NGIDS, 76.72% in 3IDS, 81.99% in NSL-KDD, 89.69% in UNSW, and 91.61% in WIND. These results underscore VLAD's capability to significantly diminish forgetting across a spectrum of scenarios, emphasizing its utility in lifelong anomaly detection concepts.

In addition to the discussed aggregated BWT values, Figure 6.7 presents ROC-AUC measured on every single concept after learning every concept in the scenario in the most demanding 3IDS scenario. This approach allows us to dissect the average ROC-AUC performance, paying particular attention to its pro-

gression throughout the model's lifespan. We can observe that, overall, VAE exhibits significant forgetting, as indicated by a decline in ROC-AUC performance for concepts other than the most recently learned one. VLAD effectively addresses this issue, demonstrating more stable performance across various concepts.



(a) VAE

(b) VLAD

Figure 6.7: Comparison of ROC-AUC values obtained for VAE and VLAD on each concept after learning different concepts (Multiple Intrusion Detection Datasets scenario). The top line showcases the performance of the model at the beginning of the scenario after learning Concept C0. For example, value 0.61 in heatmap *a)* presents the model performance on Concept C0 after learning Concept C2. Only entries below the main diagonal are deemed relevant for analyzing forgetting since values above the main diagonal indicate performance on concepts that have not yet been seen.

For example, VAE demonstrates an initially high ROC-AUC performance of 0.99 on Concept 0, which catastrophically drops to 0.13 after learning Concept C1. After learning Concept C2, the performance on Concept C0 improves to 0.61 (barely better than random), dropping again to 0.019 after learning Concept C3 and improving to 0.43 (below random) after learning Concept C4. At the same time, VLAD can maintain some detection performance on Concept C0 after learning Concept C1 (ROC-AUC 0.72, compared to 0.13 for VAE), Concept C2 (0.85, compared to 0.61 for VAE), Concept C3 (0.82, compared to 0.019 for VAE), and Concept C4 (0.95, compared to 0.43 for VAE). Even if VAE performs better on single concepts just after learning them, its overall performance on all concepts is severely affected by forgetting. Similar observations can be drawn by analyzing single columns in the heatmap considering the entries on and below the main diagonal from top to bottom. We present the detailed description of VLAD and the comprehensive experimental study in Chapter 14.

Chapter 7

Active lifelong anomaly detection

While VLAD, described in the previous chapter, answers the problem of lifelong anomaly detection in concept-agnostic scenarios, in this work, we explore another type of challenging conditions present in lifelong anomaly detection scenarios (Contribution **C2**). One of the challenges existing in real-world unsupervised anomaly detection applications is the contamination of data, where the unlabeled training set contains not only normal data but also anomalies [54]. Including anomalies in the model's training may lead to decreased anomaly detection capabilities [108]. This problem is also essential from a lifelong anomaly detection perspective but is yet to be explored. We focus on the problem of data contamination from the perspective of replay-based methods, as they are particularly vulnerable to including contaminated data in the replay buffer R. In such a case, the contaminated data is utilized during each following model update. As a result, the inference capabilities of anomaly detection models may be significantly decreased due to the repeated incorporation of anomalies in the learned normal data distribution. Consequently, the model may tend to flag anomaly data points as normal, resulting in increased false negatives. This problem is exacerbated when anomalies from multiple concepts are included in data used to update models, which represents a more complex scenario than non-lifelong counterparts typically analyzed in the literature.

To answer this problem, we propose an active learning framework that supports any memory-based replay-based method, any query strategy, and any anomaly detection model. It leverages a limited number of queries to an external oracle in order to reduce the number of anomalies memorized in the replay buffer. Moreover, we propose two strategies that automatically identify and remove additional data points that are likely to be anomalies based on the oracle's feedback. Our experimental evaluation includes three intrusion detection datasets, therefore supporting our Contribution **C3**.

Our active lifelong anomaly detection framework was published in the paper "Active Lifelong Anomaly Detection with Experience Replay". We present the overview of the framework and experimental results below, while the full text is available in Chapter 15.

7.1 Active lifelong anomaly detection framework

As we explained in Chapter 2, the replay-based methods store a small subset of samples R_i from data T_i from each concept i to ensure knowledge retention. Whenever new data T_i is presented, our proposed framework takes the created corresponding replay buffer R_i as input and returns a cleaned version of the replay buffer $R_i^{'}$. The cleaning process involves asking an oracle (for example, an external operator) to label a few data points selected from the replay buffers according to a desired budget of B, which defines the maximum number of queries that we have available (for example, due to the available operator's time). Subsequently, an automated querying and cleaning strategy exploits the oracle's feedback to remove a more significant number of possible anomalies without performing additional queries. Leveraging a cleaned version of the

replay buffer should yield a less contaminated and more robust anomaly detection model. Figure 7.1 shows a graphical representation of the proposed framework.

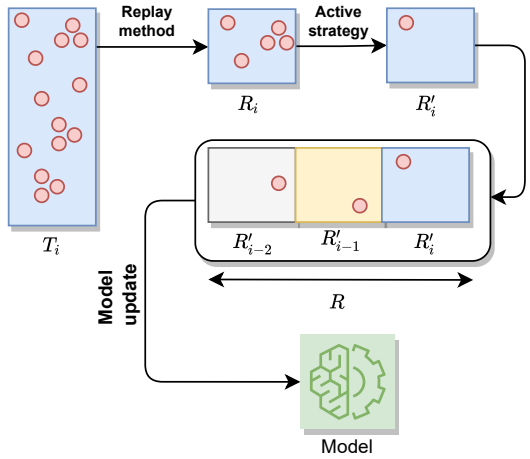


Figure 7.1: Proposed framework for active lifelong anomaly detection. Data from each concept T_i is provided to a *replay method* that yields a compact version of the concept R_i. The *active learning strategy* reduces the number of potential anomalies (represented as red circles) residing in the replay buffer and yields R'_i. All concepts are included in R, which is used for model updates.

We devise two novel distinct querying and cleaning strategies that aim to clean the buffer as much as possible without overusing budget B. While the first strategy, the Anomalous Cluster Removal strategy (ACR), analyzes the input data, the other one, the Similarity of Anomaly Scores strategy (SAS), focuses on the analysis of anomaly scores returned by the model. Both strategies try to leverage the oracle's feedback to further clean the replay buffer, removing also data points that are likely to be anomalies but were not included in the oracle's queries.

The Anomalous Cluster Removal strategy (ACR) consists of two steps. In the first step, the strategy leverages cluster data stored in the replay buffer. After that, the strategy randomly selects a given percentage of the number of clusters according to the available budget. One data point from each cluster is selected and used as a query to the oracle. If the data point from the given cluster is found to be anomalous, the whole cluster is removed from the replay buffer. The strategy exploits both diversity and similarity in the original feature representation of input data. Figure 7.2 presents the intuitive

The second strategy, Similarity of Anomaly Scores (SAS), focuses on exploiting anomaly scores. While query strategies typically focus on selecting data points with the highest anomaly score as queries, this leads to not fully exploring the spectrum of the anomaly scores provided by the model, neglecting the fact that anomalies may also present low anomaly scores, depending on the quality of the detection model. SAS strategy queries data points throughout the whole anomaly detection score range. In the second step, the strategy cleans the replay buffer from data points with anomaly scores similar to those labeled as anomalous by the oracle. We provide further details about the framework and strategies, along with formalization, in Chapter 15.

7.2 Experimental results overview

The aim of our experiments is to assess how contamination impacts anomaly detection performance and whether our proposed framework can improve performance by removing anomalies from the replay buffer.

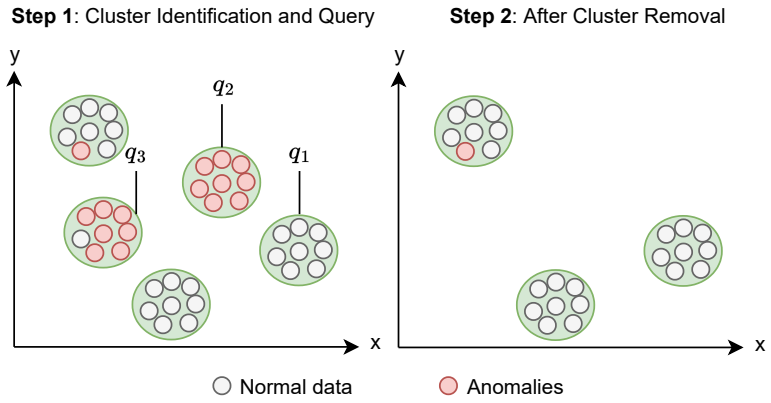


Figure 7.2: ACR strategy: Step 1 is focused on clusters identification and selection of a small number of clusters (q_1, q_2, q_3), from each of which the data point closest to the cluster center is subject to the oracle queries. Queries labeled as anomalies lead to the removal of the entire cluster to which they belong.

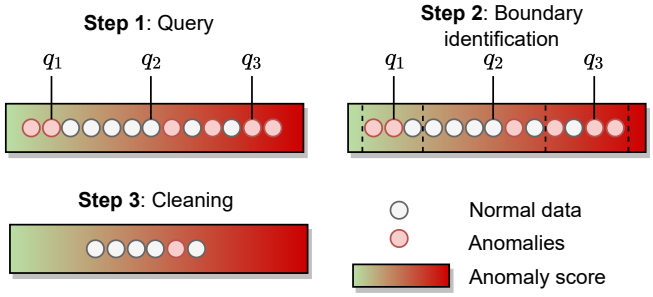


Figure 7.3: SAS strategy: Step 1 queries data samples for the oracle with significantly different anomaly score values, taking into account the full range of anomaly scores returned by the model. Step 2 identifies boundaries for the neighborhood of each sample labeled as an anomaly. Step 3 removes identified anomalies and their neighborhood from the replay buffer.

We carry out experiments on three cybersecurity datasets: *i)* NSL-KDD; *ii)* UNSW; *iii)* NGIDS; with three anomaly detection models: *i)* Autoencoder (AE); *ii)* One-Class Support Vector Machine (OC-SVM); *iii)* Local Outlier Factor (LOF). In addition to two our proposed strategies (ACR and SAS), we also include two standard query strategies for the comparison: *Random (RND)* – selecting random samples; *Highest Anomaly Score (HAS)* – selecting samples with the highest anomaly score.

Our first goal is to assess the impact of contamination on anomaly detection performance. Figure 7.4 shows the performance of all models at varying contamination rates in combination with different percentages of anomalies cleared from the replay buffer. The results show that LOF is severely affected by the contamination, and its detection capabilities are significantly crippled until most anomalies (80% – 100%) are removed. On the other hand, OC-SVM shows a consistent improvement on NSL and UNSW datasets along with percentages of removed anomalies, while it does not show a capability to deal with the NGIDS dataset. AE generally presents less consistent behavior, which we attribute to the very limited replay buffer size (500 samples per concept). Overall, our analysis shows that clearing a sufficient amount of anomalies from the replay buffer positively impacts the anomaly detection model's performance to a varying degree based on the model's characteristics. Therefore, the combination of active learning and lifelong anomaly detection has the potential to yield more robust models.

We also assess the impact of the proposed framework on anomaly detection performance. Table 7.1 shows that active strategies provide a moderately low improvement for LOF and a more significant im-

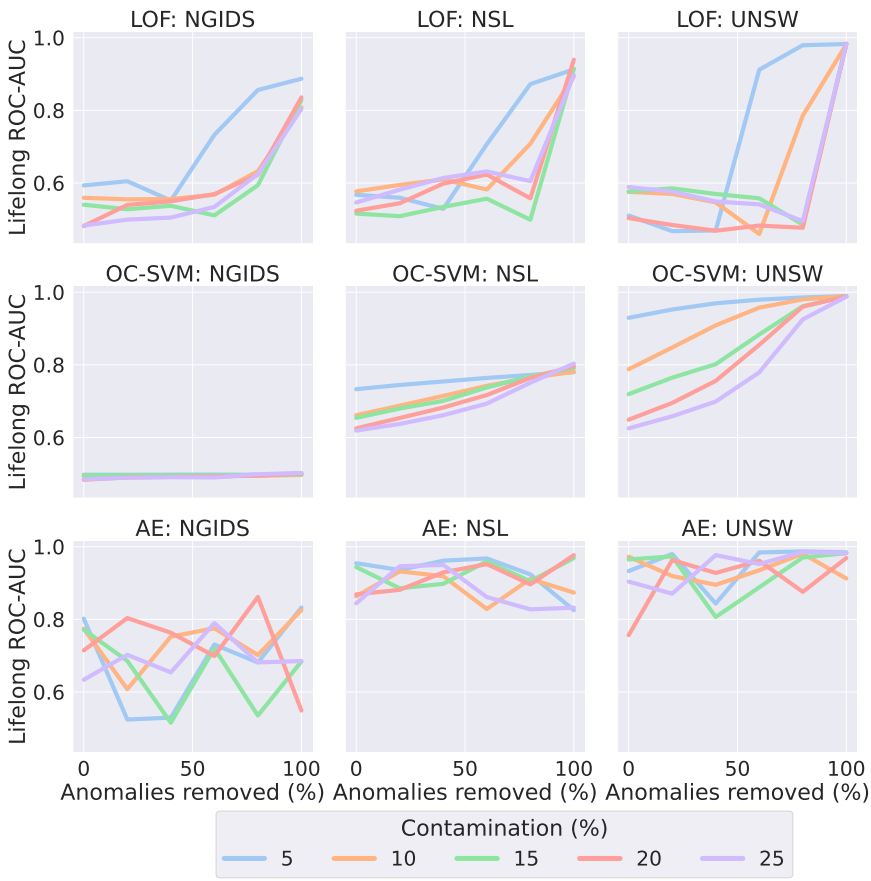


Figure 7.4: Experimental study of the impact of contamination and replay buffer cleaning capabilities on the anomaly detection performance of all methods (LOF, OC-SVM, AE) and datasets (NGIDS, NSL, UNSW). Performance is measured in terms of Lifelong ROC-AUC with varying contamination rates (5%-25%) and simulating the removal of different percentages of anomalies from the replay buffer.

provement for OC-SVM and AE. A great example of significant improvement with a very limited budget $B = 25$ is the Highest Anomaly Score (HAS) strategy for OC-SVM, which yields an improvement from 0.788 to 0.943 and from 0.719 to 0.842 in the scenarios with contamination rate $C = 0.10$ and $C = 0.15$, respectively. With the same budget, results with AE and the SAS strategy show an improvement from 0.893 to 0.978 and from 0.860 to 0.966 with contamination rate $C = 0.10$ and $C = 0.20$, respectively. We present the complete results and analysis for all datasets in Chapter 15. In general, our analysis shows that active lifelong anomaly detection can increase the longevity and robustness of models. Moreover, the proposed framework yields improvement in the detection performance in the lifelong anomaly detection problem.

We present the detailed algorithms and the comprehensive experimental study in Chapter 15.

Table 7.1: Experimental results for the UNSW dataset. Lifelong ROC-AUC for all models (LOF, OC-SVM, AE), contamination rates C, budgets B and strategies (RND, HAS, ACR, SAS). The best results for each combination of method, dataset, budget and contamination rate are shown in bold.

C	B	LOF				OC-SVM				AE			
		RND	HAS	ACR	SAS	RND	HAS	ACR	SAS	RND	HAS	ACR	SAS
0.05	0	0.511				0.929				0.893			
	5	**0.531**	0.513	0.513	0.511	0.943	**0.971**	0.950	0.947	0.931	**0.984**	0.919	0.891
	15	**0.580**	0.515	0.516	0.546	0.944	**0.988**	0.944	0.947	0.899	**0.985**	0.921	0.895
	25	0.505	0.516	0.476	**0.592**	0.943	**0.989**	0.950	0.947	0.905	**0.986**	0.907	0.879
	50	0.520	**0.526**	0.502	0.500	0.945	**0.990**	0.947	0.953	**0.983**	0.859	0.822	0.826
	100	0.511	**0.535**	0.488	0.505	0.959	**0.990**	0.960	0.960	0.892	**0.927**	0.895	0.910
0.10	0	0.576				0.788				0.893			
	5	**0.577**	**0.577**	0.532	**0.577**	0.798	**0.834**	0.803	0.792	**0.978**	0.888	0.841	0.814
	15	**0.607**	0.581	0.558	0.575	0.812	**0.912**	0.812	0.799	0.879	0.890	**0.965**	0.788
	25	0.576	0.583	**0.608**	0.568	0.809	**0.943**	0.815	0.795	0.911	0.933	0.916	**0.978**
	50	0.557	**0.598**	0.560	0.548	0.821	**0.958**	0.827	0.805	**0.981**	0.910	0.895	0.890
	100	0.558	**0.613**	0.565	0.582	0.847	**0.971**	0.852	0.824	0.979	0.915	**0.980**	0.919
0.15	0	0.577				0.719				0.860			
	5	0.599	0.577	0.571	**0.584**	0.718	**0.756**	0.748	0.726	**0.971**	0.880	0.889	0.879
	15	0.561	0.577	0.601	**0.612**	0.743	**0.805**	0.753	0.730	0.829	**0.890**	0.870	0.768
	25	0.568	**0.582**	0.572	0.580	0.733	**0.842**	0.742	0.734	0.877	0.867	0.813	**0.885**
	50	**0.608**	0.590	0.601	0.564	0.741	**0.862**	0.738	0.737	0.881	**0.897**	0.887	0.747
	100	0.577	**0.628**	0.584	0.560	0.762	**0.906**	0.772	0.747	0.887	0.910	**0.976**	0.875
0.20	0	0.503				0.649				0.860			
	5	0.521	0.504	**0.547**	0.503	0.651	**0.676**	0.671	0.645	0.787	0.816	**0.864**	0.849
	15	0.470	0.506	**0.552**	0.514	0.651	**0.732**	0.658	0.646	0.815	**0.889**	0.733	0.736
	25	0.524	0.509	**0.542**	0.509	0.682	**0.762**	0.702	0.650	0.796	0.892	0.722	**0.966**
	50	0.512	**0.513**	0.509	0.489	0.653	**0.789**	0.670	0.663	0.846	0.783	**0.968**	0.846
	100	0.519	**0.538**	0.537	0.477	0.699	**0.821**	0.706	0.679	**0.971**	0.917	0.844	0.789
0.25	0	0.589				0.625				0.739			
	5	0.525	**0.589**	0.569	0.587	0.619	**0.655**	0.623	0.631	**0.958**	0.801	**0.958**	0.790
	15	0.522	**0.592**	0.545	0.575	0.620	**0.707**	0.660	0.626	0.846	0.833	**0.963**	0.806
	25	0.555	**0.594**	0.554	0.547	0.636	**0.727**	0.637	0.623	0.808	0.814	**0.959**	0.725
	50	0.557	**0.606**	0.547	0.531	0.650	**0.751**	0.637	0.631	0.961	0.858	0.867	**0.963**
	100	0.570	**0.629**	0.532	0.532	0.674	**0.777**	0.647	0.658	0.725	**0.909**	0.779	0.882

Chapter 8

Lifelong distributed collaborative intrusion detection

An intrusion detection system (IDS) is a vital tool in safeguarding organizations and companies against hostile attacks [58]. Monitoring network traffic offers the possibility of identifying suspicious activities and, if necessary, implementing suitable defensive actions. Real-life applications involve overseeing critical infrastructure like data centers, smart grids, and banking networks [58, 46]. Additionally, it extends to everyday appliances such as IoT devices in smart homes or mobile devices, showcasing the range of its applicability [96, 90]. As a result, intrusion detection systems help ensure security on multiple levels, from personal applications to national security. Machine learning methods have been the primary area of ongoing intrusion detection exploration due to their detection capabilities that go beyond pre-defined rules and patterns [4].

Recently, we have observed an emerging shift toward constructing intrusion detection systems as distributed service-oriented architectures due to the scalability and adaptability of such approaches [78]. A similar relevant avenue of research involves collaborative intrusion detection, in which an intrusion detection system leverages insights from multiple sources and perspectives to offer a comprehensive outlook and provide more resilient detection of sophisticated attacks [42].

In our previous works on lifelong anomaly detection, described in Chapters 6 and 7, we evaluated our methods on intrusion detection datasets as one of the included domains and showcased the benefits that lifelong anomaly detection may bring in the scenarios with multiple recurring concepts. However, no lifelong learning approaches for intrusion detection leverage the advantages of multi-sourced collaboration. While all our previously presented studies on lifelong anomaly detection include intrusion detection as one of the investigated domains, they are more focused on single-source analysis.

To address the aforementioned issue, we present our two research papers focused on intrusion detection (Contribution **C3**). In our initial work, "Autoencoder-based IDS for cloud and mobile devices," we design an autoencoder-based IDS for cloud and mobile devices that implements collaborative sharing of processed data between multiple Intrusion Detection Nodes to improve the detection performance. Then, in our work "Distributed Continual Intrusion Detection: A Collaborative Replay Framework," we extend our previous idea by proposing a novel collaborative lifelong intrusion detection framework that leverages multi-sourced data. The critical aspect of the framework is a collaborative adaptive double-balanced replay algorithm, a key element offering a comprehensive overview of traffic data gathered from multiple nodes. This strategy enables the model to retain past knowledge while adjusting to a changing environment. By leveraging collective information from all nodes, our approach allows for constructing a more comprehensive model encompassing inter-node dynamics and patterns, which could facilitate recognizing normal behaviors observed previously across different nodes and strengthen the detection of abnormal patterns. We present an

overview of both works below, while the full description is provided in Chapter 16 (for "Distributed Continual Intrusion Detection: A Collaborative Replay Framework") and Chapter 17 (for "Autoencoder-based IDS for cloud and mobile devices").

8.1 Autoencoder-based IDS for cloud and mobile devices

In our preliminary work, we introduce an autoencoder-based intrusion detection system designed for both cloud and mobile environments. This system features multiple data collection points, enabling monitoring in networks like cloud-based virtual networks or across diverse networks with mobile and IoT devices. Captured network traffic records are directed to specific intrusion detection nodes responsible for carrying out the detection process. If suspicious behavior is detected, an alert regarding a potential intrusion can be sent to the respective device owner. The core of the detection process relies on an autoencoder neural network, offering significant advantages: it operates on an anomaly-based approach, is trained solely on benign samples, and exhibits robust performance. To enhance detection efficacy, we have developed time-window-based features. Moreover, we designed collaborative sharing of computed data statistics among intrusion detection nodes to improve their ability to provide a comprehensive overview of the environment and increase the robustness of the detection. The system also supports continual retraining of the module through the Training Module along with the Traffic Database.

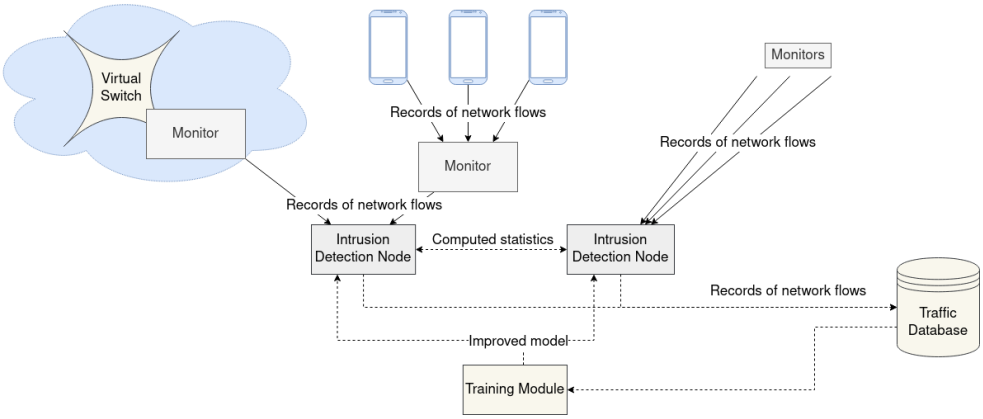


Figure 8.1: A structure of the proposed Intrusion Detection System.

We present the structure of the distributed IDS architecture in Figure 8.1. There are three main components of the proposed IDS:

- M – a set of Monitors;
- N – a set of Intrusion Detection Nodes;
- C – a set of connections between Monitors and Intrusion Detection Nodes.

The "Monitor" serves as the component responsible for collecting the data required for intrusion detection. Its responsibilities encompass preprocessing this data before transmitting it to the Intrusion Detection Node (IDN). Each IDN receives data from multiple Monitors and builds derived features based on time-window parameters. Subsequently, it shares computed statistics with other IDNs to enhance the overall detection performance. The IDN's primary function revolves around an anomaly-based detection process.

Our experimental evaluation showcases that the proposed approach can achieve high detection results. Moreover, we observe improvements introduced by new features computed across Intrusion Detection

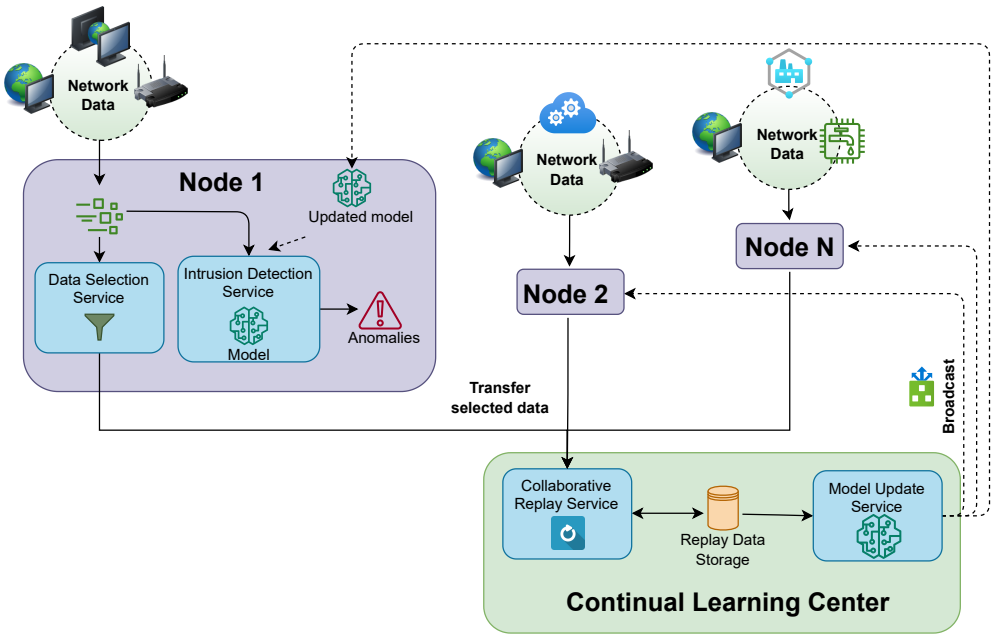


Figure 8.2: Proposed architecture for collaborative continual intrusion detection consisting of four types of services: Data Selection, Intrusion Detection, Collaborative Replay, and Model Update.

Nodes. A more detailed description of the framework, its components, and experiments can be found in Chapter 17.

8.2 Collaborative framework for distributed lifelong intrusion detection

In this section, we provide a system-level overview of our framework. Then, we explain the details of our replay strategy and describe how it manages multi-sourced data. Finally, we present an overview of our experimental results.

System-level overview

The proposed framework's architecture comprises multiple Intrusion Detection Nodes (called "Nodes" from now on) and a Continual Learning Center, each serving specific roles realized in the form of independent services. Conceptually, Nodes monitor network traffic from any type of source, such as conventional network segments, IoT devices, and cloud infrastructures. Every Node is responsible for intrusion detection in the monitored network traffic, employing a machine learning model shared across all Nodes. On the other hand, the Continual Learning Center manages model updates and their quick transfer to Nodes, ensuring the Nodes conduct intrusion detection according to the most current information available. To achieve this, the Continual Learning Center maintains a collaborative experience replay buffer constructed from selected summarized data transferred from Nodes. The architectural layout of the proposed framework is visually depicted in Figure 8.2

Focusing on details of the specific services, The *Data Selection Service* selects the key summarized network flows from individual nodes for transfer to the *Continual Learning Center*, reducing network overhead. We use Reservoir Sampling [107] as an effective method to select data points from the monitored network traffic. Essentially, it assigns probabilities to samples and selects those with the highest probability within a budget limit. It is important to note that sensitive data stays within the local Nodes, as the framework operates only summarized network flows instead of full network packets.

Selected data is transmitted to the *Collaborative Replay Service*. Replay is crucial in continuous learning strategies to retain previously acquired knowledge while updating models with new data [87]. The replay buffer summarizes data for model updates according to the replay strategy, which is crucial to ensure proper knowledge retention. The *Collaborative Replay Service* uses combined data from multiple nodes to decide which samples should be kept in *Replay Data Storage* for future model updates. More specifics on our replay strategy are in Section 8.2. The size of the replay buffer can be tailored to specific domain characteristics and available storage.

Whenever there is a significant change in network traffic, the framework updates the model to reflect relevant changes. The *Model Update Service* achieves this by updating the last model version with the updated data from the replay buffer. Notably, the framework is model-agnostic and can accommodate any machine learning-based anomaly detection model. In our case, we utilize one-class learning models as they do not rely on knowledge of anomalous patterns. The updated model is distributed to the *Intrusion Detection Service* across all *Nodes*, enabling them to leverage the newly acquired collaborative knowledge. Sharing a single model allows us to avoid the computational load of training and maintaining separate models for each node. Moreover, it gives the model a more global perspective of network traffic.

The framework capabilities are designed as separate services, clearly outlining their responsibilities. While some services focus on machine learning models (*Model Update* and *Intrusion Detection*), others prioritize handling large volumes of data (Data Selection and Collaborative Replay Service). Such design offers flexible deployment options leveraging specific hardware resources accordingly. Moreover, service-based architecture promotes the system's robustness, reliability, and resilience.

Replay strategy

In this section, we introduce our innovative adaptive double-balanced collaborative experience replay strategy. We start with the description of the settings. Let us consider a network composed of N Nodes. Each node k observes a sequence of concepts denoted as $C_1^k, C_2^k, \ldots$, where each concept represents a consistent behavior identified in network traffic. A new concept emerges when there is a significant change in the normal behavior, indicating a need for model adaptation.

We devise an experience replay strategy that implements collaborative knowledge sharing across Nodes. Data sharing has the potential to improve the representation of network traffic patterns by incorporating locally observed concepts from distinct Nodes. By collaboratively integrating this knowledge into the broadcasted model, we aim for more robust intrusion detection compared to relying on a single information source.

Our replay strategy follows a double-balanced approach, evenly distributing space among various Nodes and then equally distributing space among concepts from each Node. The equal distribution aims to ensure that nodes and concepts are equally represented within the replay buffer. This adaptive balancing occurs without prior knowledge of the number of Nodes or concepts. Figure 8.3 illustrates the outcome of our replay strategy, demonstrating rebalancing managed by the *Collaborative Replay Service*.

In a formal sense, our replay buffer comprises multiple parts determined by the number of nodes and concepts observed so far. Each part of the replay buffer R_i^k retains a summarized version of concept i from node k. We can define the entire experience replay buffer R as:

$$R = \{R_i^k \; ; \; k \in \{0, 1, \ldots, N\}, i \in \{0, 1, \ldots, c_k\}\}, \tag{8.1}$$

where c_k is the number of concepts observed at Node k.

The replay buffer maintains a fixed size, represented by a budget B, which relies on available storage resources. In lifelong learning scenarios, this budget is typically notably smaller than the observed data volume [19]. In our approach, the budget allocated for each observed concept is dynamically adjusted

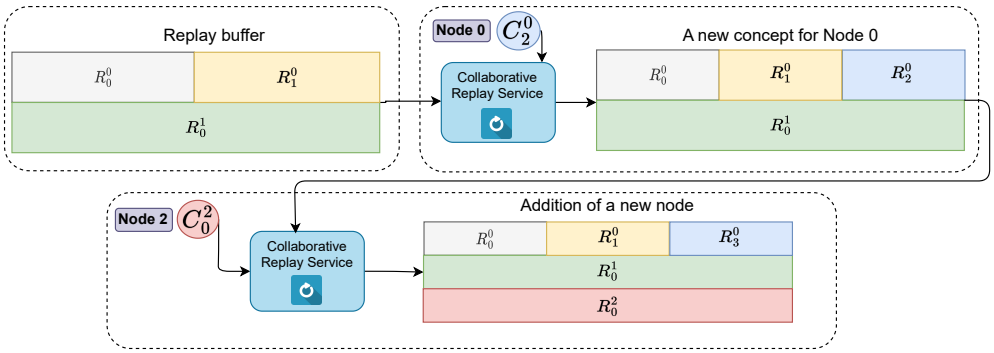


Figure 8.3: Example of our adaptive double-balanced replay strategy. When a new concept for node 0 is presented, the strategy resizes the space of the other two concepts for the same node. When a new node is added, the replay service accommodates it by resizing all existing concepts from all nodes.

in relation to the total budget B. More precisely, the budget allocated for a single concept in Node k is determined by the current number of nodes N in the network and the number of concepts c_k observed for node k:

$$\beta(N, c_k) = \frac{B}{N \cdot c_k} \tag{8.2}$$

As a consequence of such an adaptive approach, the replay buffer must be adaptively rebalanced to include every new concept. We designed two levels of rebalancing: within-node and across-nodes.

We present across-node rebalancing in Algorithm 4 that outlines the behavior of the replay upon receiving new data from the node k. The algorithm's operation depends on whether the source node k already has an allocated space in the replay buffer or is entirely new. If a new node is discovered, the algorithm accommodates it by reducing space allocated to all other nodes (lines 2-5). Subsequently, data for the newly added node is incorporated into the replay buffer using Reservoir Sampling r [107] to select data samples within the budget (lines 6-7). As a result, the algorithm increases the number of nodes in the network (line 8) and sets the number of concepts for node k to 1 (line 9). When the node k is already included in the replay buffer, the algorithm rebalances observed concepts within the node k by reducing space allocated to existing concepts (lines 11-12) and including a new concept (line 13). As a result, the algorithm increases the number of concepts present in the node k (line 14).

We present within-node rebalancing required when a new concept is observed at an existing node in Algorithm 5. Intuitively, it scales down buffer R_i^k for each summarized concept i for node k leveraging Reservoir sampling r [107] to select data samples according to budget $B^k = \beta(N, c_k)$.

The double-balanced replay strategy offers the crucial advantage of assigning equal importance to information gathered from multiple nodes. Additionally, when dealing with varying numbers of concepts assigned to different Nodes, this method prevents the emergence of dominant or vanishing nodes. Moreover, the adaptive nature of the strategy enables the complete utilization of capacity throughout the whole scenario without prior knowledge of the number of concepts, unlike non-adaptive replay strategies.

Experimental results overview

The main goal of our experiments is to assess: i) whether the knowledge-sharing capability of collaborative continual anomaly detection improves the performance of models in comparison to its non-collaborative counterpart; ii) whether our devised adaptive double-balanced collaborative replay provides good results while working in the resource-limited environment.

We present experimental results of our work in Tables 8.1 and 8.2, which show two Lifelong ROC-AUC scores for each base model (OC-SVM, COPOD, LOF, IF, VAE; see Chapter 2). We leverage CICIDS2017

Input: $\mathbf{X^k}$ – new data from node k
Input: N – the total number of nodes in the network so far
Input: $C = \{c_0, c_1, \ldots, c_N\}$ – a number of concepts observed so far for each node $0, 1, \ldots, N$
Input: R – replay buffer built so far
Output: R – updated replay buffer
Output: N – updated number of nodes in the network

if $k > N$ **then** // New node (not yet present in replay buffer)
  for $l \in \{0, 1, \ldots, N\}$ **do**
    $B^l \leftarrow \beta(N + 1, c_l)$ // Budget for each concept in node l; Eq: 8.2
    $R^l \leftarrow Alg.5(R^l, B^l)$ // Resize concepts from node l
  end
  $B^k \leftarrow \beta(N + 1, 1)$ // Budget for new node
  $R^k_0 \leftarrow r(\mathbf{X^k})$ // Add new concept with Reservoir sampling (r)
  $N \leftarrow N + 1$
  $c_k \leftarrow 1$
else // Node already present in replay buffer
  $B^k \leftarrow \beta(K, c_k + 1)$ // Budget for each concept in node k; Eq: 8.2
  $R^k \leftarrow Alg.5(R^k, B^k)$ // Resize concepts from node k
  $R^k_{c_k+1} \leftarrow r(X^k, B^k)$ // Add new concept for node k
  $c_k \leftarrow c_k + 1$
end
return R, N

Algorithm 4: Adaptive double-balanced experience replay algorithm.

Input: $R^k = \{R^k_0, R^k_1, \ldots\}$ – replay buffer for node k
Input: B^k – budget for every concept in node k
Output: R'^k – rebalanced and resized replay buffer

for $R^k_i \in R^k$ **do**
  $R'^k_i \leftarrow r(R^k_i, B^k)$ // Scale down using reservoir sampling
end
$R'^k \leftarrow \{R'^k_0, R'^k_1, \ldots\}$
return R'^k

Algorithm 5: Within-node rebalancing algorithm.

[97], UNSW [82], and NSL-KDD [102] datasets, transformed into three versions of lifelong scenarios (CC, CR, R), leveraging the algorithm proposed in Chapter 3. There are two values for each base model and scenario: one for the model without collaborative knowledge sharing (N-C) and one with collaborative knowledge sharing (C). Table 8.1 showcases the results for the Cumulative lifelong strategy, which is an unrealistic approach that stores all data encountered in the scenario. Table 8.2 presents the results achieved with our Adaptive Double-balanced Replay algorithm.

Table 8.1: Results in terms of Lifelong ROC–AUC with the **Cumulative** strategy for all datasets and base models in non-collaborative (N-C) and collaborative (C) setup.

	OC-SVM		COPOD		LOF		IF		VAE	
	N-C	C	N-C	C	N-C	C	N-C	C	N-C	C
CICIDS (CC)	0.614	0.685	0.458	0.526	0.621	0.993	0.624	0.707	0.622	0.735
CICIDS (CR)	0.625	0.732	0.388	0.488	0.611	0.993	0.645	0.728	0.660	0.825
CICIDS (R)	0.530	0.620	0.386	0.476	0.572	0.995	0.707	0.797	0.604	0.714
NSL-KDD (CC)	0.761	0.793	0.646	0.814	0.560	0.856	0.795	0.907	0.762	0.852
NSL-KDD (CR)	0.845	0.874	0.761	0.899	0.525	0.889	0.879	0.945	0.843	0.906
NSL-KDD (R)	0.925	0.954	0.418	0.857	0.526	0.898	0.929	0.970	0.907	0.955
UNSW (CC)	0.682	0.725	0.474	0.516	0.641	0.890	0.539	0.605	0.594	0.709
UNSW (CR)	0.710	0.759	0.431	0.498	0.654	0.907	0.531	0.640	0.599	0.776
UNSW (R)	0.565	0.566	0.399	0.429	0.630	0.900	0.502	0.558	0.536	0.620

Table 8.2: Results in terms of Lifelong ROC–AUC with the **Adaptive Double-balanced Replay** strategy for all datasets and base models in non-collaborative (N-C) and collaborative (C) setup.

	OC-SVM		COPOD		LOF		IF		VAE	
	N-C	C	N-C	C	N-C	C	N-C	C	N-C	C
CICIDS (CC)	0.612	0.711	0.458	0.394	0.621	0.982	0.617	0.679	0.592	0.716
CICIDS (CR)	0.623	0.781	0.388	0.302	0.611	0.984	0.643	0.754	0.632	0.722
CICIDS (R)	0.529	0.641	0.386	0.306	0.572	0.996	0.705	0.798	0.550	0.685
NSL-KDD (CC)	0.761	0.781	0.646	0.627	0.560	0.875	0.795	0.908	0.760	0.893
NSL-KDD (CR)	0.845	0.874	0.761	0.738	0.525	0.896	0.878	0.947	0.844	0.929
NSL-KDD (R)	0.925	0.950	0.418	0.359	0.526	0.894	0.929	0.971	0.907	0.964
UNSW (CC)	0.682	0.735	0.474	0.439	0.641	0.777	0.536	0.631	0.560	0.805
UNSW (CR)	0.710	0.779	0.431	0.386	0.654	0.831	0.527	0.653	0.584	0.828
UNSW (R)	0.565	0.573	0.399	0.374	0.630	0.864	0.501	0.563	0.529	0.669

The findings presented in Table 8.1 allow us to address the question whether the knowledge-sharing capability of collaborative continual anomaly detection improves the performance of models in comparison to its non-collaborative counterpart. We can observe a uniform trend across various base models and scenarios, illustrating that models with collaborative knowledge sharing (right) present better detection performance than models without knowledge sharing (left). On average, analyzing aggregated performance across the three scenarios (CC, CR, R) with the Cumulative strategy, we observe an improvement of 27.08% (from 0.58 to 0.73 for CICIDS), 20.64% (from 0.74 to 0.89 for NSL-KDD), and 18.98% (from 0.57 to 0.67 for UNSW).

Continuing our analysis, we focus on assessing the effectiveness of the proposed Adaptive Double-balanced Collaborative Replay strategy. In our experiments, the Data Selection Service selects just 200 data samples for each node and concept to transfer them to the Continual Learning Center. Moreover, we restrict experience replay buffer size to 200 samples per node (less than 2% of total data). The findings in Table 8.2 showcase that our proposed replay strategy (right) outperforms its non-collaborative counterpart (left) in 4 out of 5 base models and scenarios. On average, analyzing aggregated performance across the three scenarios (CC, CR, R) with our proposed replay strategy, we observe an improvement of 22.39% (from 0.57 to 0.70 for CICIDS), 13.77% (from 0.74 to 0.84 for NSL-KDD), and 17.61% (from 0.56 to 0.66 for UNSW). Comparing these results with the cumulative strategy (Table 8.1) allows us to measure the existing gap between the proposed learning strategy and an unrealistic upper bound without memory constraints. We observe that the gap is minimal: 5.38% for CICIDS, 6.05% for NSL-KDD, and 1.92% for UNSW. The results indicate that our collaborative adaptive double-balanced replay strategy is effective in yielding an increased detection performance while operating under limited budget constraints.

In Chapter 16, we present a more detailed discussion, along with an analysis of the efficiency of continual learning strategies in comparison to constant model updates, as well as the effectiveness of different base models.

Chapter 9

Conclusions

In this extended summary, we presented an overview of papers included in this dissertation. In Chapter 2, we provided background on lifelong learning and anomaly detection, identifying gaps that we are filling in this work. Then, in Chapter 3, we laid the foundation for lifelong anomaly detection in terms of evaluation protocol, scenario types, and metrics. Moreover, we also proposed the scenario creation algorithm and evaluated popular anomaly detection methods in lifelong scenarios. In Chapter 4, we benchmarked multiple lifelong classification methods in extensive experiments, which confirmed that replay, leveraged by us across all our lifelong anomaly detection works, is an effective knowledge retention strategy. Then, in Chapters 5-7, we addressed the problem of lifelong learning in challenging anomaly detection conditions. Specifically, Chapter 5 introduced `WATCH`, our change point detection method for high-dimensional data, and `LIFEWATCH`, a lifelong change point detection method explicitly designed to support lifelong learning methods in concept-agnostic scenarios. The idea of lifelong change point detection is then leveraged in `VLAD`, our replay-based method working in challenging concept-agnostic scenarios, as described in Chapter 6. Then, in Chapter 7, we addressed another type of challenging condition faced by lifelong anomaly detection by proposing an active learning framework that allows the minimization of the impact of training data contamination in replay-based methods. Finally, in Chapter 8, we explored the idea of collaborative data sharing in intrusion detection by proposing a distributed collaborative lifelong intrusion detection framework.

The thesis of this dissertation was as follows:

It is possible to leverage lifelong learning in anomaly detection to provide robust detection performance in environments with evolving and recurring conditions in challenging scenarios, typical for real-world domains such as cybersecurity.

We believe that the aforementioned thesis was proven, taking into account the following key contributions presented in this work:

- Bridging the gap between anomaly detection and lifelong learning (see Chapter 3; Contribution **C1**).
  - Showcasing the benefits of lifelong anomaly detection over conventional anomaly detection.
  - A characterization of lifelong anomaly detection scenarios and scenario generation algorithm that can be applied to any anomaly detection dataset, along with its open-source implementation.
  - Describing the evaluation protocol and metrics, aiming to lay the foundation for future lifelong anomaly detection research.
- `LIFEWATCH` – a change point detection method explicitly designed for challenging lifelong anomaly detection scenarios, capable of recognizing new and recurrent concepts, as well as working with high-dimensional data due to being built on top of our `WATCH` (see Chapter 5; Contribution **C2**).

- VLAD – a lifelong anomaly detection method utilizing lifelong change point detection, hierarchical memory, and replay to handle challenging concept-agnostic scenarios (see Chapter 6; Contribution **C2**).
- Active learning framework capable of improving the performance of replay-based lifelong anomaly detection in challenging scenarios with contaminated training data (see Chapter 7; Contribution **C2**).
- Evaluation study justifying the choice of replay as a base for our works in lifelong anomaly detection (see Chapter 4; Contribution **C2**).
- Showcasing the efficacy of proposed lifelong anomaly detection methods in intrusion detection (see Chapters 6 and 7; Contribution **C3**).
- Collaborative lifelong intrusion detection framework that leverages multi-sourced data to yield more robust detection of intrusions across the whole monitored network (see Chapter 8; Contribution **C3**).

Our work lays the groundwork for lifelong anomaly detection, providing a crucial starting point for further exploration. We believe this foundation will inspire other researchers to address the outstanding challenges and opportunities within this field. Future research directions include adapting various lifelong image classification strategies for anomaly detection, particularly within concept-agnostic scenarios. An intriguing idea would be utilizing our LIFEWATCH methodology as a support tool for architectural-based strategies that depend on knowing concept identifiers and changes. Further expansion of lifelong anomaly detection into new areas exhibiting dynamic behavior represents another vital investigation pathway. This direction requires the development of new, realistic datasets specifically tailored to these domains. Such datasets would greatly facilitate research in the lifelong anomaly detection field, contributing to a deeper understanding of its applicability and effectiveness across different settings. Additionally, an invaluable advancement would be the creation of an open-source library dedicated to lifelong anomaly detection. This would enable researchers to conduct reliable comparisons of new methods, fostering an environment of collaboration and innovation within the community. Tackling the identified avenues for future research would allow to create sophisticated and adaptable anomaly detection models. These models would enhance our ability to monitor and respond to evolving behaviors, bringing significant advantages to many domains and applications.

Part II

Part II: Contributions

Chapter 10

Lifelong Learning for Anomaly Detection: New Challenges, Perspectives, and Insights

In this paper, we addressed lifelong learning in the context of anomaly detection by providing a common ground for scenarios, metrics, and evaluation protocol. Moreover, we proposed an algorithm that would allow the adoption of any anomaly detection dataset in lifelong learning scenarios. Our experimental study revealed that lifelong anomaly detection is a challenging problem for commonly adopted anomaly detection methods and that lifelong learning strategies such as replay have the potential to tackle these challenges.

Contributions of Kamil Faber: In this work, Kamil Faber co-designed and developed the scenario creation algorithm. Kamil Faber carried out the experiments and was co-author of all sections and contributions of the paper.

This paper was published in the IEEE Access journal (IF 3.9; 100 points).

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 March 2024, accepted 12 March 2024, date of publication 18 March 2024, date of current version 22 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3377690

RESEARCH ARTICLE

Lifelong Continual Learning for Anomaly Detection: New Challenges, Perspectives, and Insights

KAMIL FABER[1], ROBERTO CORIZZO[1,2], (Member, IEEE), BARTLOMIEJ SNIEZYNSKI[1], AND NATHALIE JAPKOWICZ[2], (Member, IEEE)

[1]Faculty of Computer Science, AGH University of Science and Technology, 30-059 Kraków, Poland
[2]Department of Computer Science, American University, Washington, DC 20016, USA

Corresponding author: Roberto Corizzo (rcorizzo@american.edu)

The paper was supported by funds of the Polish Ministry of Science and Higher Education allocated to the AGH University, and by program "Excellence initiative - research university" for AGH University.

ABSTRACT Anomaly detection is of paramount importance in many real-world domains characterized by evolving behavior, such as monitoring cyber-physical systems, human conditions and network traffic. Current research in anomaly detection leverages offline learning working with static data or online learning focusing on constant adaptation to evolving data. At the same time, lifelong learning represents an emerging trend, answering the need for machine learning models that continuously adapt to new challenges in dynamic environments while retaining past knowledge. Although this aspect could be beneficial to build effective and robust anomaly detection models, lifelong learning research is mainly dedicated to proposing new model update strategies in image classification and reinforcement learning domains. The limited scope addressed by lifelong learning works thus far creates a gap in understanding whether such techniques and capabilities can be fruitfully exploited in anomaly detection contexts, which represents the main motivation of this paper. More specifically, anomaly detection provides unique challenges, such as an evolving normal class and limited availability of anomalies, which significantly differs from the landscape and scenarios of lifelong image classification and reinforcement learning. In this paper, we face this issue by exploring, motivating, and discussing lifelong anomaly detection, as well as providing foundations with regard to scenarios, strategies, and metrics. First, we explain why lifelong anomaly detection is relevant, defining challenges and opportunities to design anomaly detection methods that deal with lifelong learning complexities. Second, we formulate and characterize lifelong learning settings tailored for anomaly detection problems, and design a scenario generation procedure that enables researchers to experiment with lifelong anomaly detection using existing datasets. Third, we perform experiments with popular anomaly detection methods on proposed lifelong scenarios, emphasizing the gap in performance that could be filled with the adoption of lifelong learning. In summary, our efforts are directed at assessing the performance of non-lifelong anomaly detection models in lifelong scenarios and how the adoption of lifelong learning impacts their learning capabilities. Overall, we conclude that the adoption of lifelong anomaly detection is important to design more robust models that provide a comprehensive view of the environment, as well as simultaneous adaptation and knowledge retention.

INDEX TERMS Lifelong anomaly detection, lifelong learning, anomaly detection, continual learning, continual anomaly detection.

The associate editor coordinating the review of this manuscript and approving it for publication was Xiong Luo.

I. INTRODUCTION

Anomaly detection is the task of finding anomalous data instances that represent a deviation from the normal

conditions of a process [1], [2]. The capability to detect anomalous behavior is of paramount importance in many disciplines and real-world applications, such as intrusions in network traffic [3], irregular behavior in cyber-physical systems such as smart grids [4], as well as IoT environments [5], or defects in manufacturing processes [6]. The most widespread approach in machine learning is to model the normal behavior of the system and identify anomalies as data instances that significantly differ from the modeled behavior [7]. This choice reflects the limited availability of anomalies compared to the large availability of normal data, which results in the inability to model the anomaly class accurately. Moreover, it responds to the necessity of detecting anomalies with varying morphology unknown at training time [7], [8].

Most works in anomaly detection deal with the problem in an offline (batch) or online (stream) manner [1]. In evolving environments, models will become outdated and require updates, either as a full retraining stage (batch models), or as an online adaptation stage following concept drift detection (stream models) [9].

Both types of approaches are equally valid depending on the domain characteristics. For instance, offline approaches are commonly used for tasks such as lesion detection in medical images [10] and gravitational waves detection [11]. On the other hand, online approaches are common in domains characterized by a temporal dimension, such as real-time fatigue detection [12] and crowd anomaly detection [13]. Updating models allows them to adapt to the changing conditions of the normal class. However, it is noteworthy that updating the model has the side effect of gradually leading to forgetting past knowledge [14], [15]. Forgetting is a widely known phenomenon in data streams and online learning, and it is considered to be a positive feature in some scenarios as it allows models to focus on the most recent data characteristics [16]. For instance, in crowd anomaly detection [13], forgetting is suitable since it is assumed that only the people present in the current monitored environment (i.e., the most recent data) are the relevant ones to predict anomalies within that particular environment and at that particular time.

On the other hand, lifelong continual machine learning[1] research shows that forgetting is a problem that negatively affects models' performance when previously experienced conditions reoccur in the future [17], [18], [19], [20]. For this reason, lifelong learning seeks to find a balance between adapting to new knowledge while retaining past knowledge, inspired by biology, neuroscience, and computer science [14], [21], [22]. For instance, a popular example in lifelong reinforcement learning is having an agent able to learn how to play new games while not losing the ability to play previously known ones.

[1]Terms "lifelong" and "continual" are used interchangeably in existing literature. From now on, we will use the term lifelong to refer to this learning setting.

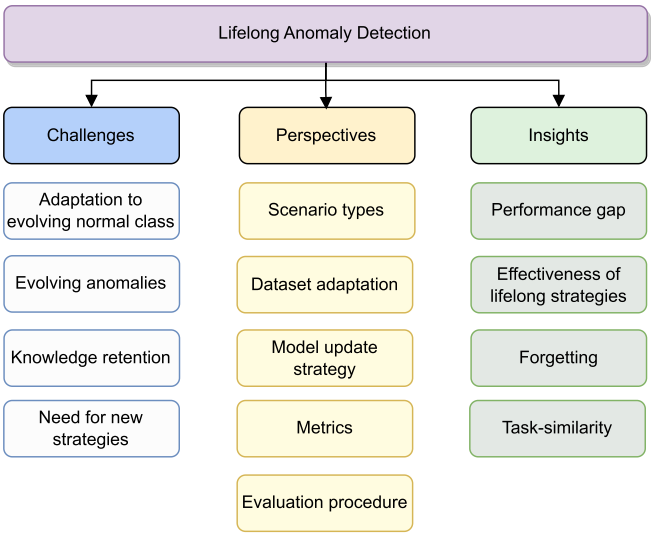


FIGURE 1. General view of lifelong learning in terms of challenges (see Section III), perspectives (see Section IV), and insights (see Section VI) examined in our paper.

Despite the emerging interest in lifelong learning, current research on this topic is mainly focused on computer vision and reinforcement learning [14], [23], [24], [25], with new trends including bridging active learning with open world learning [26] and applications to robotics [27], as well as online image classification [28]. In contrast, anomaly detection problems are still poorly explored. In this paper, we focus on lifelong learning from an anomaly detection perspective, showing that lifelong learning capabilities can bring several advantages in many real-world settings. We argue that considering them will yield more sophisticated models that can detect anomalies while adapting to changing environments and avoiding forgetting knowledge acquired in the past. One of the domains where this capability is crucial is cybersecurity [29]. For example, monitoring network traffic has to deal with dynamic conditions such as changes in the infrastructure, user behaviors, as well as new types of traffic and protocols [30]. Other examples of domains that we further describe in the paper include human condition monitoring and fault detection in industrial settings, although many more can be drawn.

Moreover, we leverage lessons learned in recent lifelong machine learning studies to showcase the current limitations of anomaly detection methods when exposed to lifelong learning scenarios. To this end, we formalize lifelong anomaly detection and devise the desiderata of models and scenarios, building the foundations for future work.

Our analytical scope focuses on two main research questions:

- **RQ1:** Do lifelong scenarios impact the performance of non-lifelong anomaly detection models?
- **RQ2:** Does the adoption of knowledge retention capabilities of lifelong learning provide a valuable improvement in the learning capabilities of existing anomaly detection models in complex lifelong scenarios?

With the first research question, we aim to assess whether current anomaly detection models are effective in lifelong learning scenarios and if there is a gap that needs to be addressed. With the second question, we aim to verify whether adopting lifelong learning strategies can be beneficial in anomaly detection contexts.

The contributions of this study can be summarized as follows:

- Bridging the gap between anomaly detection and lifelong learning, presenting the benefits of lifelong anomaly detection over conventional anomaly detection, which paves the way for new methods that are more robust in real-world domains;
- Devising a characterization of anomaly detection scenarios from a lifelong learning perspective, which sheds light on how to design and build more challenging benchmarks for anomaly detection;
- An open-source[2] implementation for scenario generation method, which can be applied to any anomaly detection dataset, facilitating wider adoption of lifelong learning in anomaly detection settings.
- Evaluating popular anomaly detection methods on our proposed lifelong scenarios, emphasizing challenges and limitations that occur in this setting.

We summarize challenges, perspectives, and insights in Figure 1. Our contributions allow us to highlight the potential of lifelong anomaly detection as a new, promising research direction. The scenarios we designed and the experimental results we obtained in our study help us showcase new challenges, perspectives, and insights brought by lifelong learning research in the context of anomaly detection. By doing so, we aim to increase awareness of the potential of lifelong anomaly detection while rationalizing and providing foundations with regard to scenarios, strategies, and metrics. Our study aims to streamline the adoption of lifelong anomaly detection for researchers and practitioners.

The paper is structured as follows. Section II summarizes related works in lifelong learning and anomaly detection. Section III explores the transition from conventional to lifelong anomaly detection. Section IV describes the proposed lifelong scenario design procedure. Section V discusses the experimental results obtained in our study. Finally, Section VII concludes the paper outlining directions of interest for future work.

II. BACKGROUND

In this section, we first briefly introduce lifelong learning along with the most popular types of methods. Second, we provide an overview of the current landscape of anomaly detection works.

A. LIFELONG LEARNING

Lifelong learning is a continuous process in which a series of different problems, defined as tasks, are presented to a learning method over time [14]. In the most common lifelong learning settings, image classification, a few scenario types built upon task characterization have been proposed. The most popular ones are task-incremental, class-incremental, and domain-incremental [31]. Both class-incremental and task-incremental scenarios provide the model with new, previously unseen classes that need to be incorporated by the model [32], [33]. On the other hand, domain-incremental scenarios provide new distributions of already known classes [34]. Emerging trends involve online lifelong learning [35], [36], simultaneous adaptation in terms of new instances and new classes [37], and repetition of observed tasks [38].

In the lifelong setting, the general goal for the learner is to be able to pick up new skills, adjust to newly presented tasks, and draw on previously learned information to tackle both new obstacles and the recurrence of previously seen tasks [39]. The key difference between lifelong learning and incremental/online learning is that, in lifelong learning, the attention is not solely focused on adaptation but also on knowledge retention and a model's ability to simultaneously handle all tasks, avoiding forgetting [18]. To this end, lifelong learning strategies are inspired by diverse disciplines, including neuroscience and biology [14]. Lifelong learning approaches proposed thus far fall into three main categories.

Regularization-based strategies work by introducing constraints on weight updates during the incremental training of neural networks. One of the initial ideas is to prevent updates for weights learned on previously trained tasks [40].

Another approach is to freeze the first layers in the model architecture to mitigate forgetting previous tasks while leaving the last fully connected layer unfrozen to be updated with new tasks [41]. On the other hand, methods such as EWC [18] and LWF [42] modify the regularization loss using a knowledge distillation to prevent drastic changes in already learned weights, which are important to solve previous tasks.

Dynamic architectures strategies adaptively manipulate the model architecture during the learning process. Methods usually expand the network by adding new neurons or layers as they encounter new tasks [43].

More efficient methods augment dynamic adaptation with pruning capabilities, which keep model capacity under control by removing insignificant weights. Popular examples are PackNet [44] and Winning Subnetworks [45], which splits the model into independent sub-networks, each specialized in addressing a different task. This type of approach is also referred to as forget-free, which holds only under assumptions such as the availability of task labels and unlimited capacity.

Replay-based strategies ensure that the knowledge from previously seen tasks is taken into consideration by the updated model by recurringly incorporating a summarized version of data from previous tasks in model updates. The most standard replay-based techniques focus on preserving knowledge by storing data samples from previously learned tasks in a memory buffer and replaying them during model update [14].

[2]https://github.com/lifelonglab/lifelong-anomaly-detection-scenarios

There are a few strategies devised to select the most relevant samples for every task, keep the replay buffer size compact, and, in turn, limit resource usage [46]. The second category leverages generative models to generate artificial data samples from previous tasks each time the model is updated [47]. By doing so, generative replay eases the burden of storing data samples, reducing the impact of memory occupation that affects conventional replay strategies.

B. ANOMALY DETECTION

We now turn our attention to anomaly detection. As we noted in Section I, anomaly detection has become crucial for decision support in many domains. For instance, the work in [5] emphasizes the importance of anomaly detection in IoT environments, such as transportation systems, health care systems, smart objects, and industrial systems. Similarly, anomaly detection in log sequences is critical for ensuring operational and security integrity in heterogeneous systems [48]. Another interesting example is Industry 4.0, with an example of 3D printing [49], where early identification of malfunctions is fundamental to limit economic losses.

In addition to the online *vs.* offline distinction mentioned in Section I, anomaly detection methods can also be characterized as supervised, semi-supervised, and unsupervised [7], [50], based on data and labels availability.

Supervised methods require labels for both normal and anomaly classes. They also need to be concerned about the class imbalance problem, and they are generally limited by the fact that they can only identify known anomalies rather than discover new ones. Semi-supervised methods are trained using exclusively normal data, and try to identify anomalies in unseen data based on their difference with respect to the learned data distribution of the normal class. Unsupervised methods are different in that they make no assumptions about labels in training data and are simply data-driven, i.e., they fit the model based on all the available unlabelled data. Works in the literature [50] suggest that semi-supervised methods should be preferred if enough labeled normal data is available in order to achieve more robust models.

Due to the peculiarities of the learning settings, semi-supervised and unsupervised methods typically entail one-class learning models. Relevant examples of one-class learning anomaly detection methods are: *i)* Variational Autoencoder (VAE) [51], a neural-network reconstruction-based model with generative capabilities. The model is trained in a one-class manner by minimizing reconstruction error on training data, and the reconstruction error is used as an anomaly score; *ii)* One-Class Support Vector Machine (OCSVM) [52], which provides anomaly scores comparing new data with a hyperplane-based decision boundary learned during the training stage; *iii)* Local Outlier Factor (LOF) [53], which yields an anomaly score based on the ratio between the local density of new data samples with respect to the average local density of its nearest neighbors; *iv)* Isolation Forest (IF) [54], which provides ensembles of trees and considers the length of the path from root to leaf to determine the anomaly score of new samples: a shorter (or longer) path means that a data point is more (or less) likely to be an anomaly; *v)* Copula-based anomaly detection (COPOD) [55], which predicts the degree of ''extremeness'' of data samples based on tail probabilities of an empirical copula, a multivariate cumulative distribution function.

However, while these methods are well-established and perform well in a wide number of scenarios, they do not provide simultaneous knowledge retention and model adaptation, lacking lifelong capabilities. We can observe that recent research works started addressing lifelong anomaly detection. Examples include the adoption of meta-learning to estimate parameters for multiple tasks in one-class image classification [56], transfer learning in video anomaly detection [57], change-point detection coupled with memory organization [58], [59], and leveraging user feedback to improve model performance [60], [61].

Despite the clear advantages that lifelong learning could provide in anomaly detection methods, the number of published works is still rather limited. We attribute this scarcity to the novel and emerging nature of the subject and to the lack of established practices, protocols, and guidelines. Our study attempts to fill this gap by increasing the awareness of the potential of lifelong anomaly detection, while rationalizing and providing foundations with regard to scenarios, strategies, and metrics, which foster a simplified adoption of the lifelong learning framework for researchers and practitioners.

III. FROM ANOMALY DETECTION TO LIFELONG ANOMALY DETECTION

In this section, we start by analyzing the challenges arising in dynamic real-world scenarios and emphasizing the limitations of currently adopted anomaly detection approaches. Second, we discuss the advantages lifelong anomaly detection could bring to the anomaly detection landscapes.

A. WHEN IS NON-LIFELONG ANOMALY DETECTION NOT ENOUGH?

Offline anomaly detection has shown to be useful in different applications where it is possible to gather and process background data, such as post-incident analysis [50], breast cancer detection [10] and gravitational waves detection [11]. However, offline models are not sufficient for many real-world dynamic applications as they do not consider any change in the normal class.

Online anomaly detection methods partially address this limitation, providing learning systems with continuous updating capabilities. However, the underlying assumption is that only the most recent information is required to maintain satisfactory performance on the anomaly detection task. In this context, forgetting is a desirable property that allows the model to prevent obsolescence [62]. This behavior is considered to be sufficient to deal with many dynamic learning settings such as crowd anomaly detection [13]

and fatigue detection [12]. It is also possible to detect whether concept drift occurred by monitoring the statistical properties or the model's error rate, and updating anomaly detection models to reflect the most recent conditions of the environment [62]. However, methods coupled with concept drift detection also follow the assumption that only the most recent data is relevant for the anomaly detection task. As a result, these methods are prone to forgetting past knowledge, which is a shortcoming in domains with recurring tasks.

Many real-world domains may greatly benefit from the adoption of lifelong anomaly detection, as they are inherently characterized by dynamic and quickly evolving conditions, as well as recurring conditions. These challenges require model capabilities that foster simultaneous adaptation and knowledge retention. In the following, we describe three out of many possible real-world domains where such model capabilities are required.

First, monitoring human conditions to detect harmful states must be able to deal with many human activities, each presenting a unique definition of the normal class. In this setting, new life habits bring new activities that can be assimilated as tasks to be learned by the model (e.g., jogging, characterized by a high heart rate that should not be considered anomalous behavior). Forgetting activities carried out in the past is not acceptable in this setting and brings a number of practical disadvantages, which are systematically described in Section III-B.

Another example is the detection of intrusions in a cloud environment, which requires the ability to deal with a dynamic environment where multiple virtual servers (cloud instances) are added or removed over time. Such instances have different characteristics of normal behavior that depend on active services and user interactions. The system must be able to adjust and detect anomalies in traffic patterns in new cloud instances, but, at the same time, it should not decrease its performance when analyzing traffic from already monitored cloud instances.

Looking at a different domain, the identification of faults and malfunctioning in cyber-physical systems, such as water treatment plants or smart grids, is characterized by a very dynamic environment with multiple operating conditions and different uncontrollable inputs (e.g., geophysical factors in nature) that change over time. Moreover, components also age over time or are replaced with other components with different specifications. In this context, the model should be able to deal with tasks corresponding to different operating conditions.

All these domains are characterized by challenges such as a number of evolving emerging conditions that require prompt model adaptation, as well as recurring conditions that require the ability to preserve the knowledge of previously observed conditions. This duality creates the ideal conditions for the adoption of lifelong anomaly detection. However, to verify this assumption, it is important to assess whether lifelong scenarios impact the performance of non-lifelong anomaly detection models **(RQ1)**, and whether adopting knowledge retention capabilities actually results in an improvement for such models in lifelong scenarios **(RQ2)**.

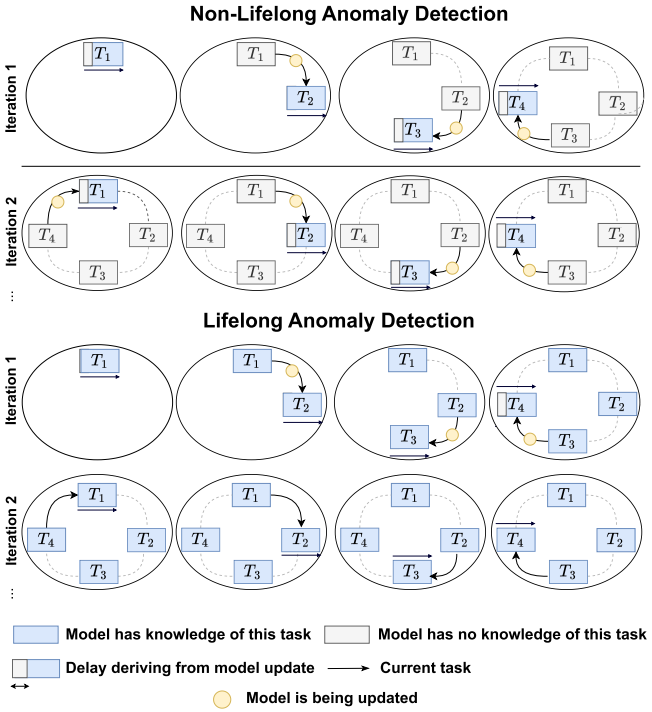


FIGURE 2. A scenario with four recurring tasks (T_1, T_2, T_3, T_4). Conventional anomaly detection requires constant model updates and results in detection delays. Lifelong learning mitigates this burden by retaining knowledge of tasks.

B. LIFELONG ANOMALY DETECTION TO THE RESCUE

Figure 2 shows a representative scenario that compares conventional anomaly detection with model updates to lifelong anomaly detection. In the second iteration, lifelong anomaly detection does not require model updates after a recurrence of each task. In contrast, conventional anomaly detection keeps updating the model, resulting in detection delays, i.e., false predictions, until the model has incorporated the new task. Moreover, a scenario with 100 iterations would require just 4 model updates for lifelong anomaly detection *vs.* 400 model updates for conventional anomaly detection, during which detection delays will occur. Many real-world scenarios with recurrence could be mapped to it, including sequences of human activities, geophysical phenomena such as weather patterns, and operating conditions of cyber-physical systems.

In Figure 3, we show a comparison between the conventional anomaly detection approaches and a counterpart for anomaly detection that entails lifelong learning. In this example, normal class data evolves over time in a task-sequential manner. As new tasks are presented, the model is updated – either in an online manner, possibly following concept drift detection, or in a lifelong learning manner. It can be observed that the non-lifelong anomaly detection with a model update approach entails forgetting as a way

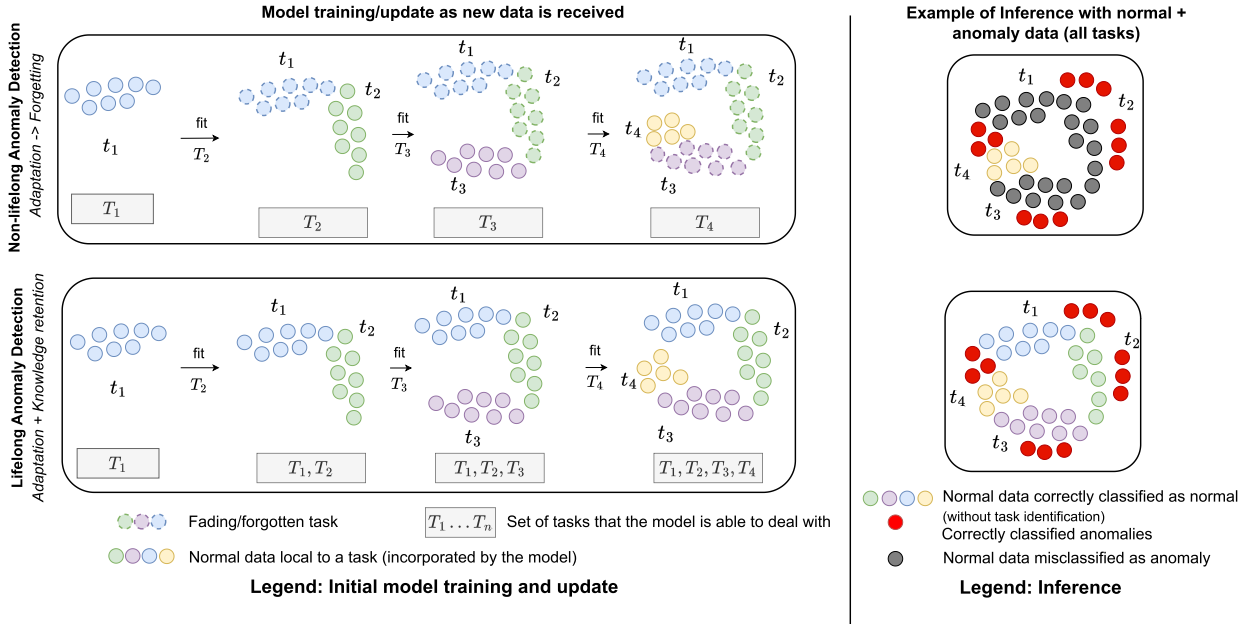


FIGURE 3. Comparison of training/update and inference for non-lifelong and lifelong anomaly detection in the scenario with four tasks (T_1, T_2, T_3, T_4). In non-lifelong anomaly detection, the model forgets the previous tasks as soon as a new task is learned (left – top). In contrast, the lifelong anomaly detection model aims to retain knowledge of all tasks (left – bottom). This characteristic has a serious impact on the model's behavior during inference (right). In non-lifelong anomaly detection, after learning task T_4, the model misclassifies data from previous tasks as anomalous since it considers only data from the current task as normal behavior (right – top). On the other hand, the ideal lifelong anomaly detection model retains the knowledge of all tasks, preventing the misclassification of normal data from previous tasks as anomalous (right – bottom). This difference in behavior between non-lifelong and lifelong anomaly detection may lead to a discrepancy in their performance scores. .

to focus on the most recent data, leading to mistakenly classifying normal data from previous tasks as anomalous. On the other hand, the lifelong learning approach for model update aims at adapting the model to incorporate new tasks without forgetting previous tasks. The advantage is to exploit knowledge from the combination of the different tasks to provide a more comprehensive and robust anomaly detection model, which correctly identifies normal data from all tasks as normal behavior.

Overall, even though the conventional anomaly detection with model update approach would be, in principle, able to re-learn previously forgotten tasks as they reoccur, this has several drawbacks, as indicated above. From a practical viewpoint, frequent model updates require additional computational resources in the presence of highly recurring tasks. Moreover, delays in the responsiveness of the model caused by the time difference between the appearance of a recurring task and updating the model once concept drift is detected can be costly in terms of false or missed detections. We aim to verify this assumption throughout an experimental analysis involving non-lifelong anomaly detection models exposed to lifelong learning scenarios (**RQ1**).

By analyzing the challenges pertaining to the lifelong anomaly detection learning setting and by generalizing the examples presented in Figure 2 and Figure 3, we identify the following drawbacks of conventional anomaly detection with constant model updates:

- By forgetting past knowledge and only adapting to the new normal class distribution, the system may trigger a large number of false positives when recurring patterns are presented, leading to a dramatic decrease in performance until a retraining phase is undertaken;
- Inability for the models to leverage skills learned in the past in combination with recent skills to solve tasks in a more compelling way;
- Experimental settings that are too simplistic to reflect the complexity of many real-world challenges, which usually involve the appearance of new conditions and the recurrence of old conditions, requiring a more comprehensive evaluation across all tasks;
- A consistent use of computational resources for data processing and model training to deal with recurring tasks;
- From a theoretical viewpoint, the model will not provide a comprehensive view of the environment without the possibility to leverage task similarity and knowledge transfer across a combination of tasks to solve every single task.

These drawbacks make current anomaly detection models rather simplistic in comparison to sophisticated human-level intelligence when faced with complexities and challenges brought by lifelong anomaly detection scenarios (**RQ1**). Intuitively, humans are exposed to different aspects of reality, building up skills incrementally throughout their lifespan and improving their general knowledge base. Introducing similar

capabilities in models should allow them to provide a more sophisticated behavior that translates into more reliable and accurate predictions [63] **(RQ2)**.

In practical terms, by leveraging discoveries in biology, neuroscience, and computer science, lifelong learning enables the possibility of incorporating comprehensive knowledge in models. Examples include the adoption of task similarity, curriculum learning, yielding positive forward and backward transfer across different tasks, as already demonstrated in image classification [64], object detection [14] and reinforcement learning problems [23].

Overall, we argue that the adoption of lifelong learning in anomaly detection would yield the ability to consider the combination of all these aspects, providing more complex learning strategies that lead to more informed decisions. Instead of constantly forgetting and learning each individual condition, an ideal model could leverage past knowledge to retain performance across all conditions. We aim to verify this assumption by designing experiments that uncover whether the adoption of lifelong learning knowledge retention strategies can be beneficial for non-lifelong anomaly detection models **(RQ2)**.

In summary, the minimal set of advantages for the adoption of a lifelong anomaly detection approach includes capabilities such as: *i)* simultaneous adaptation and knowledge preservation; *ii)* inference that exploits a more comprehensive knowledge of the domain or environment at hand; *iii)* resource-savvy model updates compared to conventional anomaly detection methods with constant model updates; *iv)* more realistic experimental settings and evaluation schemes that consider all tasks in combination.

IV. LIFELONG ANOMALY DETECTION: SCENARIOS AND EVALUATION PROTOCOLS

Given the novelty of lifelong anomaly detection, as shown by the limited availability of research works on the subject, it is important to devise procedures and guidelines to standardize its adoption. This section aims to provide new perspectives on how to address lifelong learning challenges in anomaly detection. To this end, we devise a categorization of lifelong learning scenarios, provide a scenario creation algorithm, and evaluation protocols that can guide researchers interested in the problem.

A. LIFELONG LEARNING SCENARIOS: AN ANOMALY DETECTION PERSPECTIVE

Current lifelong learning approaches are focused on classification tasks, where *tasks* are defined as sets of classes (e.g., in the MNIST dataset, any combination of two classes among its 10 classes), and the learning workflow encompasses a sequence of n tasks $T = t_1, t_2, \ldots, t_n$ where the model is challenged to learn new tasks without forgetting previous tasks.

Another important element of comparison is the type of *learning scenarios*. We recall that lifelong image classification usually describes three types of scenarios: task-incremental, class-incremental, and domain-incremental [31]. These scenarios differ based on the availability of task labels and task boundaries. For all scenarios except domain-incremental, in the simple classification example mentioned above (MNIST), task labels identify a specific subset of 10 classes currently being presented to the model, whereas task boundaries identify the beginning/end of such task. On the other hand, in a domain-incremental scenario, tasks represent a new distribution of already known classes, where task labels identify specific distributions and task boundaries indicate the moment when distribution changes. The availability of the task labels and boundaries depends on the degree of available domain knowledge [65]. A more detailed description of learning scenarios in lifelong classification may be found in [31]. It is worth noting that emerging scenarios are being proposed to account for the limitations of previously existing ones. One example is the consideration of the temporal dimension in online lifelong learning scenarios [66].

The notion of a task in anomaly detection clearly differs from the conventionally adopted definition in image classification since we usually deal with two classes: normal and anomaly. Tasks in this context represent various aspects of the normal class, which is expected to evolve over time. Moreover, the normal class may also change its role depending on the context, i.e., what is normal in one context can be anomalous in another, further increasing the complexity of the problem. To differentiate lifelong anomaly detection setting, we define a self-consistent behavior[3] of the normal class, alongside the specific anomalies occurring with it, as a *concept*. For instance, in monitoring human conditions to detect harmful states, the entire normal class can be thought of as a set of concepts: resting, jogging, and eating, all presenting different characteristics. In lifelong anomaly detection, multiple consistent behaviors of the normal class are presented over time instead of new classes as in class and task-incremental scenarios, and we are focused on the evolution of a single normal class instead of the evolution of all classes as in domain-incremental scenarios. For example, a high heart rate can be considered an anomaly in resting conditions but is expected during jogging, and therefore it does not represent an anomaly in this context. Therefore, to deal with the inadequacy of lifelong image classification scenarios in the context of anomaly detection, we define distinctive scenarios for this setting. Following the example of anomaly detection in human conditions, *concept identifiers* define a consistent behavior of the normal class (a specific activity), whereas *concept boundaries* represent explicit information on whether the currently analyzed concept (a specific activity) has changed. Concept identifiers and concept boundaries may correspond to task labels and task boundaries, respectively, in lifelong image classification.

[3] A behavior could correspond to a new distribution, change of a performed activity, or a new state of the environment, depending on the specific analytical context considered.

Based on this consideration, we identify the following learning scenarios in increasing order of complexity:

- *Concept-aware*: Known concept identifier and concept boundaries.
- *Concept-incremental*: Unknown concept identifier but known concept boundaries.
- *Concept-agnostic*: Unknown concept identifier and concept boundaries

In reference to our example of anomaly detection in human conditions, a concept-aware scenario implies that the model is aware of the currently processed activity and its lifespan (at both training and inference time). On the other hand, a concept-incremental scenario only provides an indication that a change of activity has occurred without any identifying information about the specific activities. Finally, a concept-agnostic scenario is the most challenging, as it does not provide any supporting information about the current activity being performed and its lifespan. These notions are general and can be adopted in any domain.

B. SCENARIOS DESIGN

In the following, we propose a scenario design procedure that applies to most datasets and enables researchers and practitioners to transition their current scenarios and evaluation setup toward lifelong anomaly detection.

Algorithm 1 presents a general pseudo-code for scenario design. It requires users to define a few parameters, which determine the creation of diverse scenarios: the number of desired concepts **c**, Normal (**N**), and Anomaly (**A**) data from a given dataset, and three functions: ϕ, γ, and λ. The algorithm leverages concept creation functions ϕ and γ to create normal and anomaly concepts based on normal and anomaly data, respectively. Their goal is to transform the original dataset into self-consistent sets of data points having common characteristics. An example implementation for ϕ and γ is a clustering algorithm of choice, based on user preferences. The assignment function λ matches each normal concept with an anomaly concept, leading to a combined concept containing one normal and one anomaly concept, which allows for model training and evaluation. The sequence of these combined concepts defines the complete scenario. An example implementation for λ is mapping an anomaly cluster to its closest normal cluster.

Focusing on Algorithm 1, first, we create concepts for the normal class through a concept creation function ϕ (Line 1). The concept creation function can leverage any aspect or feature value that allows us to delineate the boundaries of one concept. Second, we create concepts for the anomaly class through anomalous concepts creation function γ (Line 2). Third, for each normal concept C_{N_i}, we select a corresponding anomaly concept C_{A_j} using a function λ (Line 3-5). The combination of C_{N_i} and C_{A_j} is a concept added to the lifelong scenario (Line 6). Each time a concept is built, the selected anomaly concept C_{A_j} is removed from the set of available anomaly concepts C_A so that it appears only once throughout the scenario (Line 7). The algorithm returns the resulting scenario as a sequence of concepts (Line 9), each of which may need to be separated into training and evaluation data depending on the learning settings, e.g., unsupervised or semi-supervised.

Algorithm 1 Scenario Design Protocol

Input: c – Number of desired concepts
Input: **N**, **A** – Normal/Anomaly data
Input: ϕ – Concepts creation function for normal data
Input: γ – Concepts creation function for anomalies
Input: λ – Assignment function
1 $C_N \leftarrow \phi(N, c)$ // Create concepts $\{C_{N_0}, C_{N_1}, \ldots, C_{N_c}\}$
2 $C_A \leftarrow \gamma(A, c)$ // Create concepts $\{C_{A_0}, C_{A_1}, \ldots, C_{A_c}\}$
3 $T \leftarrow \emptyset$ // Result scenario
4 **for** $C_{N_i} \in C_N$ **do**
5 $\quad j \leftarrow \lambda(C_A, C_{N_i})$ // Match anomaly-normal concepts
6 $\quad T \leftarrow T \cup (C_{N_i}, C_{A_j})$ // Add concepts to scenario
7 $\quad C_A \leftarrow C_A - C_{A_j}$ // Remove used anomaly concept
8 **end**
9 **return** T

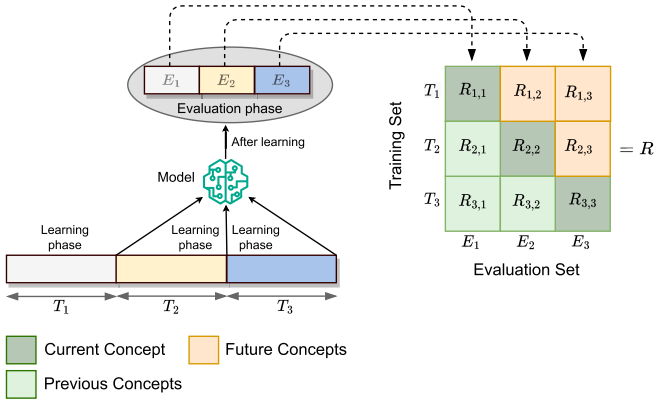


FIGURE 4. Lifelong evaluation protocol. The model handles a sequence of concepts $i = 1, 2, 3$. For each concept i, the model is trained on training set T_i (learning phase). After each learning phase, the evaluation phase is triggered, where the model anomaly detection performance (in terms of ROC-AUC) is evaluated on all testing sets E_j from all concepts (previous, current, and future). The evaluation protocol creates a matrix R, in which the entry $R_{i,j}$ represents model performance in terms of ROC-AUC on concept j after learning concept i. This matrix is used to compute final metric values, such as Lifelong ROC-AUC, BWT, and FWT.

The proposed method allows us to create diverse scenarios depending on the user selection of ϕ, γ, and λ. For example, it is possible to leverage k-Means to cluster normal and anomaly concepts (leveraging ϕ, and γ functions) and assign each anomaly concept to the closest normal concept (leveraging λ function). Alternative scenarios can be designed by customizing ϕ, γ, and λ, paving the way for a wide range of possible scenarios. To simplify this task, our code is publicly available and can be used to easily generate scenarios for any dataset chosen

Algorithm 2 Pseudo-Code of the Evaluation Protocol

Input: **T** – Sequence of N training sets
Input: **E** – Sequence of N testing sets
Input: L – Learning model
Input: ρ – Evaluation function computing ROC–AUC
1 $R_{N\times N} = \{\}$ // Initialize results matrix NxN
2 **for** $T_i \in$ **T do**
3 $L \leftarrow$ update L with T_i // Train/update model
4 **for** $E_j \in E$ **do**
5 $R_{i,j} \leftarrow \rho(L, E_j)$ // Evaluate L on E_j
6 **end**
7 **end**
8 **return** R

by users: https://github.com/lifelonglab/lifelong-anomaly-detection-scenarios. We note that different choices of ϕ and γ may lead to concept imbalance, i.e., some concepts may present significantly fewer samples than others. This situation may be problematic in some settings, or exacerbate the learning complexity for some anomaly detection models. In these cases, consideration of strategies for imbalanced learning such as resampling [67], cost-sensitive learning, and special-purpose algorithms [68]. It is worth noting that, in our experiments, we did not experience high imbalance ratios for concepts generated with our protocol.

C. MODEL EVALUATION

Lifelong learning scenarios require a continuous evaluation across all concepts. To realize this goal, we adopt a lifelong learning evaluation protocol that considers the performance of all concepts across a learning scenario.

Algorithm 2 provides a general overview applicable to a vast number of use cases. Without loss of generality, the protocol can be modified to accommodate specific requirements, e.g., recurring concepts, time-based concepts, unsupervised learning, etc. Moreover, our protocol can support any base model of choice, and any data preprocessing step, such as data augmentation and missing data treatment, to deal with specific data challenges.

First, the evaluation protocol initializes a matrix R to accommodate anomaly detection results for specific tasks (Line 1). Second, the protocol iterates over training sets for all concepts (Line 2). For each concept, the model is trained/updated (Line 3) and evaluated on all testing sets for all concepts (Lines 4-5), i.e., previous, current, and future concepts. Our protocol yields a matrix R, where entries $R_{i,j}$ define the ROC-AUC metric of the model evaluated on concept j after learning concept i. A graphical representation of this protocol is shown in Figure 4.

The matrix R can be used to directly compute lifelong learning metrics, such as backward and forward transfer.

These metrics allow us to assess model behavior more extensively than standard performance metrics by taking into account the model's performance on different concepts (previous, current, and future).

Inspired by [69], we propose a **Lifelong ROC-AUC** – a lifelong variant of ROC-AUC that can adequately assess models' performance on all concepts after learning every new concept, instead of models' performance on just a single concept. It is defined as:

$$\text{Lifelong ROC-AUC} = \frac{\sum_{i\geq j}^{N} R_{i,j}}{\frac{N(N+1)}{2}} \tag{1}$$

The metric is computed considering previously learned concepts, including the current concept, which corresponds to averaging over $\frac{N(N+1)}{2}$ entries from lower triangular. We favor ROC-AUC over threshold-dependent metrics such as Precision, Recall, and F–Score, since it allows us to evaluate the model's performance more comprehensively. ROC–AUC may be swapped with other metrics of choice without impacting the validity of the protocol.

Backward Transfer for ROC–AUC (*BWT*) measures the impact of learning new concepts on the performance of all previously learned concepts. Negative backward transfer suggests that the model is prone to forgetting. A strongly negative value is also sometimes regarded as catastrophic forgetting. On the other hand, positive backward transfer suggests that learning new concepts benefits models' performance on previously learned concepts. Backward transfer is computed over all concepts as:

$$BWT = \frac{\sum_{i=2}^{N}\sum_{j=1}^{i-1} R_{i,j} - R_{j,j}}{\frac{N(N-1)}{2}} \tag{2}$$

The impact of learning each concept on the model's performance on future concepts is measured by **Forward Transfer for ROC–AUC** (*FWT*). Forward transfer can also be thought of as the zero-shot model performance on future concepts since it assesses model performance on unseen concepts. It partially depends on concept similarity (task similarity) and the model's knowledge transfer ability. It is computed as:

$$FWT = \frac{\sum_{i<j}^{N} R_{i,j}}{\frac{N(N-1)}{2}} \tag{3}$$

It is noteworthy that the protocol slightly differs based on the learning setting. Specifically, in concept-aware and concept-incremental scenarios, batches T_i (training) and E_i (evaluation) correspond to the single $i-$th concept. As for concept-agnostic settings, a batch does not necessarily correspond to a single concept since the setting assumes that no explicit concept boundaries are provided to the lifelong algorithm. As a result, the evaluation may require considering multiple batches as belonging to the same concept or a single batch including data for more than one concept.

V. EXPERIMENTS

Our experiments are directed at answering two main research questions: *i)* Do lifelong scenarios impact the performance of

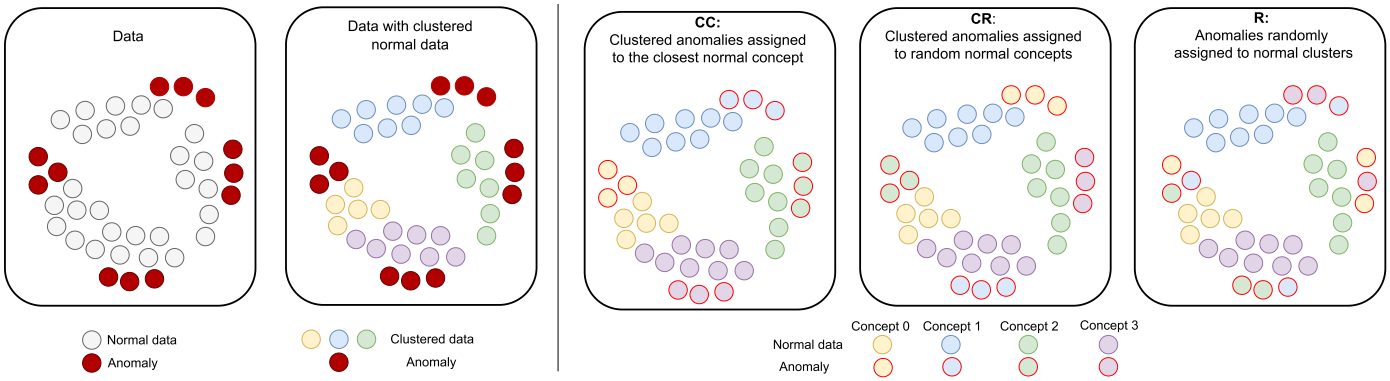


FIGURE 5. **Lifelong scenarios variants based on different choices of concept creation functions γ and λ: *i)* clustered anomaly concepts assigned to the closest normal concept (CC), *ii)* clustered anomaly concepts assigned randomly to normal concepts (CR), and *iii)* anomalies randomly assigned to normal concepts (R).**

non-lifelong anomaly detection models?; ii) Does adopting lifelong learning provide a valuable improvement in the learning capabilities of anomaly detection models in lifelong scenarios? We empirically address these two questions in the following two subsections and provide insights on anomaly detection performance, forgetting, and task similarity. To answer the above question, we design three lifelong scenario variants based on different choices:

- **CC**: clustered anomaly concepts assigned to the closest normal concept, where γ and ϕ leverage a clustering function, and λ maps a given normal concept to the closest anomaly concept.
- **CR**: clustered anomaly concepts assigned randomly to normal concepts, where γ and ϕ leverage a clustering function, and λ assigns a random anomaly concept to a given normal concept.
- **R**: anomalies randomly assigned to normal concepts, where γ leverage a clustering function, ϕ creates anomaly concepts using random sampling, and λ assigns a random anomaly concept to a given normal concept.

For all scenario variants, we use k-Means as the clustering function. These scenarios are conceptually represented as two-dimensional plots in Figure 5. In our experiments, we extract between 5 and 20 concepts depending on the complexity and the size of each dataset.

In our experiments, we employ a diverse set of datasets encompassing cybersecurity and smart grids.

- **NSL-KDD** [70]: Widely adopted dataset containing records of network traffic gathered during the DARPA Intrusion Detection Systems evaluation program;
- **UNSW-NB15** [71]: The dataset containing records of network traffic describing hybrid real modern normal and contemporary synthesized attack activities;
- **Energy** [72]: Sensor-based anomaly detection in photovoltaic/solar power plants. The data was collected from power plants located in Italy (17 plants, 2.5 years);
- **Wind** [4]: Wind power production dataset, containing anomalous patterns occurred in eolic/wind parks (Wind). The data was modeled using the Weather Research & Forecasting (WRF) model (5 plants, 2 years).

All datasets present fairly different feature representations and represent data collected by heterogeneous systems. They also present different types of anomalies and complexity levels.

We use five popular anomaly detection models described in Section II: One-Class Support Vector Machines (OC-SVM), Local Outlier Factor (LOF), Isolation Forests (IF), Variational Auto-Encoders (VAE), and Copula-based Outlier Detection (COPOD).

We leverage the following learning strategies:

- **Naive lifelong**: models are updated as new data becomes available, without any smart lifelong learning strategy to tune adaptation and knowledge retention. By updating the model only based on the new data, a reasonable expectation is that the model will gradually or catastrophically forget knowledge of previously presented data. It can be considered as a lower-bound non-lifelong baseline learning strategy.
- **Multiple Single-Task Experts (MSTE)**: a way to simulate upper-bound model performance in a non-lifelong scenario. In this strategy, a pool or ensemble of models, each of which is an expert for a single concept, is adopted. Whenever a new concept is presented, a new model is trained on the new data and added to the pool. We note that this is an unrealistic setting since, in real-world scenarios, it requires extremely high computational resources to deal with a potentially infinite number of models, and the availability of concept identifiers, which are not available for concept-incremental and concept-agnostic scenarios.
- **Replay**: a replay-based method that preserves selected data samples from previous concepts in a memory buffer, which is limited in size by a parameter known as a budget. When the model faces a new task (concept), the replay buffer is updated to include the data from the new concept. As a result, the replay buffer contains knowledge of all concepts presented so far. The replay

buffer is then used while updating the model to mitigate forgetting [46]. The expectation is that, by providing a summarized representation of all concepts, the model should, in principle, be able to preserve a satisfactory performance on all concepts, without a significant degree of forgetting for any of the concepts. In our experiments, we use a simple balanced replay buffer with a very constrained budget of 3,000 data samples.

To this end, we adopt the learning evaluation workflow in Algorithm 2, which provides the model with the challenge of adapting to new data while retaining knowledge of previously observed data. Technically, we create a matrix where the performance of the model is measured on all concepts after learning each single concept. Metric computation takes place according to Equations 1-3.

We address **RQ1** by devising experiments to assess whether non-lifelong anomaly detection methods are impacted by the challenges brought by lifelong scenarios. To this aim, leverage the two mentioned strategies (Naive lifelong and MSTE) to showcase if there is a gap between models' performance in non-lifelong scenarios vs. lifelong scenarios, which would suggest the need for adopting lifelong learning strategies. Particularly, we are interested in observing whether MSTE achieves higher performance values in terms of ROC-AUC, BWT, and FWT compared to Naive. This expectation is motivated by the observation that Naive only adapts to new data without any knowledge retention mechanism, while MSTE does not experience forgetting due to the creation of a single expert model for each task.

We address **RQ2** by devising experiments to assess whether the adoption of lifelong learning strategies has the potential to increase the performance of non-lifelong anomaly detection models in lifelong scenarios. To this aim, we analyze the impact of the adoption of a lifelong Replay strategy on the performance of all base models to show that the adoption of lifelong learning strategies may be beneficial to improve model performance in a lifelong anomaly detection scenario. Experimental results[4] are shown in Tables 1, 2.

VI. RESULTS DISCUSSION

In this section, we discuss results alongside two main perspectives: the impact of lifelong scenarios on non-lifelong anomaly detection problems, and the impact of the adoption of lifelong learning strategies providing knowledge retention capabilities.

A. IMPACT OF LIFELONG SCENARIOS ON NON-LIFELONG ANOMALY DETECTION

In this subsection, we focus on providing insights on how non-lifelong anomaly detection methods are impacted by the challenges brought by lifelong scenarios (**RQ1**).

[4]We note that our results are obtained by averaging multiple executions with different hyperparameter values (see Appendix) to showcase the reliability of the results, similarly to a cross-validation evaluation [73], [74].

We present the experimental results in Table 1. The general trend that can be observed across all methods and datasets is that there is a gap between the anomaly detection performance in terms of ROC-AUC achieved by widely adopted anomaly detection methods and the hypothetical upper-bound defined by multiple single-task experts (MSTE). Notably, with the Energy dataset, base models with a Naive learning strategy are significantly outperformed by the MSTE approach (for example, for Isolation Forest, CC: 0.64 vs. 0.88 — CR: 0.65 vs. 0.97 — R: 0.59 vs. 0.97). This is not an isolated case but applies to the other datasets as well.

It is noteworthy that MSTE achieves very high performance in most cases, highlighting that the scenario generation procedure decomposes each dataset's complexities into sub-complexities (concepts), which are much more manageable to learn in isolation but much more challenging when provided as a lifelong scenario. This phenomenon has been observed in [65] in the context of non-lifelong one-class classification. Overall, it looks clear that non-lifelong anomaly detection methods are penalized in lifelong scenarios, leading to sub-optimal performance scores, as evident by the low ROC-AUC values. We present a graphical illustration of this phenomenon in Figure 8, which shows a clear performance gap between non-lifelong and lifelong strategies.

Another lifelong metric worth analyzing is the backward transfer (BWT) as it allows us to analyze how learning new tasks affects model performance on previous tasks. We recall that negative values of BWT indicate that learning new tasks introduces forgetting in previously learned tasks, whereas positive values are indicative of effective knowledge transfer capabilities across tasks (see Section IV-C). We observe that all base models with Naive strategy present negative values of BWT, e.g., with Wind (R), models showcase values from -0.11 to -0.52, which shows that they are affected by a degree of forgetting (from mild to catastrophic). This phenomenon is more clearly visible in Figure 6 (VAE) and 7 (LOF), where the performance on C_0 drops from 1 to 0.69 (VAE) and from 0.99 to 0.41 (LOF) after the last concept C_4 is learned. Negative backward transfer is also observed for C_2, where model performance gradually drops from 0.99 to 0.5 (VAE) and from 0.99 to 0.34 (LOF) after learning C_3. Subsequently, learning C_4 leads to increased forgetting of C_2, and the performance on C_2 drops to 0.078 (VAE) and 0.098 (LOF). Another exciting aspect that can be observed is a positive backward transfer, which indicates that learning a new concept improves the performance of the model on a previous concept. This is emphasized by the performance on C_0 before and after learning C_2. Results show that learning C_2 increases the performance on C_0 from 0.51 to 0.96 (VAE) and from 0.42 to 0.55 (LOF). It means that the model can leverage the knowledge acquired while learning C_2 to present better anomaly detection capabilities on concept C_0 by leveraging similarity between concepts (task-similarity). By analyzing BWT, we uncover concept similarity relationships and measure models' ability to retain and reuse

TABLE 1. Experimental results for Naive lifelong strategy with all methods (IF, LOF, COPOD, OC-SVM, and VAE) and datasets (Energy, NSL-KDD, UNSW, Wind) in the three concept-incremental scenarios (CC, CR, R), according to ROC-AUC, BWT, FWT metrics. For ROC-AUC, we also report results obtained with multiple single-task experts (MSTE) in parenthesis (upper bound).

	IF			LOF			COPOD			OC-SVM			VAE		
Dataset	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT
Energy (CC)	0.64 (0.88)	-0.29	0.61	0.71 (0.96)	-0.31	0.81	0.85 (0.91)	-0.07	0.90	0.76 (0.94)	-0.22	0.84	0.75 (0.92)	-0.21	0.81
Energy (CR)	0.65 (0.97)	-0.40	0.53	0.70 (0.99)	-0.35	0.73	0.91 (0.98)	-0.09	0.86	0.75 (0.99)	-0.29	0.71	0.74 (0.99)	-0.31	0.68
Energy (R)	0.59 (0.97)	-0.46	0.56	0.66 (1.00)	-0.41	0.70	0.89 (0.98)	-0.11	0.87	0.69 (0.99)	-0.37	0.70	0.68 (0.99)	-0.38	0.69
NSL-KDD (CC)	0.68 (0.97)	-0.32	0.77	0.62 (0.93)	-0.33	0.56	0.54 (0.66)	-0.14	0.43	0.67 (0.99)	-0.36	0.85	0.67 (0.98)	-0.35	0.85
NSL-KDD (CR)	0.70 (0.96)	-0.28	0.72	0.60 (0.94)	-0.37	0.54	0.52 (0.62)	-0.13	0.44	0.75 (0.96)	-0.23	0.73	0.73 (0.96)	-0.25	0.73
NSL-KDD (R)	0.85 (0.99)	-0.16	0.82	0.63 (0.95)	-0.35	0.52	0.74 (0.81)	-0.08	0.67	0.88 (1.00)	-0.13	0.85	0.87 (1.00)	-0.14	0.86
UNSW (CC)	0.49 (0.73)	-0.29	0.45	0.57 (0.88)	-0.38	0.45	0.32 (0.43)	-0.11	0.29	0.59 (0.78)	-0.22	0.53	0.55 (0.81)	-0.32	0.50
UNSW (CR)	0.60 (0.94)	-0.41	0.50	0.67 (0.98)	-0.38	0.49	0.48 (0.67)	-0.24	0.24	0.73 (0.97)	-0.31	0.56	0.72 (0.98)	-0.32	0.51
UNSW (R)	0.53 (0.90)	-0.45	0.45	0.60 (0.96)	-0.42	0.46	0.60 (0.76)	-0.18	0.44	0.55 (0.91)	-0.44	0.46	0.56 (0.94)	-0.46	0.44
Wind (CC)	0.83 (0.90)	-0.10	0.74	0.65 (0.98)	-0.49	0.58	0.89 (0.93)	-0.06	0.89	0.77 (0.96)	-0.28	0.77	0.76 (0.96)	-0.30	0.76
Wind (CR)	0.74 (0.95)	-0.32	0.71	0.62 (0.99)	-0.57	0.53	0.89 (0.96)	-0.12	0.90	0.72 (1.00)	-0.42	0.70	0.68 (0.98)	-0.46	0.73
Wind (R)	0.74 (0.95)	-0.31	0.69	0.65 (0.99)	-0.52	0.50	0.90 (0.97)	-0.11	0.91	0.74 (0.99)	-0.38	0.66	0.71 (0.99)	-0.42	0.64

TABLE 2. Experimental results for Replay strategy with all methods (IF, LOF, COPOD, OC-SVM, and VAE) and datasets (Energy, NSL-KDD, UNSW, Wind) in the three concept-incremental scenarios (CC, CR, R), according to ROC-AUC, BWT, FWT metrics.

	IF			LOF			COPOD			OC-SVM			VAE		
	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT	ROC-AUC	BWT	FWT
Energy (CC)	0.82	-0.06	0.64	0.95	-0.01	0.78	0.89	-0.01	0.89	0.83	-0.04	0.85	0.77	-0.16	0.83
Energy (CR)	0.78	-0.13	0.51	0.96	-0.01	0.64	0.89	-0.04	0.83	0.85	-0.06	0.77	0.81	-0.15	0.66
Energy (R)	0.75	-0.18	0.58	0.97	-0.02	0.61	0.87	-0.06	0.83	0.84	-0.07	0.76	0.75	-0.25	0.68
NSL-KDD (CC)	0.85	-0.11	0.92	0.89	-0.03	0.33	0.79	0.00	0.78	0.71	-0.21	0.89	0.82	-0.09	0.93
NSL-KDD (CR)	0.81	-0.07	0.82	0.88	-0.02	0.39	0.72	-0.02	0.62	0.79	-0.11	0.74	0.84	-0.04	0.76
NSL-KDD (R)	0.95	-0.03	0.90	0.90	-0.01	0.34	0.92	0.04	0.84	0.93	-0.03	0.86	0.93	-0.01	0.85
UNSW (CC)	0.51	-0.03	0.35	0.83	-0.00	0.32	0.43	0.00	0.29	0.56	-0.02	0.54	0.65	-0.12	0.38
UNSW (CR)	0.68	-0.02	0.45	0.89	-0.01	0.39	0.67	0.00	0.30	0.82	0.01	0.57	0.73	-0.05	0.33
UNSW (R)	0.48	-0.08	0.35	0.82	-0.05	0.40	0.52	-0.05	0.39	0.50	-0.08	0.46	0.66	-0.08	0.39
Wind (CC)	0.91	-0.01	0.74	0.97	-0.01	0.60	0.92	-0.01	0.90	0.91	-0.04	0.83	0.86	-0.13	0.72
Wind (CR)	0.88	-0.08	0.70	0.97	-0.03	0.57	0.89	-0.06	0.89	0.88	-0.08	0.75	0.84	-0.18	0.74
Wind (R)	0.90	-0.06	0.67	0.98	-0.02	0.52	0.91	-0.06	0.89	0.91	-0.06	0.72	0.86	-0.17	0.72

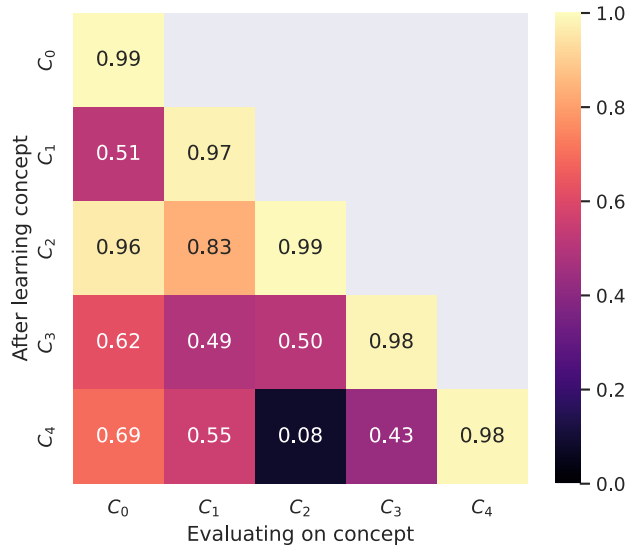


FIGURE 6. Concept-level ROC-AUC performance for the lifelong learning scenario. Each row $i = 0, 1, \ldots$ represents the performance on all concepts observed so far after learning the concept C_i (WIND (R) dataset; Naive lifelong strategy; VAE base model).

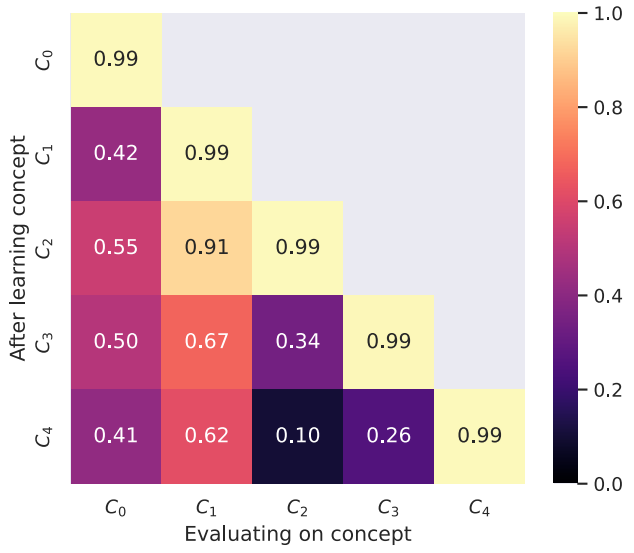


FIGURE 7. Concept-level ROC-AUC performance for the lifelong learning scenario. Each row $i = 0, 1, \ldots$ represents the performance on all concepts observed so far after learning the concept C_i (WIND (R) dataset; Naive lifelong strategy; LOF base model).

knowledge for improving its overall performance across all concepts.

Finally, moving our focus to Forward Transfer (*FWT*), we can observe values from 0.24 – UNSW (CR) with

COPOD to 0.91 – Wind (R) with COPOD. This result suggests a moderate-to-high concept similarity that could be leveraged by models. Interestingly, these values are higher than those commonly seen in image classification since the one-class learning setting is inherently different from multi-class classification. Specifically, since we compute FWT using ROC-AUC as a base metric, the random reference value is 0.5, whereas, in image classification, it is the ratio between 1 and the number of classes in the single task. This difference exacerbates the complexity of a comparison of FWT in these two settings. However, we argue that interpreting *FWT* in this context requires additional in-depth research focused on leveraging concepts similarity for one-class anomaly detection.

In summary, we observed a gap in ROC-AUC performance (Naive lifelong *vs*. MSTE) alongside with the presence of forgetting (emphasized by negative values of BWT) and degrees of concept similarity (observed via BWT and *FWT*). These phenomena show that lifelong learning scenarios generated with our approach allow us to adapt non-lifelong datasets to lifelong anomaly detection scenarios, introducing lifelong challenges which yield more demanding conditions for models (**RQ1**).

Additionally, our results show that tackling anomaly detection problems from a lifelong learning perspective can enable a more comprehensive evaluation of models, including measuring the impact of forgetting and task transferability on models. Concept-level granularity in results also allows us to uncover relationships between concepts in the learning scenario, as well as highlight specific concepts for which models are challenged more than others, enabling a more comprehensive evaluation and in-depth model analysis.

B. IMPACT OF THE ADOPTION OF A REPLAY-BASED LIFELONG LEARNING STRATEGY

In this subsection, we analyze the impact of the adoption of a lifelong knowledge retention strategy (Replay) on the performance of anomaly detection models. Our effort is aimed at verifying whether the adoption of lifelong learning strategies may be beneficial to improve model performance in a lifelong anomaly detection scenario (**RQ2**).

In order to assess the impact of introducing Replay strategy, we compare anomaly detection performance (ROC-AUC) and backward transfer (BWT) for Replay (Table 2) and the Naive lifelong strategy (Table 1). We can observe that the Replay strategy brings various levels of improvement, as shown by higher values of ROC-AUC and BWT. In some cases, the improvement margin is particularly high. For example, in Energy (CC), the Replay strategy can improve ROC-AUC by 0.18 (IF). There are also many cases in which the improvement margin is moderate. For example, in Wind (CC), Replay yields a ROC-AUC improvement of 0.08 (IF). Finally, the improvement margin is quite limited in cases such as UNSW (CC), in which Replay improves the performance of Naive by just 0.02 (IF). Similar patterns can be observed across all base models (IF, LOF, COPOD, OC-SVM, VAE). A summarized visual perspective of these results is shown in Figure 8, which emphasizes the differences between Naive, Replay, and MSTE.

Results in Table 2 show that in the majority of cases (54 out of 60), the simple Replay strategy achieves better results in terms of ROC-AUC than the Naive lifelong approach. Most of the exceptions regard the UNSW (R) scenario. We attribute this result to the concept complexity in UNSW and the simplicity of the Replay strategy. In this scenario, it is evident that more complex lifelong strategies are required to outperform the Naive baseline.

The improvement in performance is also supported by the observation of a decrease in forgetting, as shown by improvements in Backward Transfer (BWT) results when comparing Naive lifelong and Replay. Improvements are considerable, for example, in UNSW (R), where BWT for VAE goes from -0.46 to -0.08 for VAE, as well as in Wind (CR), where BWT goes from -0.46 to -0.18. Improvements with the Replay strategy compared to Naive lifelong can also be observed in Figure 9 in comparison to Figure 6, where we observe a significant increase in performance and a decrease in forgetting in all cases except two (performance on C_0 after learning C_2, and performance on C_1 after learning C_1). These two cases highlight that, although the Replay strategy is expected to improve the average model performance due to consideration of all concepts, it also provides additional challenges for the model, which is tasked to learn multiple concepts. As a result, minor decreases in performance for specific concepts should be expected.

We note that there is still a gap between Replay results and simulated upper-bound MSTE results. We can observe that MSTE presents better results than Replay in most cases (56 out of 60). There are 4 cases in which the Replay strategy is better than MSTE. The reason behind this phenomenon can be attributed to the fact that while MSTE can build specialized models for each concept, it cannot leverage knowledge from multiple tasks, thus exploiting task similarity, which, in contrast, is supported in a basic form by Replay.

The improvements in ROC-AUC and BWT observed with Replay when compared with Naive suggest that adopting lifelong learning techniques and learning strategies referenced in Section II, as well as designing new ones, will be beneficial for the advancement of anomaly detection in lifelong learning scenarios (**RQ2**). At the same time, the discussed gap in performance between Replay and MSTE suggests that more robust lifelong strategies may be devised to further improve model performance in complex lifelong anomaly detection scenarios.

C. SUMMARY OF GAINED INSIGHTS

In summary, our experimental analysis discussed above led us to learn the following insights:

- A performance gap exists between non-lifelong and lifelong learning strategies in lifelong anomaly detection scenarios (comparing MSTE vs. Naive) (**RQ1**). This

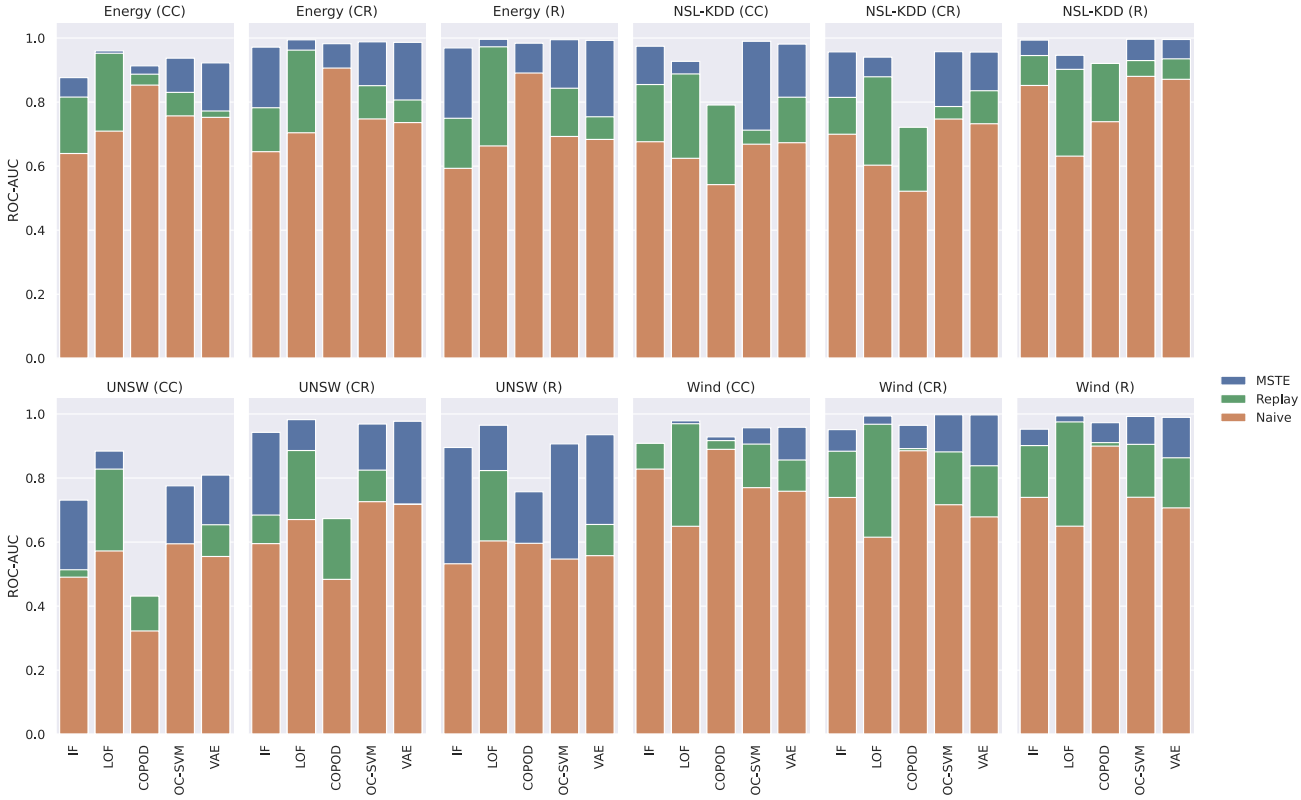


FIGURE 8. Summary of experimental results for all datasets (Energy, UNSW, Wind) in three scenario types (CC, CR, R) comparing non-lifelong (Naive), lifelong (Replay), and upper bound (MSTE) learning strategies. This figure illustrates the performance gap between non-lifelong and lifelong strategies in lifelong anomaly detection scenarios.

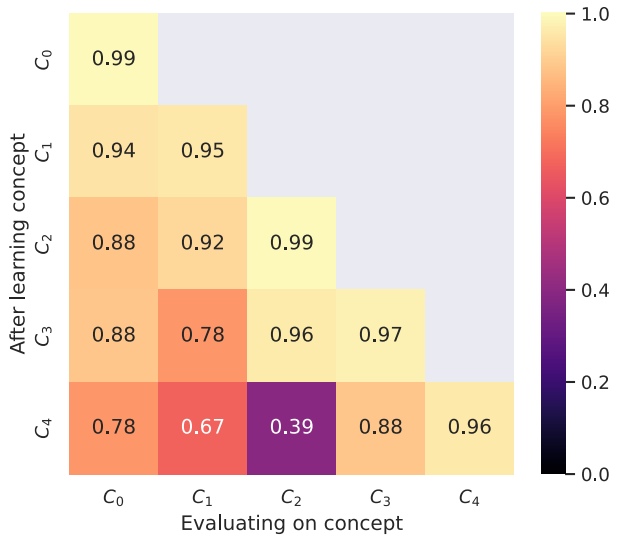


FIGURE 9. Concept-level ROC-AUC performance for the lifelong learning scenario. Each row $i = 0, 1, \ldots$ represents the performance on all concepts observed so far after learning the concept C_i (WIND (R) dataset; Replay strategy; LOF base model).

gap was systematically observed across all scenarios and datasets, and needs to be addressed to increase the reliability of anomaly detection models.

- In such scenarios, the adoption of (even simple) lifelong learning strategies such as Replay enables an improvement in anomaly detection performance when compared to non-lifelong models (comparing Replay vs Naive) (**RQ2**).
- Forgetting was broadly observed for many anomaly detection base models when exposed to the complexity of lifelong scenarios, as emphasized by negative values of BWT. This phenomenon suggests that there is an opportunity to build more robust models that simultaneously deal with adaptation and knowledge retention (**RQ1, RQ2**).
- Our in-depth lifelong evaluation of model performance revealed cases of positive concept similarity (task similarity). This aspect translates in high values of the forward transfer metric (FWT), leading to above-random anomaly detection performance when models are faced with data from not yet learned concepts. Concept similarity is also leveraged by models to improve performance on previously learned concepts after learning a similar concept, resulting in improved values of backward transfer (BWT).

VII. CONCLUSION

In this paper, we addressed lifelong learning in the context of anomaly detection. In general, our contribution stands in the definition of a common ground for research on this topic and showcasing challenges, perspectives and insights brought by lifelong learning for anomaly detection. Specifically, we devised a learning setting characterization that could be

TABLE 3. Hyperparameters for all the anomaly detection models considered in our experiments. All models are trained and evaluated five times using all hyperparameter values in the sets shown in the table, and the final results are averaged.

VAE	hidden layers size $\in \{(16, 4, 16); (12, 8, 12); (19, 9, 19); (32, 16, 32); (32, 8, 32)\}$
LOF	n-neighbors $\in \{5; 10; 15; 20; 25\}$
IF	n-estimators $\in \{25; 35; 50; 75; 100\}$
OC-SVM	$\nu \in \{0.01; 0.02; 0.03; 0.05; 0.1\}$
	$\gamma \in \{0.01; 0.02; 0.03; 0.05; 0.1\}$
COPOD	parameterless

useful to adopt lifelong learning in the context of anomaly detection. Moreover, we designed a procedure for scenario generation that can be used to create lifelong learning scenarios adopting any standard anomaly detection dataset. Insights from our experiments revealed that anomaly detection in lifelong learning scenarios is a challenging problem, and that there is a performance gap between non-lifelong and lifelong learning strategies, indicating that lifelong scenarios are challenging for commonly adopted non-lifelong anomaly detection methods. Moreover, we observed that lifelong learning strategies such as Replay have the potential to tackle these challenges. Overall, we advocate that lifelong learning is essential in anomaly detection to further bring real-life complexity to the experimental setting, providing advantages compared to static and online scenarios currently adopted in the literature. We identified a number of domains, such as cybersecurity, human activity, and industrial processes, where such capabilities can be fruitful due to their dynamic characteristics. We showed that lifelong learning metrics and concept-level performance observed in the learning scenario enable a more detailed model evaluation that uncovers the impact of forgetting and task transferability.

As we provided an overview of lifelong learning challenges from an anomaly detection perspective, we believe that our work will enable other researchers to take action by working on open problems that are relevant in lifelong anomaly detection. To start with, anomaly detection researchers may adopt the scenarios (e.g., concept-aware, concept-incremental) in their benchmark datasets to expose anomaly detection models to new complexities. Moreover, they can adopt lifelong metrics (Lifelong ROC-AUC, BWT, FWT), as well as the evaluation protocol proposed in this paper. In the future, efforts should be directed at designing or improving lifelong scenarios and metrics, so that they reflect the real-world anomaly detection complexities even better. Avenues for future research also include the design of different types of lifelong learning strategies beyond replay-based, such as regularization and architectural, which are popular in image classification, and could be tailored to lifelong anomaly detection.

APPENDIX. HYPERPARAMETERS OF THE DIFFERENT BASE MODELS

For reproducibility, Table 3 shows the five hyperparameter configurations for all models considered in our experiments.

REFERENCES

[1] C. C. Aggarwal, "An introduction to outlier analysis," in *Outlier Analysis*. Cham, Switzerland: Springer, 2013, pp. 1–40.
[2] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: A comprehensive evaluation," *Proc. VLDB Endowment*, vol. 15, no. 9, pp. 1779–1797, May 2022.
[3] K. Faber, L. Faber, and B. Sniezynski, "Autoencoder-based IDS for cloud and mobile devices," in *Proc. IEEE/ACM 21st Int. Symp. Cluster, Cloud Internet Comput. (CCGrid)*, May 2021, pp. 728–736.
[4] R. Corizzo, M. Ceci, G. Pio, P. Mignone, and N. Japkowicz, "Spatially-aware autoencoders for detecting contextual anomalies in geo-distributed data," in *Proc. Int. Conf. Discovery Sci.* Cham, Switzerland: Springer, 2021, pp. 461–471.
[5] M. Fahim and A. Sillitti, "Anomaly detection, analysis and prediction techniques in IoT environment: A systematic literature review," *IEEE Access*, vol. 7, pp. 81664–81681, 2019.
[6] A. L. Alfeo, M. G. C. A. Cimino, G. Manco, E. Ritacco, and G. Vaglini, "Using an autoencoder in the design of an anomaly detector for smart manufacturing," *Pattern Recognit. Lett.*, vol. 136, pp. 272–278, Aug. 2020.
[7] G. Pang, C. Shen, L. Cao, and A. Van Den Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, vol. 54, no. 2, pp. 1–38, 2021.
[8] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," 2019, *arXiv:1901.03407*.
[9] G. Fenza, M. Gallo, and V. Loia, "Drift-aware methodology for anomaly detection in smart grid," *IEEE Access*, vol. 7, pp. 9645–9657, 2019.
[10] R. Hou, Y. Peng, L. J. Grimm, Y. Ren, M. A. Mazurowski, J. R. Marks, L. M. King, C. C. Maley, E. S. Hwang, and J. Y. Lo, "Anomaly detection of calcifications in mammography based on 11,000 negative cases," *IEEE Trans. Biomed. Eng.*, vol. 69, no. 5, pp. 1639–1650, May 2022.
[11] R. Corizzo, M. Ceci, E. Zdravevski, and N. Japkowicz, "Scalable auto-encoders for gravitational waves detection from time series data," *Expert Syst. Appl.*, vol. 151, Aug. 2020, Art. no. 113378.
[12] R. Laxhammar and G. Falkman, "Online learning and sequential anomaly detection in trajectories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1158–1173, Jun. 2014.
[13] Y. Feng, Y. Yuan, and X. Lu, "Learning deep event models for crowd anomaly detection," *Neurocomputing*, vol. 219, pp. 548–556, Jan. 2017.
[14] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Netw.*, vol. 113, pp. 54–71, May 2019.
[15] A. Cossu, A. Carta, V. Lomonaco, and D. Bacciu, "Continual learning for recurrent neural networks: An empirical evaluation," *Neural Netw.*, vol. 143, pp. 607–627, Nov. 2021.
[16] J. Gama, *Knowledge Discovery From Data Streams*. Boca Raton, FL, USA: CRC Press, 2010.
[17] G. I. Parisi, X. Ji, and S. Wermter, "On the role of neurogenesis in overcoming catastrophic forgetting," 2018, *arXiv:1811.02113*.
[18] K. James, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.
[19] B. Wickramasinghe, G. Saha, and K. Roy, "Continual learning: A review of techniques, challenges and future directions," *IEEE Trans. Artif. Intell.*, vol. 1, no. 1, pp. 1–21, Dec. 2023.
[20] H. Liu, Y. Yang, and X. Wang, "Overcoming catastrophic forgetting in graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, 2021, pp. 8653–8661.
[21] A. Chaudhry, A. Gordo, P. Dokania, P. Torr, and D. Lopez-Paz, "Using hindsight to anchor past knowledge in continual learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 8, pp. 6993–7001.
[22] M. Masana, X. Liu, B. Twardowski, M. Menta, A. D. Bagdanov, and J. van de Weijer, "Class-incremental learning: Survey and performance evaluation on image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5513–5533, May 2023.
[23] D. Abel, Y. Jinnai, S. Y. Guo, G. Konidaris, and M. Littman, "Policy and value transfer in lifelong reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 20–29.
[24] J. Parmar, S. Chouhan, V. Raychoudhury, and S. Rathore, "Open-world machine learning: Applications, challenges, and opportunities," *ACM Comput. Surv.*, vol. 55, no. 10, pp. 1–37, Oct. 2023.
[25] M. M. Baker, "A domain-agnostic approach for characterization of lifelong learning systems," *Neural Netw.*, vol. 160, pp. 274–296, Mar. 2023.

[26] M. Mundt, Y. Hong, I. Pliushch, and V. Ramesh, "A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning," *Neural Netw.*, vol. 160, pp. 306–336, Mar. 2023.
[27] X. Nie, Z. Deng, M. He, M. Fan, and Z. Tang, "Online active continual learning for robotic lifelong object recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2024.
[28] Z. Mai, R. Li, J. Jeong, D. Quispe, H. Kim, and S. Sanner, "Online continual learning in image classification: An empirical survey," *Neurocomputing*, vol. 469, pp. 28–51, Jan. 2022.
[29] K. Faber, B. Sniezynski, and R. Corizzo, "Distributed continual intrusion detection: A collaborative replay framework," in *Proc. IEEE Int. Conf. Big Data (BigData)*, Los Alamitos, CA, USA, Dec. 2023, pp. 3255–3263.
[30] D. Javaheri, S. Gorgin, J.-A. Lee, and M. Masdari, "Fuzzy logic-based DDoS attacks and network traffic anomaly detection methods: Classification, overview, and future perspectives," *Inf. Sci.*, vol. 626, pp. 315–338, May 2023.
[31] G. M. van de Ven and A. S. Tolias, "Three scenarios for continual learning," 2019, *arXiv:1904.07734*.
[32] E. Belouadah, A. Popescu, and I. Kanellos, "A comprehensive study of class incremental learning algorithms for visual tasks," *Neural Netw.*, vol. 135, pp. 38–54, Mar. 2021.
[33] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3366–3385, Jul. 2022.
[34] N. Gunasekara, H. Gomes, A. Bifet, and B. Pfahringer, "Adaptive neural networks for online domain incremental continual learning," in *Discovery Science*, P. Pascal and D. Lenco, Eds. Cham, Switzerland: Springer, 2022, pp. 89–103.
[35] M. De Lange and T. Tuytelaars, "Continual prototype evolution: Learning online from non-stationary data streams," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 8230–8239.
[36] A. Soutif–Cormerais, A. Carta, A. Cossu, J. Hurtado, H. Hemati, V. Lomonaco, and J. Van de Weijer, "A comprehensive empirical evaluation on online continual learning," 2023, *arXiv:2308.10328*.
[37] V. Lomonaco, D. Maltoni, and L. Pellegrini, "Rehearsal-free continual learning over small non-I.I.d. batches," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 989–998.
[38] A. Cossu, G. Graffieti, L. Pellegrini, D. Maltoni, D. Bacciu, A. Carta, and V. Lomonaco, "Is class-incremental enough for continual learning?" *Frontiers Artif. Intell.*, vol. 5, pp. 1–6, Mar. 2022.
[39] Y.-H. Wang, C.-Y. Lin, T. Thaipisutikul, and T. K. Shih, "Single-head lifelong learning based on distilling knowledge," *IEEE Access*, vol. 10, pp. 35469–35478, 2022.
[40] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2014, pp. 512–519.
[41] V. Lomonaco and D. Maltoni, "CORe50: A new dataset and benchmark for continuous object recognition," in *Proc. Conf. Robot Learn.*, 2017, pp. 17–26.
[42] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, Dec. 2018.
[43] T. Diethe, T. Borchert, E. Thereska, B. Balle, and N. Lawrence, "Continual learning in practice," in *Proc. NeurIPS Continual Learn. Workshop*, 2018, pp. 1–9.
[44] A. Mallya and S. Lazebnik, "PackNet: Adding multiple tasks to a single network by iterative pruning," 2017, *arXiv:1711.05769*.
[45] H. Kang, R. J. L. Mina, S. Rizky, H. Madjid, J. Yoon, M. Hasegawa-Johnson, S. Ju-Hwang, and C. D. Yoo, "Forget-free continual learning with winning subnetworks," in *Proc. ICML*, 2022, pp. 10734–10750.
[46] P. Buzzega, M. Boschini, A. Porrello, and S. Calderara, "Rethinking experience replay: A bag of tricks for continual learning," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 2180–2187.
[47] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Proc. NeurIPS*, vol. 30, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2017, pp. 1–10.
[48] P. K. Mvula, P. Branco, G.-V. Jourdan, and H. L. Viktor, "HEART: Heterogeneous log anomaly detection using robust transformers," in *Proc. Int. Conf. Discovery Sci.* Cham, Switzerland: Springer, 2023, pp. 673–687.
[49] J. Sendorek, T. Szydlo, M. Windak, and R. Brzoza-Woch, "Dataset for anomalies detection in 3D printing," in *Proc. Int. Conf. Comput. Sci.*, Jun. 2021, pp. 647–653.
[50] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PLoS ONE*, vol. 11, no. 4, Apr. 2016, Art. no. e0152173.
[51] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[52] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 582–588.
[53] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 93–104.
[54] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Min.*, Dec. 2008, pp. 413–422.
[55] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu, "COPOD: Copula-based outlier detection," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2020, pp. 1118–1123.
[56] A. Frikha, D. Krompaß, and V. Tresp, "ARCADe: A rapid continual anomaly detector," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 10449–10456.
[57] K. Doshi and Y. Yilmaz, "Continual learning for anomaly detection in surveillance videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 1025–1034.
[58] R. Corizzo, M. Baron, and N. Japkowicz, "CPDGA: Change point driven growing auto-encoder for lifelong anomaly detection," *Knowl.-Based Syst.*, vol. 247, Jul. 2022, Art. no. 108756.
[59] K. Faber, R. Corizzo, B. Sniezynski, and N. Japkowicz, "VLAD: Task-agnostic VAE-based lifelong anomaly detection," *Neural Netw.*, vol. 165, pp. 248–273, Aug. 2023.
[60] K. Faber, R. Corizzo, B. Sniezynski, and N. Japkowicz, "Active lifelong anomaly detection with experience replay," in *Proc. IEEE 9th Int. Conf. Data Sci. Adv. Anal. (DSAA)*, Oct. 2022, pp. 1–10.
[61] M. Du, Z. Chen, C. Liu, R. Oak, and D. Song, "Lifelong anomaly detection through unlearning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 1283–1297.
[62] B. Krawczyk and M. Woźniak, "One-class classifiers with incremental learning and forgetting for data streams with concept drift," *Soft Comput.*, vol. 19, no. 12, pp. 3387–3400, Dec. 2015.
[63] D. Kudithipudi et al., "Biological underpinnings for lifelong learning machines," *Nature Mach. Intell.*, vol. 4, no. 3, pp. 196–210, 2022.
[64] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with A-GEM," 2018, *arXiv:1812.00420*.
[65] S. Sharma, A. Somayaji, and N. Japkowicz, "Learning over subconcepts: Strategies for 1-class classification," *Comput. Intell.*, vol. 34, no. 2, pp. 440–467, May 2018.
[66] Y. Ghunaim, A. Bibi, K. Alhamoud, M. Alfarra, H. A. A. K. Hammoud, A. Prabhu, P. H. S. Torr, and B. Ghanem, "Real-time evaluation in online continual learning: A new hope," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 11888–11897.
[67] K. Fujiwara, M. Shigeno, and U. Sumita, "A new approach for developing segmentation algorithms for strongly imbalanced data," *IEEE Access*, vol. 7, pp. 82970–82977, 2019.
[68] K. Ghosh, C. Bellinger, R. Corizzo, P. Branco, B. Krawczyk, and N. Japkowicz, "The class imbalance problem in deep learning," *Mach. Learn.*, vol. 111, pp. 1–57, Dec. 2022.
[69] N. Díaz-Rodríguez, V. Lomonaco, D. Filliat, and D. Maltoni, "Don't forget, there is more than forgetting: New metrics for continual learning," 2018, *arXiv:1810.13166*.
[70] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.
[71] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.
[72] R. Corizzo, M. Ceci, and N. Japkowicz, "Anomaly detection and repair for accurate predictions in geo-distributed big data," *Big Data Res.*, vol. 16, pp. 18–35, Jul. 2019.
[73] J. Khan, E. Lee, and K. Kim, "A higher prediction accuracy-based alpha–beta filter algorithm using the feedforward artificial neural network," *CAAI Trans. Intell. Technol.*, vol. 8, no. 4, pp. 1124–1139, Dec. 2023.
[74] J. Khan, M. Fayaz, A. Hussain, S. Khalid, W. K. Mashwani, and J. Gwak, "An improved alpha beta filter using a deep extreme learning machine," *IEEE Access*, vol. 9, pp. 61548–61564, 2021.

KAMIL FABER received the B.S. and M.S. degrees from the AGH University of Krakow, where he is currently pursuing the Ph.D. degree in computer science. He was a Research Intern and later a Research Assistant with the DARPA-funded project with American University, from 2021 to 2022. His research interests include lifelong learning, anomaly detection, and machine learning applications in cybersecurity.

ROBERTO CORIZZO (Member, IEEE) received the Ph.D. degree. He is an Assistant Professor with the Department of Computer Science, American University. He has coauthored over 40 articles, including 13 publications in journals, such as IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *Neural Networks*, and *Machine Learning*. He was involved in the scientific committee of international conferences and served as a reviewer for several international journals.

BARTLOMIEJ SNIEZYNSKI received the Ph.D. degree in computer science from the AGH University of Science and Technology, Kraków, Poland, in 2004. In 2004, he was a Postdoctoral Fellow with the Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, USA, under the supervision of Prof. R. S. Michalski. Currently, he is an Associate Professor with the Institute of Computer Science, AGH University of Science and Technology. His research interests include machine learning, multi-agent systems, and knowledge engineering. He is a member of the Polish Information Processing Society (PTI) and the Polish Artificial Intelligence Society (PSSI).

NATHALIE JAPKOWICZ (Member, IEEE) is a Professor of computer science with American University. Previously, she was with the School of Electrical Engineering and Computer Science, University of Ottawa, where she leads the Laboratory for Research on Machine Learning for Defense and Security. Over the years, she has supervised over 30 graduate students; received funding from Canadian federal and provincial institutions (NSERC, DRDC, Health Canada, OCE, and MITACS CITO); and worked with private companies (Girih, Larus Technologies, Weather Telematics, TechInsights, and Ciena). She has published over 100 articles, papers, and books, including *Evaluating Learning Algorithms: A Classification Perspective*, with Mohak Shah (Cambridge University Press, 2011); and *Big Data Analysis: New Algorithms for a New Society*, with Jerzy Stefanowski (Springer, 2016).

Chapter 11

From MNIST to ImageNet and Back: Benchmarking Continual Curriculum Learning

In this paper, we proposed two novel benchmarks that involve multiple heterogeneous tasks with varying qualities and complexities. The heterogeneity across datasets allowed us to inject realistic complexities into the learning scenario, resulting in challenging conditions for lifelong strategies. We put particular emphasis on the rigorous and reproducible evaluation of model generalization capabilities and forgetting.

Our extensive experimental evaluation showed that popular lifelong image classification strategies, which are known to be robust on commonly adopted scenarios, fail to achieve satisfactory performance with our benchmarks. Moreover, the results revealed that the replay strategy consistently maintains the ability to retain knowledge while adapting to new conditions. Therefore, it confirms the validity of our choice to leverage replay in our lifelong anomaly detection research.

Contributions of Kamil Faber: In this work, Kamil Faber was responsible for co-developing the benchmarks as well as carrying out the experiments. Kamil Faber was co-author of all sections of the paper.

This paper was accepted for publication in the Machine Learning journal (IF 7.5, 140 points). It is currently undergoing the publication process.

Note: In this paper, we started using the term “continual” instead of “lifelong” due to its rising popularity.

Noname manuscript No.
(will be inserted by the editor)

From MNIST to ImageNet and Back: Benchmarking Continual Curriculum Learning

Kamil Faber · Dominik Zurek · Marcin Pietron · Nathalie Japkowicz · Antonio Vergari* · Roberto Corizzo*

Received: date / Accepted: date

Abstract Continual learning (CL) is one of the most promising trends in recent machine learning research. Its goal is to go beyond classical assumptions in machine learning and develop models and learning strategies that present high robustness in dynamic environments. This goal is realized by designing strategies that simultaneously foster the incorporation of new knowledge while avoiding forgetting past knowledge. The landscape of CL research is fragmented into several learning evaluation protocols, comprising different learning tasks, datasets, and evaluation metrics. Additionally, the benchmarks adopted so far are still distant from the complexity of real-world scenarios, and are usually tailored to highlight capabilities specific to certain strategies. In such a landscape, it is hard to clearly and objectively assess models and strategies. In this work, we fill this gap for CL on image data by introducing two novel CL benchmarks that involve multiple heterogeneous tasks from six image datasets, with varying levels of complexity and quality. Our aim is to fairly evaluate current state-of-the-art CL strategies on a common ground that is closer to complex real-world scenarios. We additionally structure our benchmarks so that tasks are presented in increasing and decreasing order of complexity – according to a curriculum – in order to evaluate if current CL models are able to exploit structure across tasks. We devote particular emphasis to providing the CL community with a rigorous and reproducible evaluation protocol for measuring the ability of a model to generalize and not to forget while learning. Furthermore, we provide an extensive experimental evaluation showing that popular CL strategies,

Kamil Faber, Dominik Zurek, Marcin Pietron
AGH University of Science and Technology, Cracow, 30059, Poland
{kfaber,dzurek,pietron}@agh.edu.pl

Antonio Vergari
University of Edinburgh, Edinburgh, EH8 9AB, UK
avergari@ed.ac.uk

Nathalie Japkowicz, Roberto Corizzo
American University, Washington, DC, 20016, USA
{japkowic,rcorizzo}@american.edu

* Equal Supervision
Corresponding Author: Roberto Corizzo E-mail: rcorizzo@american.edu

when challenged with our proposed benchmarks, yield sub-par performance, high levels of forgetting, and present a limited ability to effectively leverage curriculum task ordering. We believe that these results highlight the need for rigorous comparisons in future CL works as well as pave the way to design new CL strategies that are able to deal with more complex scenarios.

Keywords continual learning, lifelong learning, curriculum learning, neural networks, computer vision, image classification

1 Introduction

Continual Learning (CL), also known as Lifelong Learning, is a promising learning paradigm to design models that have to learn how to perform *multiple tasks* across different environments over their lifetime [44] [1]. Ideal CL models in the real world should be able to quickly adapt to new environments and tasks while perfectly retaining what they learned in the past, thus only increasing, and not decreasing, their performance as they experience more tasks. In practice, this is quite challenging due to the hardness of generalizing from one environment to another when there is a huge *distribution shift* between them [16,30,35,10], and to the fact that models tend to (sometimes catastrophically) *forget* what they learned for previous tasks.

The great attention around this paradigm has brought many communities to focus on how to address these challenges, including reinforcement learning [5] [1] and anomaly detection [22] [13]. It is noteworthy that significant efforts in CL have been devoted to computer vision, leading to a large number of proposed models [16,35,2,28,12,54,45,27], where the most common task is to learn models that can classify different kinds of images while preventing catastrophic forgetting or quickly adapting to new image classes or image datasets. Every new model has been evaluated in a slightly different setting – *using a different dataset, evaluation metrics and learning protocols* – thus generating a number of CL learning and evaluation schemes. The result is that *the benchmark panorama of CL in computer vision is quite fragmented*, and therefore it has become tougher to measure catastrophic forgetting and domain adaptation in a fair and homogeneous way for the many CL models we have in the literature these days. Furthermore, all previous evaluation protocols are designed to highlight some specific model characteristics and, as such, are generally over-simplified w.r.t. real-world data [14].

For example, one of the most popular evaluation protocols for CL models in computer vision is to design different tasks to classify different (subsets of the) classes of a single dataset [33,52]. The most prominent example is splitMNIST in which the 10 digits from MNIST [15] are (usually) divided into 5 tasks consisting of 2 digits each. Similar approaches are proposed for CIFAR10 [31], and TinyImagenet [34]. Other datasets, such as Continuous Object Recognition (CORe50) [38], specifically designed for CL, still make the same assumptions to generate tasks. Clearly, these protocols are not suited to detect distribution shifts due to the high inter-task similarity. Consequently, catastrophic forgetting is much easier to prevent in these cases. Therefore the reported metrics for models evaluated in this way can be overly optimistic.

[1] To uniform the language and enhance the readability of the paper we adopt the unique term continual learning (CL).

Table 1 Benchmarks comparison considering only multi-dataset benchmarks. Columns refer to: i) supporting multiple heterogeneous tasks; ii) varying task complexity and quality; iii) evaluating curriculum strategies; iv) rigorous way to measure generalization and forgetting; and v) exactly reproducible out-of-the-box. In addition, we consider the coverage of class (CI) and task-incremental (TI) learning settings. The symbols have the following meaning: ✓- criterion is covered; ✓- criterion is covered at some part or with some limitations; x- criterion is not covered at all.

Benchmark	i)	ii)	iii)	iv)	v)	CI)	TI)
Imagenet/Places365 to VOC/CUB/Scenes [35]	✓	x	x	✓	✓	x	✓
Imagenet to Places365 [41]	✓	x	x	✓	✓	x	✓
5-Datasets [20]	✓	✓	x	✓	✓	✓	x
RecogSeq [3] [33]	✓	✓	x	✓	✓	✓	✓
M2I, I2M *(ours)*	✓	✓	✓	✓	✓	✓	✓

To deal with domain shifts, researchers have recently started to sample tasks from two different datasets. For instance, [16] proposed to train and evaluate a model on Imagenet first and then challenge its performance on the Places365 dataset. [35] considers more scenarios, starting with Imagenet or Places365, and then moving on to the VOC/CUB/Scenes datasets. Few works propose more advanced scenarios built on top of more than two datasets. The two most prominent examples are the so-called 5-datasets [20] and RecogSeq [3], which provide models with more challenging scenarios than previous attempts, increasing the number of considered datasets to 5 and 8, respectively. Unfortunately, those datasets provide a similar task complexity due to the limited differences across datasets. Furthermore, when different datasets are employed, it is important to "calibrate the meaning" of the employed metrics, taking into account the number of classes involved in each task.

Despite all this progress, we argue that there is still not a robust and standardized evaluation benchmark for the many CL models in the literature. We argue that a modern benchmark for CL should provide the following aspects. First, ***multiple heterogeneous tasks*** that do not restrict to a single set of concepts, e.g., digits in MNIST or SVHN or naturalistic images as in Imagenet or CIFAR10. Second, ***a varying quality and complexity of the tasks***, e.g., alternating from black and white (B&W) to RGB images and vice-versa, considering different image sizes, and a number of concepts. Third, a way to systematically evaluate if ***learning on a curriculum*** of task complexities help with domain generalization and catastrophic forgetting. For example, evaluating if a model trained on B&W digits can better generalize to B&W letters and then to RGB digits and letters, or if learning them in the inverse order is more beneficial. Fourth, ***a rigorous way to measure generalization and forgetting*** in terms of modern backward and forward transfer metrics [19] in a number of different evaluation scenarios, i.e., when classes or tasks are introduced incrementally [52]. Lastly, all results should be ***exactly reproducible out-of-the-box***. We argue that all the previous CL works discussed above do not consider one or more of these criteria, as highlighted in Table 1. In addition to the five desiderata, we also cover both class and task-incremental learning settings, which is not usually the case for other surveyed works. In this paper, we aim to overcome these limitations.

Specifically, the contributions of the paper are as follows:

– We propose *a set of benchmarks* built on 6 image datasets ordered in a curriculum of complexity – *from MNIST to TinyImageNet* (M2I) and back *from TinyImageNet to MNIST* (I2M) – that simultaneously satisfies all the above desiderata. These benchmarks have varying task complexity, starting with simple digits and going to complex naturalistic images and vice versa (see Figure 1);
– We provide an exhaustive experimental evaluation including 10 state-of-the-art continual learning methods, covering the key categories of approach (architectural, regularization, and rehearsal) in both class and task-incremental settings, which naturally fit our multi-task curriculum learning scenario, and evaluating results using the most recent metrics adopted in the continual and lifelong learning community.

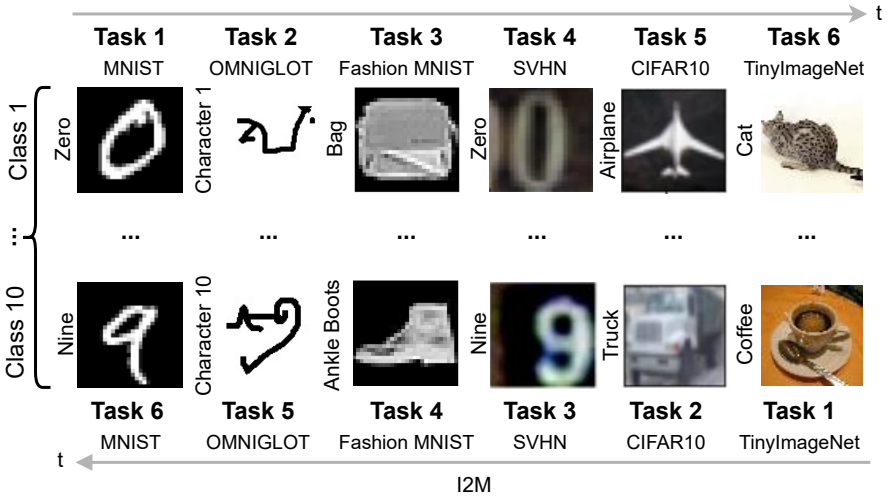


Fig. 1 The Proposed M2I and I2M continual learning benchmarks. There are 6 different tasks, each sampled from a different dataset: MNIST [15], OMNIGLOT [32], Fashion MNIST [53], SVHN [43], CIFAR10 [31] and TinyImageNet [34]. Tasks are organized in two curriculum ordering, from simple to harder (left to right) and backward (right to left). Every task sports 10 classes, as to make the performance metric meaning intuitive and faithful.

2 Background

2.1 CL scenario types

A wide range of scenarios was designed and discussed in recent studies to design effective CL models while trying to reflect real-world challenges. In image classification, two main scenarios are the most widely adopted: i) ***task-incremental*** [17] and ii) ***class-incremental*** [6]. In both scenarios, the model has to learn new tasks, which are presented sequentially. Each incoming task provides the model with new, previously unseen classes, that need to be incorporated.

A common characteristic for both mentioned scenarios is the availability of task boundaries, which make the CL method aware that a new task is presented. The most relevant difference between the two scenarios is the availability of task labels, which provide the model with additional information on which task is being processed at the moment, during both training and inference. Specifically, a task-incremental scenario assumes the availability of task labels, whereas in class incremental learning, this information is not available.

It is worth stressing that the same data presented in different types of scenarios can yield significantly different results, since certain CL methods may be tailored for task-incremental scenarios, and as such the exploitation of task labels improve their performance, while they may significantly suffer in class-incremental scenarios, where this information is not available.

The most widely adopted benchmark for class and task-incremental scenarios is split-MNIST [29] consisting of 5 tasks. It leverages the original MNIST, separating it into five tasks, each containing two digits. In this class-incremental scenario, the model is not aware of what the current task is. It is only aware of the fact that it encountered a new task and needs to adjust itself. During the testing phase, the model is also not informed about which set of digits is currently provided, so the model has to classify one of the ten classes (digits 0-9). On the other hand, in the task-incremental variant of split-MNIST, the model is aware of which task is currently being presented, and only decides whether the image belongs to the first or the second class of the current task. This prediction, combined with information about the current task id, leads to the specific digit prediction.

Less commonly, certain scenarios relax the assumptions of class and task-incremental scenarios. Authors in [39] propose new scenarios that tackle the presentation of new training patterns of both known and unknown classes (New Instances and Classes - NIC), which include real-world challenges identified in some applications that were not considered before in continual learning scenarios. A recent trend in class-incremental learning is to adopt the repetition of previously encountered concepts in the learning scenario, which softens the disruption of previous knowledge [14]. Another example of constraint relaxation is domain-incremental scenario [5], where new distributions of the same classes are presented over time, as well as task-agnostic scenarios [21, 23], where neither task labels nor task boundaries are available, and reliance on external methods is necessary to detect task changes. Another interesting direction is that of online continual learning, which aims to learn directly from a data stream with shifting distribution [11, 18].

Overall, despite the widespread adoption of class-incremental and task-incremental settings, these studies highlight the opportunity for new continual learning scenarios that entail additional real-world complexities. Similarly, the same trend of extending CL towards challenging real-world conditions can be observed in studies that propose novel benchmarks/datasets.

The CLEAR benchmark [37] proposes scenarios with a natural temporal evolution of visual concepts, entailing challenges similar to domain incremental learning settings. Another recent benchmark named CLiMB [50] puts emphasis on multimodal data, evaluating candidate CL models and learning algorithms on their forgetting, knowledge transfer, as well as their downstream low-shot transfer capability on both multimodal and unimodal tasks. The LECO benchmark [36] addresses the problem of refinement to ontologies with text data in continual

learning, introducing a new ontology of "fine" labels that describe specific concepts and refine old ontologies of "coarse" labels that describe general concepts (e.g., dog breeds that refine the previously observed dog). The work in [25] proposes a practical real-time evaluation of continual learning, in which the stream does not wait for the model to complete training before revealing the next data. The authors perform experiments with the CLOC dataset [9], which contains 39 million time-stamped images with geolocation labels. Authors in [42] propose a novel dataset in the remote sensing domain, built as a combination of three previously introduced datasets containing images from airborne, satellite, and drone sources.

From a CL scenario viewpoint, in our study, we adopt class-incremental and task-incremental settings as they naturally fit the multi-task nature of our curriculum learning scenario, where we are interested in assessing how effectively models incorporate diverse classes belonging to different datasets by increasing their difficulty over time. From a benchmark viewpoint, our work presents differences from the aforementioned studies. Specifically, differently than CLEAR and the studies in [25] and [42], we attend to a multi-task overview where multiple heterogeneous datasets are analyzed, also resorting to a curriculum learning ordering of tasks. Moreover, unlike CLiMB, which puts emphasis on multi-modal data, and LECO, which analyzes textual data, our focus is strictly on the assessment of model longevity in the computer vision domain, which was not investigated before.

2.2 CL Strategies

From a broad perspective, CL strategies belong to three main groups: using ***regularization***, ***dynamic architectures***, and ***rehearsal*** (also known as experience replay). In this paper, we consider popular and largely adopted CL strategies. We now describe each strategy and the rationale for its adoption in the benchmarks.

Regularization strategies influence the model weights adjustment process that takes place during model training in the attempt to preserve knowledge of previously learned tasks. The regularization strategies considered include Elastic Weight Consolidation (EWC) [29], Learning without Forgetting (LwF), Synaptic Intelligence (SI) [54], and Memory Aware Synapses (MAS) [2]. LwF [35] aims at achieving output stability through knowledge distillation. When a new task is observed, the new model is incentivized to predict values that are close to the outputs of the model learned prior to this task. EWC [29] and SI [54] adopt a weighted quadratic regularization loss, which penalizes moving weights that are important for previous tasks. The EWC loss is based on the Fisher Information Matrix, which presents a higher computational complexity than the surrogate loss used in the SI method. Similarly, Memory Aware Synapses (MAS) [2] estimates the cumulative importance of model weights as new tasks are encountered, penalizing changes to weights that are crucial for previously learned tasks. Shifting the focus on dynamic architectures, Progressive Neural Networks (PNN) is one of the first attempts to consider node expansion in a neural network architecture to accommodate different tasks [46]. CWRStar [39] adapts weights exclusively for the last layer before the prediction layer, freezing all previous layers. AR1Star extends this capability by also tuning the representation layers [39]. Finally, rehearsal strategies considered include GDumb [4], Replay [45], Gradient Episodic Memory (GEM) [16], and Average Gradient Episodic Memory (AGEM) [12]. GDumb [4] is a greedy

strategy that stores samples for all classes in a buffer, and uses them to iteratively retrain a model from scratch. It should be noted that GDumb was seen as a thought-provoking experiment showing that a performance close to state-of-the-art CL strategies could be achieved with a less sophisticated approach not specifically designed for CL problems. For this reason, it may be regarded as a baseline rather than a CL strategy. Replay [45] follows a similar approach but stores a balanced number of samples per task, which are used to fine-tune previously trained models. A variant of this approach using generative models is pseudo-rehearsal [47]. GEM [16] is a fixed-size memory that stores a subset of old patterns and influences the loss function through inequality constraints. AGEM [12] is a revised version of GEM that performs averaging to increase efficiency.

The rationale for the adoption of the aforementioned strategies in our benchmark is that they are heterogeneous in terms of groups of approach, and are particularly prevalent in the CL community. They represent the foundations in the CL field, and are often used to assess the competitiveness of emerging CL methods with respect to consolidated and diversified approaches. Moreover, they are easy to use and favor reproducibility, thanks to publicly available tools such as the Avalanche library [40].

2.3 CL evaluation protocol and metrics

The standard evaluation procedure applied in continual image classification assumes the availability of a set of tasks, each defined with a set of classes. The learning scenario consists of N tasks $T = t_1, t_2, \ldots, t_n$ where the model has to learn new tasks without forgetting previous tasks.

Metrics in continual learning usually focus on assessing the performance of a model (e.g., its accuracy) with respect to (at least one of) three crucial properties: *i)* performance on newly encountered tasks; *ii)* performance retention capabilities on previously learned tasks (i.e., the *ability to avoid or mitigate forgetting*); and *iii)* knowledge transfer from learned tasks to new ones (i.e., *the ability to generalize over newly occurring challenges*). The first works in CL proposed three metrics: average accuracy, backward transfer, and forward transfer to measure the above desiderata [16]. However, in their original definition, only the performance of the model after learning all tasks was considered.

Instead, we consider model performances for all tasks and at all stages of the learning process, as understanding how performance changes before and after every task can provide several insights into the strengths and weaknesses of every model [19]. For simplicity, we will be storing the partial model performance, measured as classification accuracy, in a matrix R whose entries $\mathsf{R}_{i,j}$ represent the accuracy on a given task j after learning task i.

Average Accuracy (ACC) – It measures the average accuracy of the model after learning each task, evaluating only the current and all previously learned tasks:

$$\mathsf{ACC} = \sum_{i \geq j}^{N} \mathsf{R}_{i,j} / (N(N-1)/2), \tag{1}$$

defined as the average performance over all tasks the model has seen so far.

Backward Transfer (BWT) – It measures the impact of learning new tasks on the performance of all previously learned tasks. Negative backward transfer

indicates that learning a new task is harmful to the performance of previously learned tasks (this issue is known as forgetting):

$$\mathsf{BWT} = \sum_{i=2}^{N} \sum_{j=1}^{i-1} (\mathsf{R}_{i,j} - \mathsf{R}_{j,j})/(N(N-1)/2), \tag{2}$$

defined as the average amount of forgetting presented by the model on the overall scenario.

Forward Transfer (FWT) – It measures the impact of learned tasks on the performance of tasks learned in the future:

$$\mathsf{FWT} = \sum_{i<j}^{N} \mathsf{R}_{i,j}/(N(N-1)/2), \tag{3}$$

defined as the average model performance on yet unseen tasks.

3 Our benchmarks: M2I and I2M

In Section 1 we pointed out essential desiderata for continual learning benchmarks that are designed to reflect real-life environments and challenges. In the following, we further elaborate on each criterion, providing a rationale for its importance, and we describe how our benchmark tackles these challenges.

First, it is important to ***consider multiple heterogeneous tasks***. The rationale is that since continual learning models should adapt to new and unprecedented situations, as human beings usually act in real environments, they should be evaluated on sequences of heterogeneous tasks. While common benchmarks focus on homogeneous tasks, such as different classes of handwritten digits (e.g. as in splitMNIST), heterogeneous tasks have the advantage of reflecting more realistic cases where the model is challenged by unprecedented tasks with great diversity. To deal with multiple heterogeneous tasks, our benchmark leverages 6 largely-varying image classification datasets: MNIST (handwritten digits) [15], Omniglot (alphabets) [32], Fashion MNIST (clothing items) [53], SVHN (street view house numbers) [43], CIFAR10 (small real-world images) [31], and TinyImagenet (multi-domain large-scale real-world images) [34]. Each dataset is regarded as a task, resulting in a learning scenario with six tasks with heterogeneous characteristics. We provide more details about the datasets included and the preprocessing they underwent into the benchmark in Table 2.

Second, it is important to devise scenarios with ***varying quality and task complexity***, since an ideal model should present generalization capabilities dealing with easy, moderate, and difficult tasks at the same time, as found in the real world. Ideal scenarios should avoid simplistic sequences of tasks with high task similarity, and prefer introducing new tasks that are different enough from the previous one, thus challenging the model in a significant way. This aspect should comprise having tasks on images varying in terms of visual and chromatic quality and difficulty of classification. Our benchmarks take this into consideration as they include very complex multi-domain real-world image classification such as TinyImagenet (harder classification), as well as handwritten digit recognition in MNIST (easier classification), and letter recognition in different alphabets in Omniglot (moderate difficulty). This choice of datasets creates ambitious but realistic challenges for CL strategies, allowing us to test their limitations.

Third, the hardness of each task is relative to the ordering in which the task is presented to the model. E.g. task ordering is important for us humans as we do not learn challenging new tasks from scratch but, instead, incrementally build up the necessary skills to perform these new tasks, leveraging a combination of skills learned in the past. We would require the same efficiency from a continual learner. Therefore, it is crucial to evaluate models ***learning on a direct or inverse curriculum***. The adoption of direct curriculum learning – learning on tasks of increasing complexity – in conventional machine learning research showcased that significant improvements in generalization can be achieved, increasing the speed of convergence of the training process [8] [24] [49].

When it comes to CL, however, direct and inverse curriculum learning are overlooked. Indeed, in the best cases, multiple random task orderings are provided in addition to a single task order. To properly consider curriculum learning, our benchmark considers curriculum learning by devising a task order according to their difficulty. To this end, we consider model performance as a proxy for task complexity. Specifically, we compute model accuracy on single datasets when considered in isolation. Table 2 shows the performance achieved by a WideVGG9 model with the different datasets considered in our study. These results create the conditions to define the ordering of datasets as tasks in our M2I and I2M benchmarks. The scenario starts with MNIST (black & white handwritten digits), which is regarded as an easy task. The following tasks are Omniglot (alphabets) and Fashion MNIST (clothing items), which present a spike of complexity compared to MNIST. Subsequently, SVHN (street view house numbers) brings real-world complexity by introducing images gathered from cameras with colors. CIFAR10 presents the same challenges of real-world colored images and extends them with more challenging patterns encountered in complex objects. Finally, the highest level of complexity is provided by multi-domain large-scale images from TinyImagenet. In addition to the direct curriculum direction where tasks are ordered as described (from MNIST to TinyImageNet, aka M2I), we also cover the opposite case of decreasing order of difficulty (from TinyImageNet to MNIST, aka I2M).

Fourth, ***rigorous way to measure generalization and forgetting***. The most important aspect of continual machine learning is to design strategies and models that are able to incorporate new tasks during their lifespan, without forgetting previous tasks. Metrics such as BWT and FWT are introduced for this reason, see Section 2.3 However, they can be cumbersome to interpret or lose their meaning, depending on the learning setting at hand. For instance, FWT is ill-defined in a class-incremental scenario since the model will never predict classes that were never presented before. Another example is that of multi-dataset benchmarks where tasks contain a varying number of classes. Specifically, tasks with a reduced number of classes will exhibit a random performance that is higher (e.g., 0.5 for 2 classes) than tasks with a higher number of classes (e.g., 0.1 for 10 classes). As results are generally aggregated (i.e., averaged) across tasks [35] [41], [20], [3], [33], CL metrics will be hard to interpret due to a different reference point for random performance. Ideal benchmarks should take these aspects into consideration to make sure that the calculation and the interpretation of results are correct.

To consider this aspect, we designed each task in our benchmark to contain 10 classes. In the case of MNIST and Fashion MNIST, SVHN, and CIFAR10, we use all classes. In the case of TinyImageNet and Omniglot, we select 10 classes. For TinyImagenet, we use Egyptian cat; reel; volleyball; rocking chair; lemon; bullfrog;

Table 2 Overview of original datasets involved in our benchmarks. The datasets present heterogeneous characteristics, i.e., domains and technical quality. For TinyImagenet, we select the following classes: Egyptian cat; reel; volleyball; rocking chair; lemon; bullfrog; basketball; cliff; espresso; plunger. As for Omniglot, we select characters from the alphabet of the Magi. Accuracy refers to WideVGG9 performance on single datasets used to estimate task complexity.

Dataset	Colors	Size	Classes	Available images	Accuracy
MNIST	BW	28x28	10	70 000	0.98
Omniglot	BW	105x105	1632	32 460	0.92
Fashion MNIST	BW	28x28	10	70 000	0.88
SVHN	RGB	32x32	10	630 420	0.85
CIFAR10	RGB	32x32	10	60 000	0.67
TinyImagenet	RGB	64x64	200	100 000	0.64

basketball; cliff; espresso; plunger. As for Omniglot, we select classes corresponding to characters from the Alphabet of the Magi. This setting allows us to preserve a high interpretability of all the resulting metric values overcoming the limitation of tasks with an imbalanced number of classes, where interpretability can be lost. Furthermore, to deal with class imbalance, we align the size of majority classes to that of minority classes. By doing so, we isolate the learning setting and avoid typical issues that arise in imbalanced learning, which might undermine the analysis of the final results.

Fifth, ***exactly reproducible out-of-the-box***. Many benchmarks are not reproducible due to the lack of precise details on model configurations and experimental settings. This issue is exacerbated when the code is unavailable and it is required to implement the scenario and the evaluation scheme from scratch. In other cases, when the code is available, it is not general enough to be leveraged in different settings, e.g. when comparing with the latest models and strategies. To this end, our benchmark is implemented on top of Avalanche [40] – the state-of-the-art open-source library for CL. This choice ensures the reproducibility of the experiments and paves the way for the adoption and extension of the benchmark for future research. The code for our benchmarks is publicly available at the following repository URL: `https://github.com/lifelonglab/M2I_I2M_benchmark`.

It is worth noting that, despite the adoption of class and task-incremental scenarios, our benchmark can be easily extended to different emerging scenarios, such as incremental data learning [18], in task-free and task-agnostic settings. To this end, a procedure can select a subset of datasets according to their complexity. Each batch may present multiple tasks, i.e. data from all datasets in the subset. After a number of batches, the procedure can remove the less (more) complex dataset and add a new more (less) complex dataset according to the curriculum ordering defined in our benchmark. It is conceivable that this procedure would let some tasks disappear while letting others appear, as frequently observed in online learning, but in increasing (or decreasing) order of difficulty, giving place to a curriculum learning scenario.

4 Experiments and discussion

We carry out experiments involving both the *task-incremental* and *class-incremental* CL scenario types described in Section 2.1, the CL strategies devised in Section

2.2, and the CL evaluation protocol and metrics defined in Section 2.3. We run an exhaustive series of experiments on our proposed M2I and I2M benchmarks for CL, resulting in 156 complete experiments (considering M2I and I2M with 14 CL strategies, 2 scenario types - Class-incremental and Task-incremental, and 3 model backbones) and 936 runs (model training and evaluation). We aim to answer the following research questions:

- **RQ1)** Do our benchmarks provide challenging scenarios for state-of-the-art CL strategies as discussed in Section 2.2? That is, are these strategies still as accurate and robust w.r.t the metrics defined in Section 2.3 as originally introduced in their papers when exposed to M2I and I2M?
- **RQ2)** Can state-of-the-art CL strategies leverage direct and indirect curriculum task ordering to maximize their backward and forward transfer? Alternatively, do different task orderings with varying task complexity have an impact on the final performance?

We first detail the experimental setup of our experiments and then provide an in-depth discussion of the results gathered. For the curious reader, the short answer to both questions is that, overall, the state-of-the-art models underperform when executed on our challenging benchmarks, despite many of these models were supposed to be robust to catastrophic forgetting and multiple tasks.

4.1 Experimental setup

As mentioned in Section 3, our benchmark provides multiple heterogeneous tasks with varying quality and task complexity. For instance, 3 of the 6 tasks contain black and white images, whereas the remainder contain colored images. Moreover, the image size varies across all tasks. There may be different ways to deal with different image channel types and sizes, which can have an impact on the final performance. However, we recognize that finding the optimal solution is an open challenge for researchers working with our benchmark, and it is out of the scope of this paper. For simplicity, for image sizes, we adopt the most frequently adopted approach, which consists of resizing all images to the same size (64×64). To deal with different image channels, we consider the largest number of channels (RGB) for all tasks (3) and replicate the single-channel encountered in BW images to all 3 channels. We recall that, in order to provide a rigorous way to measure generalization and forgetting, we balance class sizes by taking 500 images from each of them for both the training and evaluation phases. By doing so, we isolate possible issues deriving from class imbalance from our evaluation.

Network architecture. We leverage three commonly used model backbones in CL with different parameter sizes as to measure the effect of overparametrization w.r.t. our performance metrics in CL. Each network architecture is being used across all strategies. We employ a Wide VGG9 [48] as a smaller neural network (4.5M parameters), EfficientNet-b1 [51] as a mid-size alternative (7.8M parameters), and ResNet34 [26] as a large-size state-of-the-art model backbone (63.5M parameters). The hyperparameter configuration used in the experiments is: { *epochs=50, learning_rate=0.001, momentum=0.9* }. Optimization takes place through Stochastic Gradient Descent (SGD) using the Cross-Entropy loss. We experimented

with different negative powers of 10 for the configuration of the learning rate as suggested in [7], For the number of epochs, we experimented with similar values to those reported in the original publications of CL strategies [45][3]. Preliminary experiments showed that different configurations did not provide a significant difference in terms of performance metric values. For PNN, our setting differs from other strategies due to the fact that the Avalanche implementation provides support only for fully connected layers, leading to the impossibility of pairing the learning strategy with specific CNN model backbones (WideVGG9, EfficientNet, ResNet34). To this end, we matched the number of fully connected layers with those of our adopted CNN model architectures, i.e., 9, 24, and 34 layers, in an attempt to simulate their capacity. We also note that our PNN results are computed exclusively in the task-incremental setting since task identifiers are not available in the class-incremental setting.

CL strategies. In addition to the state-of-the-art CL strategies covered by our experiments and described in Section 2.2, we adopt three additional baseline approaches that loosely correspond to lower and upper bound model performance:

- **Naive (fine-tuning)**: The model is incrementally fine-tuned without considering any mechanism to preserve past knowledge, which, in principle, should yield a high degree of forgetting. This strategy allows us to compare the performance (in terms of accuracy) and forgetting (in terms of backward transfer) of smarter CL strategies.
- **Cumulative**: New data is accumulated as it comes, and the model is retrained using all available data. The rationale for this baseline is to simulate upper-bound performance assuming full knowledge of the data, and unlimited computational resources to deal with stored data (storage) and model retraining (time). Cumulative can also be regarded as a variant of Replay with unlimited memory. This baseline is interesting since it allows us to estimate the accuracy that could be achieved at a much higher computational cost.
- **Multiple Single-Task Expert (MSTE)**: A new model is created as soon as a new task is presented in the scenario. It can be regarded as a way to simulate upper-bound model performance despite a few unrealistic assumptions, such as unlimited computational resources to deal with additional models and availability of task identifiers (which make this baseline suitable only for Task-incremental scenarios).

4.2 Discussion: RQ1

We present our results both as aggregated metrics computed after all the tasks have been learned in Tables 3 – 8 as well as disaggregated accuracy results evaluating every task past task after learning a new one (as entries in the matrix R, see Section 2.3) as heatmaps shown in Figures 2 – 13. These heatmaps allow us to understand at a finer grain what are the failure modes of a strategy and whether certain tasks are harder than another. We now discuss four different settings, comprising either a class-incremental or task-incremental scenario and two task orderings (M2I or I2M). We start from employing the VGG9 model backbone.

4.2.1 VGG9

For both M2I and I2M (see Tables 3 – 4), aggregated ACC, BWT and FWT (only for task-incremental scenarios, see Section 3) are disappointing for all strategies discussed in Section 2.2. They do not achieve a positive backward transfer, and all highlight a systematic catastrophic forgetting, while the forward transfer floats around the chance level (10%[2]). Staple strategies such as LwF, MAS, and SI are generally comparable with the Naive strategy (i.e., just applying fine-tuning).

We inspect rankings over the ACC score to see if any notable trend manifests between different scenario types. We found that AGEM is very weak in the class-incremental setting (ranked 12 and 10, respectively), but quite robust in the task-incremental setting (ranked 6 and 5). Surprisingly, GEM seems robust in both settings (ranked 3, 4, 5, and 10 across the four settings). CWRStar achieves a moderately high position in the class-incremental setting (ranked 5 in the ranking for M2I). While its ranking is surprisingly high, we remark that the raw performance is clearly unsatisfactory when considered in absolute terms in this context. A much lower ranking is observed in the task-incremental setting despite the slight improvement in its performance. Overall, CWRStar appears ineffective in preventing catastrophic forgetting across all tasks in our settings, and it appears that it just focuses on memorizing the first task. We also observe that PNN presents a remarkably low performance, which consistently ranks 14 throughout both scenarios.

The method that seems to be the least prone to forgetting is Replay. This result is surprising since the total memory size chosen for the replay buffer in the experiment is just 200 samples. It is also interesting to observe that Replay presents a performance that is quite close to Cumulative (an $8\% - 18\%$ decrease in accuracy across the four mentioned settings) using a fraction of available data (less than 1%). Surprisingly, GenerativeReplay performs quite poorly when compared to standard Replay, although it outperforms many other strategies and presents a rank of 3 or 4 in all scenarios. As expected, Cumulative presents the best performance across three out of four learning settings. MSTE outperforms Cumulative in the I2M scenario - Task-incremental setting. However, both strategies should be seen as an unrealistic upper bound, since Cumulative assumes that infinite memory and training time are allowed for the model, whereas MSTE assumes the ability to add new models continuously. On a different note, the positive results confirm that the scenarios designed in our benchmark are reasonable and can be, in principle, learned by the model but current CL strategies. While these have shown to be reliable in conventional CL scenarios, they are not bulletproof and present limited robustness when exposed to more complex scenarios, such as our M2I (**RQ1**).

Observing the heatmaps in Figure 2 – 5 allows us to zoom in and pinpoint the performance drops of different strategies on specific tasks. In this context, observing a decreasing performance on previously learned tasks is a clear manifestation of forgetting. Results are quite negative for M2I in the class-incremental setting (see Figure 2). GEM preserves a good performance until the third task is presented and then dramatically drops in the following tasks due to their increasing complexity. GDumb presents a high performance on the second task throughout the entire

[2] We remark that this trend is easy to spot and understand in our benchmarks as all tasks sports only 10 classes.

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.220 (8)	-0.271	0.395 (10)	-0.224	0.102
LwF	0.222 (7)	-0.270	0.349 (12)	-0.083	0.096
MAS	0.215 (9)	-0.260	0.440 (7)	-0.212	0.100
SI	0.206 (11)	-0.255	0.419 (8)	-0.219	0.107
AGEM	0.188 (12)	-0.241	0.472 (6)	-0.161	0.101
GEM	0.572 (3)	-0.074	0.613 (5)	-0.053	0.099
GenerativeReplay	0.558 (4)	-0.142	0.616 (4)	-0.133	0.102
Replay	0.755 (2)	-0.038	0.730 (3)	-0.086	0.101
CWRStar	0.324 (5)	-0.044	0.356 (11)	-0.019	0.094
PNN			0.091 (14)	0.000	0.093
Naive	0.213 (10)	-0.261	0.411 (9)	-0.228	0.093
GDumb	0.304 (6)	-0.040	0.235 (13)	-0.071	0.092
Cumulative	0.868 (1)	0.004	0.819 (2)	0.012	0.102
MSTE			0.872 (1)	0.000	0.100

Table 3 Experimental results (Wide-VGG99 – I2M) in terms of average performance (and rank) for all CL strategies grouped by type (from top to bottom: regularization, rehearsal, architectural, and baseline) in two learning settings (class-incremental, task-incremental)

scenario but an unsatisfactory performance on all other tasks. This behavior likely depends on the fact that it is possible to learn the second task (Omniglot) with a limited number of samples, whereas this is too difficult for all other tasks. All other strategies, except Replay and Cumulative, struggle to preserve the knowledge of previous tasks and are successful at learning the last task exclusively, as evidenced by the very low-performance scores. GenerativeReplay presents a slightly different behavior, where the performance on certain tasks (0 and 2) is preserved, while the method appears unreliable on other tasks (1, 3, 4, and 5). The results for M2I in the task-incremental setting (see Figure 4) showcase a higher overall performance, with lower forgetting than the class-incremental setting. For instance, it can be observed that MAS and SI is able to preserve much more knowledge for some tasks, while its forgetting was rather drastic in the class-incremental setting.

For I2M in the class-incremental setting (see Figure 3), a similar behavior to the class-incremental counterpart of M2I can be observed, with drastic forgetting, which is even worse than the M2I scenario. As for I2M in the task-incremental setting (see Figure 5), it is also interesting to observe worse results than in the M2I task-incremental setting. This is also counterintuitive, as successfully learning a harder task should provide the model with enough knowledge not to perform so poorly on much simpler tasks that, as MNIST, might require learning only simple edge detectors. We conjecture that this behavior might depend on the fact that once a model is presented with very different but complex tasks earlier in the scenario (e.g., ImageNet and CIFAR10) it might have a harder time learning to abstract useful features for simpler tasks later.

4.2.2 EfficientNet

Results on M2I and I2M (see Tables 5 – 6) show that the Naive strategy achieves a performance that is close to some of the CL strategies (e.g. MAS, SI, EWC) but

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.190 (9)	-0.202	0.340 (7)	-0.161	0.126
LwF	0.216 (6)	-0.238	0.262 (11)	-0.105	0.088
MAS	0.223 (5)	-0.241	0.342 (6)	-0.148	0.130
SI	0.205 (7)	-0.223	0.323 (9)	-0.170	0.134
AGEM	0.175 (10)	-0.180	0.345 (5)	-0.094	0.131
GEM	0.297 (4)	-0.031	0.269 (10)	0.005	0.117
GenerativeReplay	0.351 (3)	-0.204	0.423 (4)	-0.139	0.136
Replay	0.550 (2)	-0.047	0.571 (3)	-0.061	0.135
CWRStar	0.079 (12)	-0.037	0.202 (12)	-0.011	0.107
PNN			0.101 (14)	0.000	0.073
Naive	0.199 (8)	-0.215	0.334 (8)	-0.160	0.131
GDumb	0.155 (11)	-0.052	0.147 (13)	0.000	0.085
Cumulative	0.735 (1)	0.012	0.663 (2)	0.018	0.120
MSTE			0.695 (1)	0.000	0.100

Table 4 Experimental results (Wide-VGG99 – I2M) in terms of average performance (and rank) for all CL strategies grouped by type (from top to bottom: regularization, rehearsal, architectural, and baseline) in two learning settings (class-incremental, task-incremental)

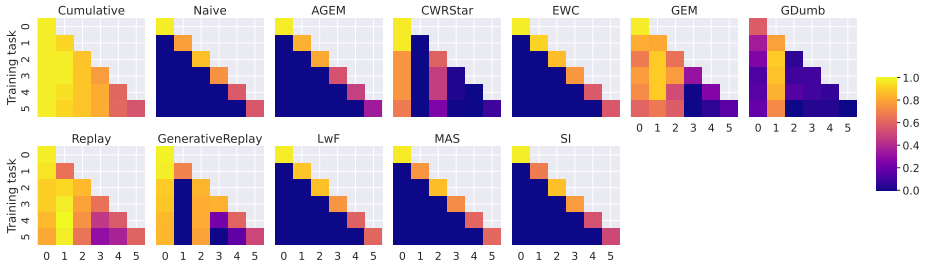


Fig. 2 Experimental results (Wide-VGG9 – M2I – Class-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

is significantly inferior to top performing strategies (Replay, Cumulative, MSTE). When comparing class-incremental and task-incremental settings for M2I, some methods appear significantly more robust in the latter, with a simultaneous increase in their performance and position in the ranking[3]. This is the case for AGEM (ranked 10 and 7, respectively). GEM preserves its ranking (5) in both settings, while improving its performance in the task-incremental setting. For I2M, comparing class and task-incremental settings, the same phenomenon can be observed for two methods: AGEM (ranked 10 and 8, respectively), SI (ranked 8 and 6, respectively). Two methods preserve their ranking in both settings: MAS and EWC (ranked 9 and 7, respectively). Some strategies present a rather stable behavior in the two learning settings, since they appear to preserve their ranking. For M2I, this is the case for GDumb, LwF, GenerativeReplay, and Replay. For I2M, this phenomenon applies to CWRStar, GDumb, GenerativeReplay, and Replay. Similarly to results

[3] It is important to track both aspects, since the task-incremental setting is fundamentally easier than class-incremental, and observing only the absolute performance of the methods is not indicative of an improvement.

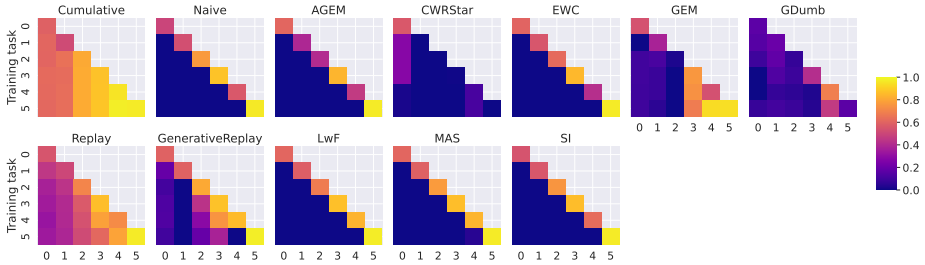


Fig. 3 Experimental results (Wide-VGG9 – I2M – Class-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

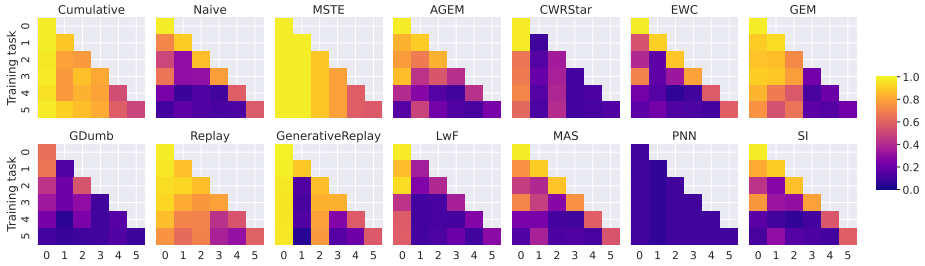


Fig. 4 Experimental results (Wide-VGG9 – M2I – Task-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

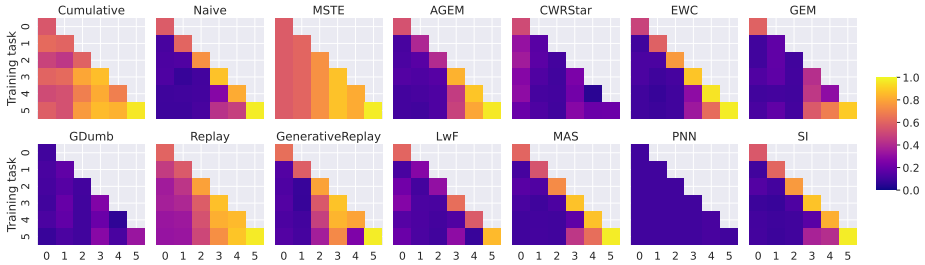


Fig. 5 Experimental results (Wide-VGG9 – I2M – Task-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

obtained with VGG9, Cumulative showcases the best performance in three out of four learning settings, while MSTE is the top-ranked strategy in one out of four cases. Therefore, in absolute terms, the performance of informed strategies can be regarded as unsatisfactory, as it appears significantly lower than Cumulative and MSTE. This result suggests that even increasing the parametrization of the backbone model does not provide these staple CL strategies to significantly improve over our simpler baselines in complex benchmarks such as our M2I and I2M (**RQ1**).

Shifting our focus to the heatmaps in Figure 6 – 9, we are able to analyze in detail the forgetting of the different strategies throughout the experimental scenario.

Figure 6 shows a vast amount of forgetting across all strategies. Some exceptions can be sparsely observed. For instance, MAS preserves its performance on task 0

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.180 (9)	-0.222	0.272 (11)	-0.199	0.106
LwF	0.165 (11)	-0.205	0.240 (12)	-0.124	0.096
MAS	0.187 (7)	-0.190	0.276 (10)	-0.198	0.104
SI	0.184 (8)	-0.231	0.277 (9)	-0.205	0.099
AGEM	0.179 (10)	-0.224	0.328 (7)	-0.171	0.091
GEM	0.312 (5)	-0.056	0.420 (5)	-0.043	0.096
GenerativeReplay	0.517 (3)	-0.143	0.551 (4)	-0.102	0.097
Replay	0.655 (2)	-0.041	0.605 (3)	-0.098	0.102
CWRStar	0.326 (4)	-0.014	0.367 (6)	0.001	0.092
PNN			0.091 (14)	0.000	0.100
Naive	0.205 (6)	-0.257	0.290 (8)	-0.220	0.103
GDumb	0.029 (12)	-0.007	0.099 (13)	0.000	0.100
Cumulative	0.834 (1)	0.003	0.656 (2)	0.025	0.101
MSTE			0.784 (1)	0.000	0.100

Table 5 Experimental results (EfficientNet – M2I) in terms of average performance (and rank) for all CL strategies grouped by type (from top to bottom: regularization, rehearsal, architectural, and baseline) in two learning settings (class-incremental, task-incremental)

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.197 (7)	-0.182	0.299 (7)	-0.169	0.100
LwF	0.203 (5)	-0.182	0.241 (11)	-0.072	0.091
MAS	0.178 (9)	-0.164	0.284 (9)	-0.148	0.105
SI	0.193 (8)	-0.177	0.302 (6)	-0.143	0.105
AGEM	0.157 (10)	-0.134	0.289 (8)	-0.101	0.122
GEM	0.218 (4)	-0.052	0.276 (10)	-0.052	0.119
GenerativeReplay	0.313 (3)	-0.179	0.390 (4)	-0.094	0.124
Replay	0.417 (2)	-0.019	0.438 (3)	-0.058	0.099
CWRStar	0.053 (11)	-0.035	0.171 (12)	-0.020	0.103
PNN			0.096 (14)	0.000	0.095
Naive	0.202 (6)	-0.193	0.305 (5)	-0.154	0.108
GDumb	0.029 (12)	0.000	0.100 (13)	0.000	0.092
Cumulative	0.606 (1)	0.021	0.596 (1)	0.015	0.118
MSTE			0.556 (2)	0.000	0.100

Table 6 Experimental results (EfficientNet – I2M) in terms of average performance (and rank) for all CL strategies grouped by type (from top to bottom: regularization, rehearsal, architectural, and baseline) in two learning settings (class-incremental, task-incremental).

after learning task 1, before dropping to values that are close to zero for previously encountered tasks. CWRStar preserves a remarkably high performance on the first task, but a very limited ability to incorporate new tasks. This result is in contrast with what was observed with VGG9, where the performance on the first task was preserved but decaying as new tasks are presented. This phenomenon may depend on the number of layers involved in the model backbone since EfficientNet is a much larger model, and the weight adaptation strategy used in CWRStar exclusively involves the last layer. Interestingly, we observe that GenerativeReplay is unable to preserve task 1, while it proves to be better than Replay in remembering task 0.

However, Replay presents a more diffused ability to preserve all tasks, as shown by its higher average performance. Moving to I2M in the class-incremental setting (see Figure 7), a noteworthy result is GEM preserving knowledge of task 3 throughout the entire scenario, while not being able to preserve its performance on the other tasks. In this setting, CWRStar is fundamentally unable to learn any of the tasks presented in the scenario.

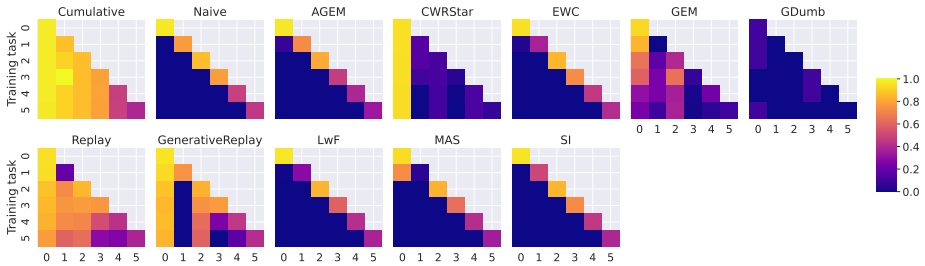


Fig. 6 Experimental results (EfficientNet – M2I – Class-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

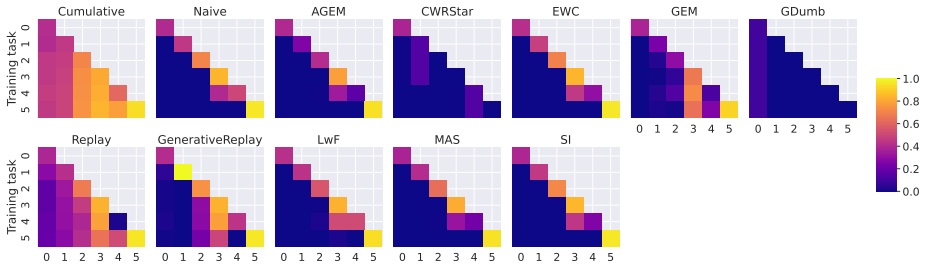


Fig. 7 Experimental results (EfficientNet – I2M – Class-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

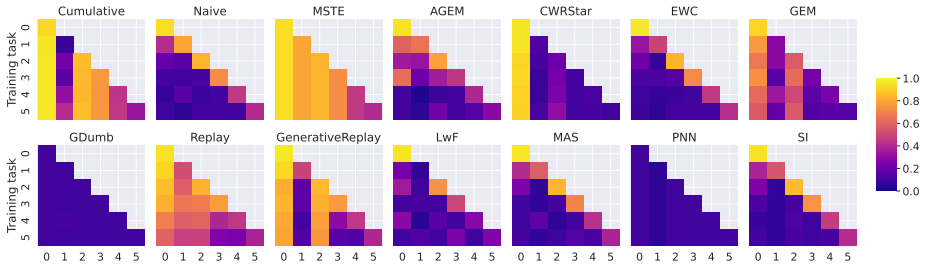


Fig. 8 Experimental results (EfficientNet – M2I – Task-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

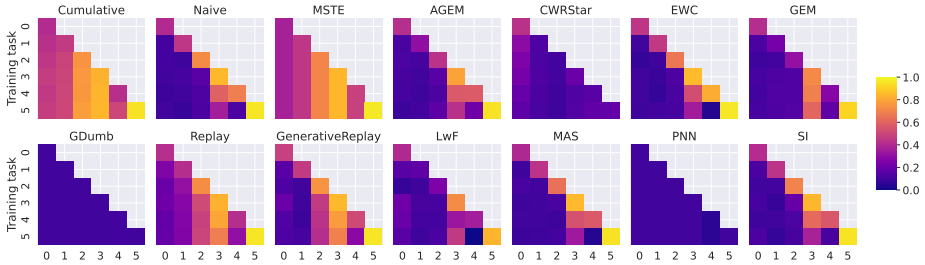


Fig. 9 Experimental results (EfficientNet – I2M – Task-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

Interestingly, task similarity between two tasks, manifested by positive backward transfer, allows for improvement on previously learned tasks in some instances. In Figure 8, for instance, learning task 4 is, in some cases, beneficial for the model's performance on task 1, as observed for MAS, GEM, and Naive.

In the task-incremental setting (see Figure 9), GEM presents a similar behavior to that observed in class-incremental on task 2, but also preserves knowledge of task 0 throughout the entire scenario, whereas the performance on other tasks is fundamentally sub-optimal. In this setting, however, CWRStar behaves as in the M2I class-incremental setting, i.e., the performance on task 0 is preserved throughout the entire scenario.

4.2.3 ResNet

Results on M2I and I2M (see Tables 7 – 8) show that the Naive strategy is relatively close to some of the CL strategies (e.g., EWC, SI, LWF) but is clearly less effective than other strategies (Replay, GenerativeReplay, Cumulative, MSTE). As observed with other model backbones, a comparison of class-incremental and task-incremental settings for M2I, highlights that some methods are more accurate in the latter and achieve a higher ranking. This phenomenon is observed for AGEM (ranked 7 and 6, respectively), Naive (11 and 8, respectively), and EWC (ranked 10 and 9, respectively). Similarly, in the I2M scenario, some of the strategies present an improved performance in the task-incremental setting: Naive (ranked 11 and 9, respectively), and CWRStar (ranked 12 and 11, respectively). Two strategies preserve their ranking across the two settings: EWC and MAS (ranked 8 and 7, respectively). Two exceptions can be identified: LwF and GDumb, which exhibit a lower ranking in the task-incremental setting (12, and 13, respectively) when compared to class-incremental (10, and 6, respectively). Strategies with stable behavior include Replay, GenerativeReplay, MSTE, and Cumulative in the two learning settings, which appears to preserve its ranking.

Cumulative confirms top-performing scores in three out of four settings. MSTE achieves the highest rank in the M2I - task-incremental setting. Overall, we observe that, unexpectedly, adopting a larger model such as ResNet did not achieve significantly better results than other model architectures with a reduced number of parameters and did not alter the considerations drawn in our discussion with other models.

Similarly to what we observed with other model backbones, Figures 10-13 show that all strategies are largely affected by forgetting. However, it is worth noting that some strategies preserve a certain degree of knowledge for specific tasks. Notable examples include CWRStar (see Figure 10), which preserves full knowledge of task 0 across the entire learning scenario. In the same scenario, GenerativeReplay is quite effective in knowledge preservation for tasks 0 and 2. Replay presents a wider retention capability that extends to multiple tasks when compared to CWRStar and GenerativeReplay. GDumb effectively incorporates and preserves knowledge of tasks 0, 1, and 2, whereas it struggles to learn the following tasks. All other strategies are essentially ineffective at retaining previously learned tasks, as shown by the large amount of forgetting.

Observing Figure 11, the most interesting result is that of GenerativeReplay, which appears ineffective in retaining knowledge of more difficult tasks, even when they are presented as the initial ones in the I2M scenario (tasks 0, 1, and 2). It is noteworthy that for Replay, we observe an improvement on task 0 after training task 3, which suggests the ability of the model and strategy to leverage tasks similarity.

An interesting takeaway in the task-incremental M2I setting (Figure 8) is that AGEM is less affected by forgetting, which is particularly visible on tasks 0, 1, and 2. A less visible but similar pattern can be observed with LwF, MAS, and SI.

Moving to the task-incremental I2M scenario (Figure 9), we observe that results are generally disappointing, with diffused forgetting. However, Replay and GenerativeReplay present an interesting behavior, where tasks encountered in the second half of the scenario are better retained. This may suggest that Replay and GenerativeReplay are able to deal with easier tasks better, even though they are introduced later in the scenario.

In general, we observe that ResNet presents a high degree of forgetting, as noticed with other model backbones. This phenomenon is emphasized in our heatmaps in Figures 10 – 13. Therefore, in absolute terms, the performance of informed strategies can be regarded as unsatisfactory, as it appears significantly lower than Cumulative. This result suggests that even increasing the parametrization of the backbone model does not provide these staple CL strategies to significantly improve over our simpler baselines in complex benchmarks such as our M2I and I2M (**RQ1**).

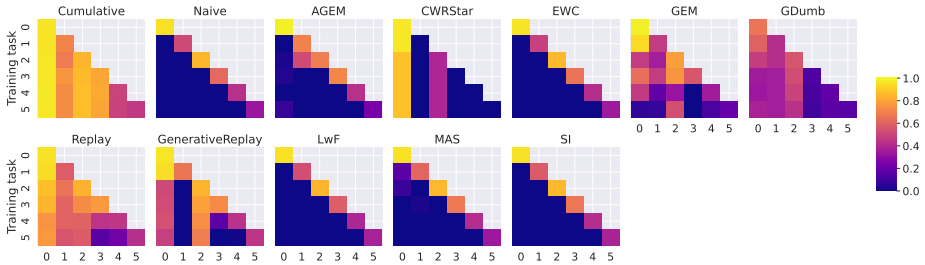


Fig. 10 Experimental results (ResNet – M2I – Class-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.175 (10)	-0.222	0.314 (9)	-0.213	0.103
LwF	0.174 (12)	-0.218	0.257 (13)	-0.124	0.100
MAS	0.195 (8)	-0.232	0.300 (11)	-0.200	0.099
SI	0.182 (9)	-0.229	0.301 (10)	-0.206	0.098
AGEM	0.208 (7)	-0.228	0.464 (6)	-0.179	0.093
GEM	0.412 (4)	-0.155	0.601 (5)	-0.088	0.099
GenerativeReplay	0.447 (3)	-0.162	0.632 (3)	-0.102	0.105
Replay	0.630 (2)	-0.087	0.612 (4)	-0.090	0.093
CWRStar	0.331 (6)	-0.006	0.403 (7)	0.013	0.097
PNN			0.089 (14)	0.000	0.097
Naive	0.175 (11)	-0.223	0.322 (8)	-0.198	0.089
GDumb	0.357 (5)	-0.033	0.294 (12)	-0.044	0.099
Cumulative	0.788 (1)	0.003	0.750 (2)	0.007	0.099
MSTE			0.784 (1)	0.000	0.100

Table 7 Experimental results (ResNet – M2I) in terms of average performance (and rank) for all CL strategies grouped by type (from top to bottom: regularization, rehearsal, architectural, and baseline) in two learning settings (class-incremental, task-incremental)

	Class-incremental		Task-incremental		
	ACC	BWT	ACC	BWT	FWT
EWC	0.191 (8)	-0.204	0.300 (8)	-0.117	0.104
LwF	0.188 (10)	-0.200	0.221 (12)	-0.076	0.094
MAS	0.196 (7)	-0.180	0.308 (7)	-0.138	0.118
SI	0.189 (9)	-0.201	0.288 (10)	-0.142	0.108
AGEM	0.221 (5)	-0.157	0.375 (6)	-0.110	0.145
GEM	0.264 (4)	-0.086	0.388 (5)	-0.073	0.131
GenerativeReplay	0.313 (3)	-0.131	0.424 (4)	-0.107	0.135
Replay	0.419 (2)	-0.074	0.443 (3)	-0.058	0.080
CWRStar	0.102 (12)	-0.026	0.228 (11)	-0.004	0.097
PNN			0.095 (14)	0.000	0.085
Naive	0.182 (11)	-0.191	0.291 (9)	-0.114	0.104
GDumb	0.204 (6)	-0.031	0.180 (13)	-0.017	0.101
Cumulative	0.614 (1)	0.006	0.562 (1)	0.004	0.104
MSTE			0.552 (2)	0.000	0.100

Table 8 Experimental results (ResNet – I2M) in terms of average performance (and rank) for all CL strategies grouped by type (from top to bottom: regularization, rehearsal, architectural, and baseline) in two learning settings (class-incremental, task-incremental).

4.2.4 Summary: RQ1

In summary, results observed across the two learning settings (class-incremental, task-incremental) in the two presentation orders (M2I, I2M) show unsatisfactory performance for all learning strategies and that catastrophic forgetting is a real burden for many of the covered methods.

Considering that the results observed are inferior when compared to what is commonly reported in continual learning research, we can argue that our benchmark provides more challenging conditions for the CL strategies. It is noteworthy that

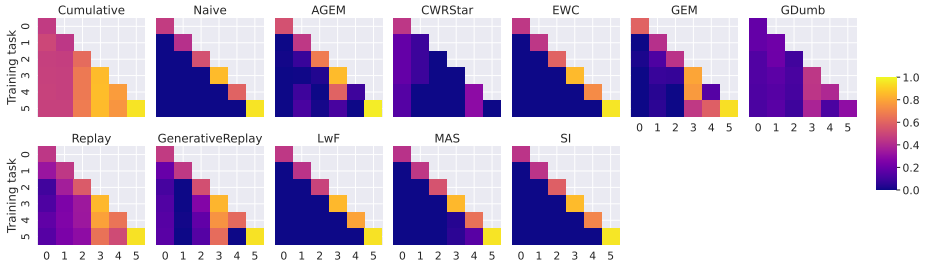


Fig. 11 Experimental results (ResNet – I2M – Class-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

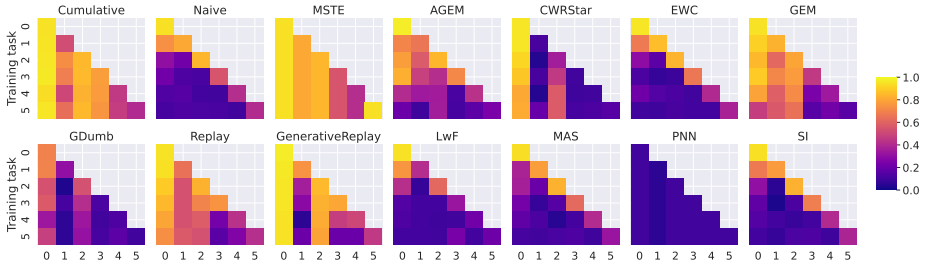


Fig. 12 Experimental results (ResNet – M2I – Task-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

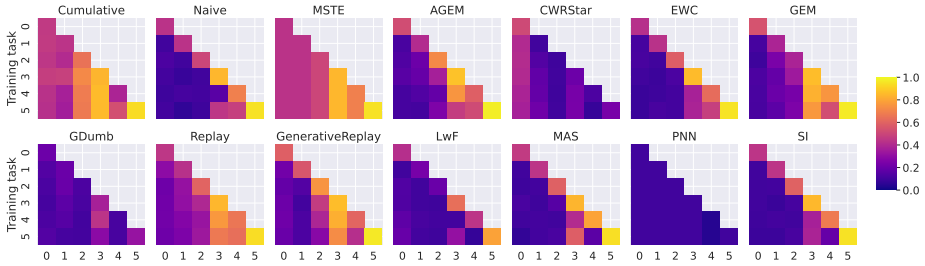


Fig. 13 Experimental results (ResNet – I2M – Task-incremental) in terms of disaggregated performance (ACC) on single tasks after learning previous tasks.

forgetting in CL strategies is also observed in perceptually similar tasks (e.g., MNIST, Omniglot, SVHN), as evident in our heatmaps. This behavior is indicative of the objective lack of robustness presented by CL strategies as they are exposed to tasks from different datasets. The five desiderata described in Section 3 and adopted to design our benchmarks set up a higher standard for the evaluation of CL strategies, and will hopefully stimulate the design and implementation of new, more robust strategies.

4.3 Discussion: RQ2

In this subsection, we focus on the assessment of the ability of CL strategies to leverage curriculum task ordering to maximize their performance when exposed to our benchmarks devised in Section 3.

To answer this question, we start by analyzing results in Tables 3 – 8, which show metric values for the two scenarios: M2I (direct curriculum learning) and I2M (inverse curriculum learning). Almost all methods present a better performance in the curriculum learning setting (M2I) when compared with the inverse curriculum setting (I2M). Comparing values in Tables 3 and 4, significant examples for VGG9 include Replay (from 0.550 to 0.755 in class-incremental and from 0.571 to 0.730 in task-incremental), GenerativeReplay (from 0.351 to 0.558 in class-incremental and from 0.423 to 0.616 in task-incremental), and GEM (from 0.297 to 0.572 in class-incremental and from 0.269 to 0.613 in task-incremental). Other CL strategies present a smaller margin of improvement. For instance, LwF (from 0.216 to 0.222 in class-incremental, and from 0.262 to 0.349 in task-incremental) and EWC (from 0.190 to 0.220 in class-incremental, and from 0.340 to 0.395 in task-incremental). Shifting the focus on results with EfficientNet (comparing Tables 5 and 6), examples include Replay (from 0.417 to 0.655 in class-incremental, and from 0.438 to 0.605 in task-incremental), GenerativeReplay (from 0.313 to 0.517 in class-incremental, and from 0.390 to 0.551 in task-incremental), and Cumulative (from 0.606 to 0.834 in class-incremental, and from 0.596 to 0.656 in task-incremental). Other CL strategies present a more limited but still significant improvement. For instance, CWRStar (from 0.053 to 0.326 in class-incremental, and from 0.171 to 0.367 in task-incremental), and GEM (from 0.218 to 0.312 in class-incremental, and from 0.276 to 0.420 in task-incremental). One counterexample, where the model's performance is higher in the inverse curriculum setting (I2M) is LwF (from 0.203 to 0.165 in class-incremental and from 0.241 to 0.240 in task-incremental). This result shows that different strategies behave differently when presented with a different task ordering. A similar pattern can be observed for ResNet (comparing Tables 7 and 8), where the biggest improvement in the class-incremental setting are presented by Cumulative (from 0.614 to 0.788), CWRStar (from 0.102 to 0.331), GenerativeReplay (from 0.313 to 0.447), GDumb (from 0.204 to 0.357), and Replay (from 0.419 to 0.630). Improvements for all other strategies appear minor. Major examples in the task-incremental setting include AGEM (from 0.375 to 0.464), Cumulative (from 0.562 to 0.750), CWRStar (from 0.228 to 0.403), GenerativeReplay (from 0.424 to 0.632), GDumb (from 0.180 to 0.294), Replay (from 0.443 to 0.612). The only exceptions where strategies do not improve in the M2I scenarios are identified with AGEM and EWC in the class-incremental setting. In the task-incremental setting, the exception is MAS.

Another interesting point pertaining to our research question is the opportunity to identify whether learning new tasks favors performance on previously learned tasks, emphasized in our heatmaps. This phenomenon may happen if the model is able to capture similarities between tasks that can be fruitfully leveraged for inference. To show some examples, we focus on task-incremental experiments. In the M2I scenario with VGG9, Figure 4 shows that different strategies (AGEM, MAS, SI) are able to improve performance on task 1 (Omniglot) after learning task 5 (TinyImageNet). We also observe that multiple strategies (AGEM, EWC, SI, MAS) can leverage the skills learned in task 3 (SVHN) to improve performance on task 1

(MNIST). This result is intuitive since learning the complexity of street numbers in images acquired with a camera strongly benefits the predictive capabilities on an easier dataset from a similar domain, i.e., MNIST. Similar behavior can be observed in Figures 8 and 12, where AGEM improves the performance on task 2 (Fashion MNIST) after learning task 5 (TinyImagenet). In the I2M scenario for all models: VGG9 (Figure 5), EfficientNet (Figure 9) and ResNet (Figure 13), we can observe that almost all strategies improve the performance on task 3 (Fashion MNIST) after learning task 5 (MNIST). This result shows that the knowledge learned from MNIST can boost the performance on more complex tasks learned before. Overall, results show that task ordering and task similarity can be leveraged to improve performance on a previously learned task, although the currently adopted CL strategies are sparsely able to yield this capability. This consideration paves the way for the design of new strategies that further leverage curriculum task ordering to boost forward and backward transfer (**RQ2**).

An additional consideration pertains to the connection between curriculum learning and the appropriateness of CL metrics in this context. While we are adopting state-of-the-art metrics that are recognized for properly measuring model accuracy and forgetting in CL scenarios, it should be noted that their interpretation could be counterintuitive in some specific scenarios, where tasks have a varying degree of complexity and metric values are compared with different task orderings. For instance, we note that MNIST is the simplest task and it is presented as the first task (M2I) in the curriculum learning setting. Therefore, it will be considered multiple times in the evaluation protocol, i.e., each time a new task is presented, which may boost the final average result presented by the accuracy metric. In turn, it is more likely for the curriculum learning setting to achieve higher average performance, when a simpler task is presented earlier in the scenario. This behavior poses issues in the interpretability of metric values, which are still unaddressed by currently available metrics. We believe that this aspect should be tackled by future work on metrics for CL.

4.3.1 Summary: RQ2

Overall, results observed across two learning settings (class-incremental, task-incremental) in the two presentation orders (M2I, I2M) show that current methods are not able to fully leverage curriculum learning. One reason may be the fact that most of the CL strategies are heavily impacted by forgetting since they are challenged by the complexities involved in our proposed benchmarks. Comparing the performance and behavior of the CL strategies between M2I and I2M scenarios, we can also observe that different task orderings significantly impact the final outcomes. On the other hand, methods appear to partially benefit from task similarity in some specific cases, as highlighted in our analysis of results. This outcome leads us to the consideration that task similarity could be further exploited by CL strategies to yield models that simultaneously use the knowledge acquired from different tasks to perform better in every single task.

5 Conclusions

In this work, we focused on the problem of benchmarking CL methods, which is often conducted in heterogeneous ways, and with significant simplifications for the learning setting. Specifically, we proposed two novel benchmarks that involve multiple heterogeneous tasks with varying qualities and complexities. Our benchmarks involved six image datasets in increasing (M2I) and decreasing (I2M) difficulty order, following the curriculum learning paradigm. The heterogeneity across datasets allowed us to inject realistic complexities into the learning scenario, resulting in challenging conditions for CL strategies. Particular emphasis was put on the rigorous and reproducible evaluation of model generalization capabilities and forgetting. Our extensive experimental evaluation showed that popular CL strategies, which are known to be robust on commonly adopted scenarios, fail to achieve satisfactory performance with our benchmarks. Moreover, CL strategies are affected by forgetting and are not able to effectively leverage curriculum task ordering to improve their performance and robustness, missing on the opportunity of simultaneously using knowledge from different tasks to perform better in every single task. Our results represent a starting point to assess the impact of curriculum learning on CL strategies. Future work includes the design of new CL strategies that are able to deal with the complexities devised in our benchmarks. Moreover, from an evaluation perspective, new metrics could be investigated to fully capture the spectrum of model behavior with different task orderings. Finally, an interesting line of research pertains to the analysis of the behavior of non-conventional CL strategies, which are not yet incorporated in known frameworks due to their emerging nature.

Appendix A: Hyperparameters of included strategies

In our study, as hyperparameters specific to each continual learning strategy, we adopt the configurations reported in their respective research papers. The chosen values are reported in Table 9. For general hyperparameters common to all strategies, we refer the reader to Section 4.

Table 9 Hyperparameter values adopted for all continual learning strategies.

Strategy	Hyperparameters
EWC	lambda = 1 mode = separate keep importance data = True
LwF	alpha = 2 temperature = 1
MAS	lambda reg = 1 alpha = 0.5 mini batch size = 128
SI	lambda = 1 epsilon = 0.001
AGEM	patterns per experience = 256 sample size = 1300
GEM	memory strength = 0.5
GenerativeReplay	replay size = 200 increasing replay size = False generator strategy = VAETraining
Replay	replay memory size = 200
CWRStar	cwr layer name = fully connected
PNN	number of columns = 200
GDumb	mem size = 200

Declarations

– **Funding** – The paper was supported by the Polish Ministry of Science and Higher Education allocated to the AGH UST.

– **Conflict of interest/Competing interests** – All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest in the subject matter or materials discussed in this manuscript.

– **Ethics approval** – Not applicable.

– **Consent to participate** – This study does not involve human subjects or any sensitive data.

– **Consent for publication** – This study does not involve human subjects or any sensitive data.

– **Availability of data and materials** – The data and materials to reproduce the experiments are available at the following repository URL: `https://github.com/lifelonglab/M2I_I2M_benchmark`

– **Code availability** – The code of the proposed benchmarks is available at the following repository URL: `https://github.com/lifelonglab/M2I_I2M_benchmark`

– **Authors' contributions** – Kamil Faber: *Data Curation, Investigation, Software, Visualization, Writing* – Dominik Zurek: *Data Curation, Investigation, Resources, Software, Writing (Review & Editing)* – Marcin Pietron, Nathalie Japkowicz: *Resources, Validation, Writing (Review & Editing)* – Antonio Vergari, Roberto Corizzo: *Conceptualization, Supervision, Methodology, Project Administration, Writing*

References

1. Abel, D., Jinnai, Y., Guo, S.Y., Konidaris, G., Littman, M.: Policy and value transfer in lifelong reinforcement learning. In: International Conference on Machine Learning, pp. 20–29. PMLR (2018)
2. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 139–154 (2018)
3. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (eds.) Computer Vision – ECCV 2018, pp. 144–161. Springer International Publishing, Cham (2018)
4. Ameya Prabhu, P.H.S.T., Dokania, P.K.: Gdumb: A simple approach that questions our progress in continual learning. Lecture Notes in Computer Science (LNIP) **12347** (2020)
5. Baker, M.M., New, A., Aguilar-Simon, M., Al-Halah, Z., Arnold, S.M., Ben-Iwhiwhu, E., Brna, A.P., Brooks, E., Brown, R.C., Daniels, Z., et al.: A domain-agnostic approach for characterization of lifelong learning systems. Neural Networks (2023)
6. Belouadah, E., Popescu, A., Kanellos, I.: A comprehensive study of class incremental learning algorithms for visual tasks. Neural Networks **135**, 38–54 (2021)
7. Bengio, Y.: Practical recommendations for gradient-based training of deep architectures pp. 437–478 (2012)
8. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: Proceedings of the 26th annual international conference on machine learning, pp. 41–48 (2009)
9. Cai, Z., Sener, O., Koltun, V.: Online continual learning with natural distribution shifts: An empirical study with visual data. In: Proceedings of the IEEE/CVF international conference on computer vision, pp. 8281–8290 (2021)
10. Cano, A., Krawczyk, B.: Rose: Robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams. Machine Learning **111**(7), 2561–2599 (2022)
11. Carta, A., Cossu, A., Hurtado, J., Lomonaco, V., Van de Weijer, J., Hemati, H., et al.: A comprehensive empirical evaluation on online continual learning. arXiv preprint arXiv:2308.10328 (2023)
12. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. Salk Institute for Biological Studies **arXiv:1812.00420** (2019)
13. Corizzo, R., Baron, M., Japkowicz, N.: Cpdga: Change point driven growing auto-encoder for lifelong anomaly detection. Knowledge-Based Systems **247**, 108756 (2022)
14. Cossu, A., Graffieti, G., Pellegrini, L., Maltoni, D., Bacciu, D., Carta, A., Lomonaco, V.: Is class-incremental enough for continual learning? Frontiers in Artificial Intelligence **5** (2022)
15. Cun, Y.L.: The mnist database of handwritten digits. http://yann. lecun. com/exdb/mnist/
16. David Lopez-Paz, M.R.: Gradient episodic memory for continual learning. arXiv **https://arxiv.org/abs/1706.08840** (2017)
17. De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. IEEE transactions on pattern analysis and machine intelligence **44**(7), 3366–3385 (2021)
18. De Lange, M., Tuytelaars, T.: Continual prototype evolution: Learning online from non-stationary data streams. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 8250–8259 (2021)

19. Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., Maltoni, D.: Don't forget, there is more than forgetting: new metrics for continual learning. arXiv preprint arXiv:1810.13166 (2018)
20. Ebrahimi, S., Meier, F., Calandra, R., Darrell, T., Rohrbach, M.: Adversarial continual learning. In: European Conference on Computer Vision, pp. 386–402. Springer (2020)
21. Faber, K., Corizzo, R., Sniezynski, B., Baron, M., Japkowicz, N.: Lifewatch: Lifelong wasserstein change point detection. In: 2022 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2022)
22. Faber, K., Corizzo, R., Sniezynski, B., Japkowicz, N.: Active lifelong anomaly detection with experience replay. In: 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), pp. 1–10. IEEE (2022)
23. Faber, K., Corizzo, R., Sniezynski, B., Japkowicz, N.: Vlad: Task-agnostic vae-based lifelong anomaly detection. Neural Networks (2023)
24. Gao, K., Wang, H., Cao, Y., Inoue, K.: Learning from interpretation transition using differentiable logic programming semantics. Machine Learning pp. 1–23 (2022)
25. Ghunaim, Y., Bibi, A., Alhamoud, K., Alfarra, M., Al Kader Hammoud, H.A., Prabhu, A., Torr, P.H., Ghanem, B.: Real-time evaluation in online continual learning: A new hope. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11888–11897 (2023)
26. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
27. Hihn, H., Braun, D.A.: Hierarchically structured task-agnostic continual learning. Machine Learning pp. 1–32 (2022)
28. Kang, H., Mina, R.J.L., Rizky, S., Madjid, H., Yoon, J., Hasegawa-Johnson, M., Ju-Hwang, S., Yoo, C.D.: Forget-free continual learning with winning subnetworks. ICML **x** (2022)
29. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., Hadsell, R.: Overcoming catastrophic forgetting in neural networks. arXiv **https://arxiv.org/abs/1612.00796** (2016)
30. Krawczyk, B.: Tensor decision trees for continual learning from drifting data streams. Machine Learning **110**(11-12), 3015–3035 (2021)
31. Krizhevsky, A.: Learning multiple layers of features from tiny images (2009)
32. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science **350**(6266), 1332–1338 (2015)
33. Lange, M.D., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G.G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**, 3366–3385 (2022)
34. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge
35. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017)
36. Lin, Z., Pathak, D., Wang, Y.X., Ramanan, D., Kong, S.: Continual learning with evolving class ontologies. Advances in Neural Information Processing Systems **35**, 7671–7684 (2022)
37. Lin, Z., Shi, J., Pathak, D., Ramanan, D.: The clear benchmark: Continual learning on real-world imagery. In: Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 2) (2021)
38. Lomonaco, V., Maltoni, D.: Core50: a new dataset and benchmark for continuous object recognition. In: S. Levine, V. Vanhoucke, K. Goldberg (eds.) Proceedings of the 1st Annual Conference on Robot Learning, *Proceedings of Machine Learning Research*, vol. 78, pp. 17–26. PMLR (2017). URL `https://proceedings.mlr.press/v78/lomonaco17a.html`
39. Lomonaco, V., Maltoni, D., Pellegrini, L.: Rehearsal-free continual learning over small non-i.i.d. batches. 1st Workshop on Continual Learning in Computer Vision at CVPR2020 (2019). URL `https://arxiv.org/abs/1907.03799`
40. Lomonaco, V., Pellegrini, L., Cossu, A., Carta, A., Graffieti, G., Hayes, T.L., De Lange, M., Masana, M., Pomponi, J., Van de Ven, G.M., et al.: Avalanche: an end-to-end library for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3600–3610 (2021)
41. Mallya, A., Lazebnik, S.: Packnet: Adding multiple tasks to a single network by iterative pruning. arXiv **https://arxiv.org/abs/1711.05769** (2017)
42. Marsocci, V., Scardapane, S.: Continual barlow twins: continual self-supervised learning for remote sensing semantic segmentation. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (2023)

43. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.: Reading digits in natural images with unsupervised feature learning (2011)
44. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: A review. Neural networks **113**, 54–71 (2019)
45. Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., Wayne, G.: Experience replay for continual learning. Advances in Neural Information Processing Systems **32** (2019)
46. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)
47. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc. (2017)
48. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
49. Song, H., Kim, S., Kim, M., Lee, J.G.: Ada-boundary: accelerating dnn training via adaptive boundary batch selection. Machine Learning **109**, 1837–1853 (2020)
50. Srinivasan, T., Chang, T.Y., Pinto Alva, L., Chochlakis, G., Rostami, M., Thomason, J.: Climb: A continual learning benchmark for vision-and-language tasks. Advances in Neural Information Processing Systems **35**, 29440–29453 (2022)
51. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning, pp. 6105–6114. PMLR (2019)
52. Van de Ven, G.M., Tolias, A.S.: Three scenarios for continual learning. arXiv preprint arXiv:1904.07734 (2019)
53. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
54. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. arXiv **https://arxiv.org/abs/1703.04200** (2017)

Chapter 12

WATCH: Wasserstein Change Point Detection for High-Dimensional Time Series Data

In this paper, we responded to the problem of change point detection in dynamic high-dimensional data by proposing WATCH – a novel Wasserstein distance-based change point detection method. WATCH provides robust detection of change points by modeling the current data distribution and leveraging threshold-ratio-based distance to detect change points adaptively. The utilization of Wasserstein distance allows for the comparison of sets of data points without limiting the distribution type.

We verify the capabilities of WATCH by an extensive experimental evaluation involving a large number of benchmark datasets. The results showcase the power of our method, as it outperforms all considered competitors in the majority of the cases.

Contributions of Kamil Faber: In this work, Kamil Faber co-designed and developed the WATCH method. Kamil Faber carried out the experiments and was co-author of all sections of the paper.

This paper was published at the 2021 IEEE International Conference on Big Data (CORE B-Rank; 70 points).

WATCH: Wasserstein Change Point Detection for High-Dimensional Time Series Data

Kamil Faber
Institute of Computer Science
AGH University of Science and Technology
Krakow, Poland
kfaber@agh.edu.pl

Roberto Corizzo
Department of Computer Science
American University
Washington, DC, USA
rcorizzo@american.edu

Bartlomiej Sniezynski
Institute of Computer Science
AGH University of Science and Technology
Krakow, Poland
bartlomiej.sniezynski@agh.edu.pl

Michael Baron
Department of Mathematics and Statistics
American University
Washington, DC, USA
baron@american.edu

Nathalie Japkowicz
Department of Computer Science
American University
Washington, DC, USA
japkowic@american.edu

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671962

***Abstract*—Detecting relevant changes in dynamic time series data in a timely manner is crucially important for many data analysis tasks in real-world settings. Change point detection methods have the ability to discover changes in an unsupervised fashion, which represents a desirable property in the analysis of unbounded and unlabeled data streams. However, one limitation of most of the existing approaches is represented by their limited ability to handle multivariate and high-dimensional data, which is frequently observed in modern applications such as traffic flow prediction, human activity recognition, and smart grids monitoring. In this paper, we attempt to fill this gap by proposing `WATCH`, a novel Wasserstein distance-based change point detection approach that models an initial distribution and monitors its behavior while processing new data points, providing accurate and robust detection of change points in dynamic high-dimensional data. An extensive experimental evaluation involving a large number of benchmark datasets shows that `WATCH` is capable of accurately identifying change points and outperforming state-of-the-art methods.**

***Index Terms*—Change Point Detection, Big Data, High-Dimensional Data, Time series.**

I. Introduction

Change point detection is of fundamental importance in modern data analysis problems characterized by dynamic time series data in which the statistical properties of the distribution change over time. One attractive characteristic of change point algorithms is their ability to analyze data in an unsupervised manner [1], [2], which is ideal in analytical settings where data flows in an unbounded way and without annotations, as in data streams [3]. The change point detection problem is also implicitly connected to anomaly detection [4]–[7] and concept drift detection in data streams [8], [9]. However, while change point detection focuses on the identification of points where the data distribution changes (and remains in that state for a certain amount of time), anomaly detection focuses on identifying out of distribution (even single) data points. Although a similarity between change point detection and concept drift detection can be noticed, change point detection assumes a more general connotation of change, whereas a more bounded characterization is made in concept drift detection [10], [11].

An analysis of the existing literature in change point detection reveals a large number of approaches. However, modern real-world applications such as traffic flow prediction [12], human activity recognition [13], and renewable energy plants monitoring in smart grids [14]–[16] and settings characterized by geo-distributed data [17], [18] are strongly characterized by multivariate and often high-dimensional data. One limitation of most change point detection methods described in the literature is that they perform well strictly with univariate data or multivariate data with a limited number of dimensions [19]. As soon as the data dimensionality increases, their performance presents a drop in terms of accuracy or a significant increase in terms of execution time, which represents a known obstacle for online data analysis algorithms that rely on timely feedback from the change point detection procedure.

In this paper, we address this gap by proposing `WATCH`, a method based on the Wasserstein distance specifically targeted for high-dimensional time series data. The method models an initial distribution and monitors its behavior while processing new incoming data points. Change points are identified considering the Wasserstein distance between data points and the modeled distribution, which evolves over time. When a change point is detected, the previously known distribution is replaced to reflect the new distribution. An extensive experimental evaluation shows that our method significantly outperforms state-of-the-art change point detection methods on high-dimensional datasets. Moreover, the method achieves competitive performance on many of the benchmark datasets commonly used in the change point detection community.

The remainder of the paper is structured as follows. Section II discusses an overview of existing change point detection

approaches in the literature. Section III describes our proposed method. Section IV provides details on our experimental setup, followed by Section V which discusses our results. Finally, Section VI concludes the paper, summarizing our contributions and showcasing directions for future work.

II. Background

In this section we give an overview of some popular change point detection approaches in the literature. Methods can be broadly grouped in three main categories: offline, Bayesian, and non-parametric. Offline approaches include likelihood ratio methods, which analyze the probability distributions of data before and after a candidate change point, in order to confirm the candidate as an actual change point if the two distributions are significantly different. One notable example is the CUSUM method, which triggers a detection when the cumulative sum of observations exceeds a given threshold value. A method known as At Most One Change [20] generalized this approach to testing for differences in the maximum likelihood. Typically, methods in this class calculate the logarithm of the likelihood ratios within two consecutive intervals and monitor it over time [21], [22]. Another variant of this method considers dimensionality reduction as a pre-processing step [23]. One major drawback of such methods is that they rely on predefined parametric models, which make them less flexible in real-world scenarios [2].

Some offline methods can discover multiple change points within a time series, performing a greedy binary segmentation according to a cost function. Examples of methods in this class are Binary Segmentation [24] and Segment Neighborhoods [25]. An interesting variant of the binary segmentation approach which leverages the CUSUM statistic is Wild Binary Segmentation [26]. Extensions leveraging pruning approaches and presenting an improved time efficiency include Pruned Exact Linear Time [27] and Robust Functional Pruning Optimal Partitioning [28], which is also robust to outliers. It is noteworthy that these methods are restricted to the analysis of univariate datasets.

Probabilistic methods include offline and online Bayesian methods. A non-conventional offline change point detection approach is the Prophet method: a Bayesian time series forecasting approach that supports time series with change points [29]. A popular online method in this class is Bayesian Online Change Point Detection [30] which is based on the assumption that a data sequence may be divided into non-overlapping states partitions and the data points in each state are independent and identically distributed according to some probability distribution. Methods in this class calculate the run length distribution according to Bayes' theorem and update the corresponding statistics. Change point prediction is then carried out by comparing probability values. Recent extensions include support for Model Selection [31] and spatio-temporal models [32]. Variants of Bayesian methods are based on the Gaussian Process (GP) method, which generalizes a Gaussian distribution as a collection of random variables, any finite number of which have a joint Gaussian distribution [33], [34].

TABLE I: Change point detection methods featured in our experiments.

Method	Name	Reference
Offline		
At Most One Change	AMOC	[20]
Binary Segmentation	BINSEG	[24]
Segment Neighborhoods	SEGNEIGH	[25]
Wild Binary Segmentation	WBS	[26]
Pruned Exact Linear Time	PELT	[27]
Robust Functional Pruning Optimal Partitioning	RFPOP	[28]
Bayesian		
Bayesian Online Change Point Detection	BOCPD	[30]
BOCPD with Model Selection	BOCPDMS	[31]
BOCPD with Spatio-temporal models	RBOCPDMS	[32]
Prophet	PROPHET	[29]
Non-parametric		
Nonparametric Change Point Detection	CPNP	[35]
Kernel Change-Point Analysis	KCPA	[36]
Energy Change Point	ECP	[37]

Non-parametric methods for change point detection are based on hypothesis testing, using a test statistic and a custom threshold. Examples of methods in this class include [35], Kernel Change-Point Analysis [36], and Energy Change Point [37]. It is worthwhile mentioning that non-conventional change detection approaches based on deep neural networks are also investigated in the literature [16], [38], even though the scope of this study focuses on purely statistical change point detection.

III. Method

In this section we describe `WATCH`, our novel change point detection approach based on the Wasserstein distance. We divide the discussion in two stages: in the first subsection we introduce the Wasserstein distance discussing its potential for the task at hand; in the second subsection we provide algorithmic details about our method.

A. Wasserstein distance

The Wasserstein metric is a distance function defined by a probability distribution on a given metric space M. An intuitive explanation of the metric provided in the literature views the distribution as a unit amount of earth (soil) piled on M, where the metric represents the minimum "cost" of turning one pile into the other. This cost is assumed to be the amount of earth that needs to be moved, multiplied by the mean distance it has to be moved. Because of this analogy, the metric is known in computer science as the earth mover's distance. Recent works in the machine learning literature incorporated this distance in model pipelines and showed its potential in providing disentangled representations [39].

According to the Wasserstein distance definition in [40], let $P(\mathbb{R}^d)$ be the set of Borel probability measures on $\mathbb{R}^d$ and $P_p(\mathbb{R}^d)$ be the subset of such measures with a finite moment of order $p \in [1, \inf)$. For P, Q $\in P(\mathbb{R}^d)$, let $\Gamma(\mathrm{P}, \mathrm{Q})$ be the set of probability measures γ on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals P and Q, i.e., such that $\gamma(B \times \mathbb{R}^d) = \mathrm{P}(B)$ and $\gamma(\mathbb{R}^d \times B) = \mathrm{Q}(B)$

for Borel sets $B \subseteq \mathbb{R}^d$. $dist(\dot{})$ is the Euclidean norm. The p-Wasserstein distance between P, Q $\in P_p(\mathbb{R}^d)$ is:

$$W_p(\mathrm{P},\mathrm{Q}) = \Big(\inf_{\gamma\in\Gamma(P,Q)} \int_{\mathbb{R}^d\times\mathbb{R}^d} dist(x-y)^p \; \partial\gamma(x,y)\Big)^{\frac{1}{p}}. \quad (1)$$

In terms of random variables X and Y with laws P and Q, respectively, the p-Wasserstein distance is the smallest value of $\{E(dist(X-Y))^p\}^{\frac{1}{P}}$ over all possible joint distributions $\gamma \in \Gamma(\mathrm{P},\mathrm{Q})$ of (X, Y). It is important to note that the original formulation of the Wasserstein distance is intractable in nature. Therefore, practitioners estimate it relying on an approximated implementation. Intuitively, given a k-dimensional vector space $V \subset \mathbb{R}^k$, we can map the distributions (P, Q) to corresponding sets of k-dimensional data points $G, H \subseteq V$. In our work, the Wasserstein distance W_P is calculated using the SAG approximated computational method as explained in [40]. From now on, we refer to the approximated version of the Wasserstein distance as $W_A \approx W_p$.

We advocate that using a distance metric for change point detection has the advantage of being directly applicable to a high number of dimensions, which does not apply for most change point detection approaches. In particular, the rationale for the adoption of the Wasserstein distance in our approach is that, unlike the Kullback-Leibler (KL) divergence, it is a probability metric that measures the work required to transport the probability mass from one distribution state to another. This characteristic provides a smooth and significant representation of the distance between distributions, making it a good candidate in domains where analyzing the similarity of outcomes is more relevant than an exact likelihood matching. Additionally, the Wasserstein distance is more informative and powerful since it does not require data distributions to be on the same probability space, unlike KL divergence.

Figure 1 shows an intuitive representation of meaningful information carried by the Wasserstein distance: although the measures between the red and the blue distributions are the same in terms of KL divergence, the Wasserstein distance measures "the work" required to transport the probability mass from the red state to the blue state.

It is noteworthy that the Wasserstein distance is gaining traction in recent machine learning methods and notably in neural network architectures such as Wasserstein Auto-Encoders (WAE), that share many of the properties of Variational Auto-Encoders (VAEs) such as stable training, encoder-decoder architecture, and a nice latent manifold structure, while also generating samples of higher quality [41]. As indicated in [39], a WAE encodes the data into a low-dimensional latent space similarly to a VAE. However, instead of regularizing each data point, it regularizes the averaged encoding distribution. This approach allows the encoding distribution to capture essential information and still match the prior distribution.

Our work builds upon distance-based change point detection approaches, which have shown success in the previous literature [42], [43]. Our intuition is that adopting a more informative distance metric such as the Wasserstein distance, which benefits from the properties described above,

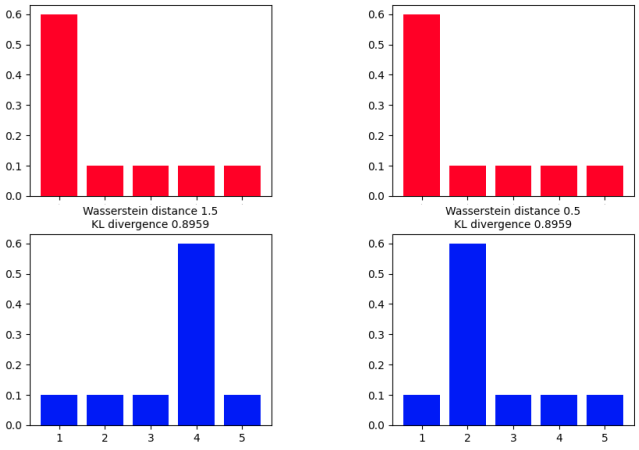


Fig. 1: Difference between Wasserstein distance and Kullbach-Leibler (KL) divergence.

could trigger more timely and accurate detection of change points. This assumption may be particularly relevant in high-dimensional datasets, where the large number of dimensions adds complexity to the change point detection task, causing phenomena such as collinearity, feature noise, and fractional distance when few of the features values change.

B. Algorithm

The input of the algorithm is a time series of data samples $\mathbf{I} = (I_1, I_2, \ldots, I_T)$, where each data sample is a d-dimensional vector: $I_t \in \mathbb{R}^d$. The parameters of the algorithm are the following:

- κ – the minimum number of points that must be present in the current representation of distribution to trigger change point detection. This parameter controls how many samples are required before knowing the current distribution well enough for change point detection.
- μ – the maximum number of points that can be present in the current representation of distribution. This parameter allows to control the size of the memory preventing possible bottlenecks deriving from a large number of samples.
- ϵ – the ratio controlling how distant the samples can be from the current distribution to be still considered a part of the same.
- ω – size of the mini-batches (non-overlapping windows) that define the data processing granularity. The two main reasons for the adoption of sliding non-overlapping windows are: *i)* minimizing the required computations compared to overlapping windows; *ii)* allowing a finer-grained data processing having low values of ω, which allows detecting changes in a timely manner.

Intuitively, our method learns a new representation of a distribution D in an unsupervised manner using an initial set of data points. Subsequently, the method incrementally stores newly available samples in a buffer. Once the desired

capacity of the buffer (which is controlled by the κ parameter) is reached, the change detection procedure is activated. In this process, we collect newly arriving points in mini-batches of length ω (sliding non-overlapping windows), and calculate the Wasserstein distance of every mini-batch B with respect to D. If the distance is lower than the desired threshold γ (controlled by the ϵ parameter), we qualify B as belonging to D. The threshold is determined as:

$$\eta(D,\ \epsilon) =\ \epsilon \max_{B \in D} W_A(B, D). \tag{2}$$

Consequently, D is updated to reflect the new data points. On the contrary, if the distance is higher than γ, we assume that the current mini-batch B being analyzed belongs to a different distribution. Therefore, we annotate this point in time as a change point and start creating a new distribution to replace D. Once D reaches its storage capacity controlled by μ, the oldest samples are removed to accommodate new ones, in order to keep an up-to-date representation of the distribution D. A pseudo-code of the algorithm is presented in Algorithm 1. A significant advantage of our approach is that it compares two sets of points instead of working on the parameters of specific distributions. This mechanism allows to compare distributions of unknown types, without limiting the scope to only the most popular ones, such as Normal distributions. In the following sections we describe our experimental setup and discuss our results.

Input : $\mathbf{I} = (I_1, I_2, \ldots, I_T)$ Time series data

Result: R: Set of change points

Parameters: κ Min points required in current distrib. before starting detection
μ Max points stored for current distrib.
ϵ Threshold ratio for the distance value
ω Mini-batch size

Set current distribution $D \leftarrow \emptyset$
$R \leftarrow \emptyset$
$\mathbf{B} \leftarrow$ Process $\mathbf{I}$ into sequence $(B_1, B_2, \ldots, B_{\frac{T}{\omega}})$ of non-overlapping mini-batches B_i of size ω
for $B_i \in \mathbf{B}$ **do**
  if $|D| < \kappa$ **then**
    $D \leftarrow D \cup B_i$
    if $|D| >= \kappa$ **then**
      Determine η from Eq. (2) with D and ϵ
    end
  else
    $v \leftarrow W_A(B_i, D)$
    if $v > \eta$ **then**
      Set change point $c \leftarrow i * \omega$
      $R \leftarrow R \cup \{c\}$
      $D \leftarrow \{B_i\}$
    else
      if $|D| < \mu$ **then**
        $D \leftarrow D \cup B_i$
        Determine η from Eq. 2 with D and ϵ
      end
    end
  end
end

Algorithm 1: Pseudo-code of the proposed `WATCH` method

IV. Experimental setup

In this study we adopt the Turing Change Point Detection (TCPD) benchmark [44] in order to compare the results obtained with our method with state-of-the-art algorithms. In the following subsection, we introduce the benchmark and motivate its adoption. Then, we discuss our additional datasets and the metrics used for the evaluation of results.

A. Turing Change Point Detection (TCPD) benchmark

This benchmark allows us to evaluate the 13 algorithms described in Table I on 42 time series datasets. There is also the possibility to extend the benchmark to new methods and datasets, which provides a solid ground for the evaluation of new change point detection algorithms designed by researchers and practitioners. It is also noteworthy that many existing algorithms work just on univariate data and cannot handle multivariate datasets. Moreover, the majority of the datasets (38 out of 42) available in the benchmark are univariate (or single-dimensional), and only 4 are multivariate (or multi-dimensional). Furthermore, these 4 datasets (*apple, bee_waggle_6, occupancy, run_log*) present a number of dimensions ranging from 2 to 4. For this reason, we extended the benchmark with 5 datasets with a significantly higher number of dimensions, ranging from 17 to 784. Further details of the added datasets are reported in Section IV-B. Table II presents all datasets used in our experimental evaluation.

There are two modes of experiments in the TCPD benchmark:

- *default* – in this mode the algorithms are run with default parameters;
- *best* – in this mode the algorithms are run with various combinations of parameter values (grid search) and the best configuration is selected. For our method, we consider the following parameter values as candidates in the grid search: $(\kappa = \{4, 10, 16, 32\}, \mu = \{10, 30, 100\}), \epsilon = \{0.5, 0.6, 0.7, ..., 1.9, 2, 2.5, 3, 5\}, \omega = \{3, 4\})$.

Summary results obtained with the *default* and *best* mode across all datasets are reported in Table III, whereas detailed results obtained with the *best* mode on each single dataset are reported in Table IV, V, VI, and VII.

B. Additional high-dimensional datasets

We added the following datasets to evaluate the performance of the methods on more challenging data presenting a higher number dimensions than 4:

1) **har:** The Human Activity Recognition dataset introduced in [45] contains records of 6 types of human

TABLE II: Multivariate datasets analyzed

Name	Samples	Dimensions
apple	622	2
bee_waggle_6	609	4
gas_sensor	1000	128
har	1623	561
mnist	131	784
movement	360	90
occupancy	509	4
run_log	376	2
traffic	135	17

activities including walking, sitting, standing. Given the sequential nature of the data, it is straightforward to determine change points based on activity changes in the labels for each human participant. We perform experiments separately for all human participants and report the average values of the metrics.

2) **gas_sensor:** The Gas Sensor Array Drift Dataset [46] contains measurements from 16 sensors exposed to 6 different gases at various concentration levels. Given the large number of dimensions (128) and time points (13910), we selected a portion of 1000 samples in order to allow all methods to complete their computation in a feasible amount of time. Change points are determined in the same way as in the **har** dataset, taking into account class changes in the time series data.
3) **movement:** The Libras movement dataset contains records of hand movements types in official Brazilian signal language [47]. There are 360 samples described by 90 dimensions. We determine the change points based on labels' changes in the same way as in the **har** dataset.
4) **mnist:** The MNIST dataset [48] contains images of handwritten digits. We reproduced a sequential scenario selecting 131 images, and a varying number of images (from 6 to 35) for each class to obtain a more challenging scenario with non-evenly spaced change points. Each image has a size of 28×28, which is flattened to obtain a 784-dimensional feature vector.
5) **traffic:** This dataset contains urban traffic records of the city of Sao Paulo in Brazil [49]. There are 135 samples described by 17 dimensions. We first perform discretization on the *slowness* attribute (continuous) to obtain a discrete attribute with three labels (*low*, *medium*, and *high*). Then, we determine change points based on label changes in the sequential data.

C. Metrics

In our experiments, we adopt the metrics provided by the authors of the TCPD benchmark [44] to keep consistency and to allow an easy comparison for future works. As indicated in [44], the existing metrics for change point detection can be divided between classification and clustering metrics. One metric for each category is adopted: F1-score for classification and Covering for clustering.

F1-score: Change point detection algorithms may be evaluated in terms of classification between change-point and non-change point classes [44]. Usually, datasets are affected by class imbalance due to a low number of change points. Therefore, metrics that are highly vulnerable to class imbalance are not suitable for this case. Examples of more robust metrics are:

- P – *Precision*: the ratio of correctly detected change points over the number of all detected change points, penalizing false alarms,
- R – *Recall*: the ratio of correctly detected change points over the number of actual change points.

The *F1-score* can be applied to provide a single performance measure, incorporating both Precision and Recall. The F1 score is defined as:

$$F_1 = \frac{PR}{P+R}. \tag{3}$$

As the TCPD benchmark may contain many ground truth sets of change points provided by human annotators, Precision and Recall are adjusted to handle this situation. This aspect implies that only detected change points that are not present in any ground truth are considered false positives, while the Recall is calculated as the average for each source of ground truth to encourage finding all change points from all ground truths (from all annotators). The details on those metrics are presented in [44] and [50]. The TCPD benchmark also considers the margin of error around the actual change points to allow for minor discrepancies. In the results provided by the original paper and in our work, we use a margin of error of value 5. This means that an early or delayed change point prediction is considered a true positive if it falls within 5 time steps from the ground truth.

Covering metric (Cover): The covering metric was initially presented in [51] to assess the quality of the segmentation for multiple segments (subsets of pixels representing separate objects) in the image domain. Simply put, the covering metric describes how well the predicted segments cover ground truth segments. In our case, each segment corresponds to a specific subsequence of contiguous points from the input time series. In order to define the covering metric, we first need to introduce the Jaccard Index. It is a statistic used for measuring the similarity of sample sets. It is also known as Intersection over Union, as it measures the overlap between the predicted segmentation and the ground truth. The Jaccard Index is defined as:

$$J(A, A') = \frac{|A \cap A'|}{|A \cup A'|} \tag{4}$$

The definition of the covering metric of partition G by partition G' is introduced in [51] for images segmentation and adjusted to time series data in [44]:

$$Cover(G', G) = \frac{1}{T} \sum_{A \in G} |A| \max_{A' \in G'} J(A, A') \tag{5}$$

To determine a single measure of performance while having a collection of ground truth partitions provided by the human

TABLE III: Results organized into sections based on dimensions of the data and the mode of experiments. The values correspond to the average of runs for each section. Blank values indicate that the method is not suited for multidimensional data. The highest values for each section of the table are shown in bold.

	Univariate				Multivariate			
	Default		Best		Default		Best	
Algorithm	Cover	F1	Cover	F1	Cover	F1	Cover	F1
AMOC	0.680	0.677	0.726	0.778				
BINSEG	**0.682**	**0.716**	0.760	0.833				
BOCPD	0.614	0.675	0.769	0.857	**0.486**	**0.558**	**0.717**	0.798
BOCPDMS	0.626	0.507	0.737	0.620	0.249	0.235	0.317	0.292
CPNP	0.514	0.594	0.538	0.648				
ECP	0.505	0.592	0.711	0.789	0.456	0.496	0.634	0.683
KCPA	0.068	0.121	0.619	0.679	0.162	0.187	0.565	0.662
PELT	0.666	0.692	0.706	0.766				
PROPHET	0.542	0.495	0.574	0.537				
RBOCPDMS	0.579	0.411	0.422	0.349	0.194	0.191	0.077	0.102
RFPOP	0.374	0.485	0.403	0.517				
SEGNEIGH	0.658	0.659	0.763	0.832				
WBS	0.319	0.408	0.417	0.519				
ZERO	0.573	0.658	0.573	0.658	0.255	0.325	0.255	0.325
WATCH	0.540	0.570	**0.797**	**0.898**	0.427	0.488	0.713	**0.836**

annotators and a partition given by an algorithm, the average of $Cover(G', G)$ for all annotators is computed.

D. Implementation

WATCH is implemented in Python v3.9 and numpy v1.19.5. We also used the R implementation of the Wasserstein distance provided in [40], and called it from the Python code using the rpy2 v3.4.5 bridge. All experiments are run on a machine with an Intel Core i7-8750H CPU, GeForce GTX 1050 Ti Mobile GPU and 32 GB of RAM.

V. Results and discussion

In the following, we discuss our experimental results in three subsections: *A)* summary results across all datasets considering *default* and *best* modes, *B)* detailed results on multivariate (multi-dimensional) datasets, and *C)* detailed results on univariate (uni-dimensional) datasets.

A. Summary results analysis

Table III shows the summary of the average results across all datasets and methods. The layout of the table organizes results into four sections based on dimensions of the data (univariate, multivariate), and the mode of the experiments (*default, best)*. The values correspond to the average of runs for each section.

Considering the results obtained for the *best* mode with multivariate datasets, the WATCH method outperforms all other ones in terms of F1. The second best performing method, BOCPD, has an average F1 of 0.798, while WATCH has an average F1 of 0.836, resulting in a 4.76% improvement. The other methods, ECP and KCPA, are strongly outperformed by 22.40% and 26.28%, respectively. Focusing on the Cover metric, the WATCH method performs very similarly to the best method BOCPD, reporting a sligthly lower value (0.55%). It is noteworthy, however, that WATCH achieves significantly better Cover than the ECP and KCPA. Focusing on the results obtained for univariate datasets, it is noteworthy that the proposed method WATCH outperforms other methods in terms of both Cover and F1 also for these simpler datasets.

B. Results on multivariate datasets

Results in terms of the F1 and Cover metrics values for multivariate datasets are reported in Table IV and V. In this work we are particularly focused on high-dimensional datasets since they most closely represent complex real-world scenarios that are not supported by the majority of change point detection methods (we note that only 6 out of 14 competitor methods considered in our experiments support the analysis of multivariate data). In the following, we briefly discuss results obtained on each multivariate dataset, whereas results on univariate datasets are reported separately in the following subsection. Results obtained by our method in terms of both F1 and Cover metrics follow a similar pattern. For the sake of conciseness, we focus our discussion on the F1 metric, which we believe is the most representative in most scenarios with time series data:

- **apple** – WATCH outperforms all other competitors, improving the results by at least 2.72% considering the second best performing method, BOCPD.
- **bee_waggle_6** – WATCH achieves the same performance as BOCPD and ZERO methods both in terms of F1 and Cover.
- **gas_sensor** – WATCH achieves a lower F1 performance than ECP and BOCPD. Considering that the WATCH behavior internally depends on a distance metric, we attribute this loss to a possible fractional distance issue depending on the large number of dimensions of this dataset (128). More specifically, if there are just few dimensions determining the change points, they may be harder to detect since the overall distance may not change significantly. However, it is noteworthy the performance in terms of F1 and Cover is still relatively close to the best performing methods: a 4.73% margin in terms of F1 with respect to ECP, which loses with most of the other datasets analyzed.
- **har** – WATCH significantly outperforms all other methods. Considering ECP, the second best performing method, WATCH improves its F1 score by 2.59%, and BOCPD by 14.64%. We consider this result as a representative case of success, considering both the large number of dimensions in this dataset (561) and the significant real-world implications, including human activity recognition and fall detection among others.
- **movement** – WATCH greatly outperforms other methods, with a 8.32% improvement in F1 score over BOCPD. Pairwise comparisons with other methods reveal even greater improvements.
- **mnist** – WATCH presents a huge margin of improvement when compared to other methods: at least 27.47% when compared to KCPA, the second best performing method.
- **occupancy** – WATCH outperforms ECP, the second best performing method, by 3.86% in terms of F1 score, achieving an almost perfect score (close to 1).

TABLE IV: F1 for multivariate datasets and methods. Values are unavailable either due to a failure (F) or the method timing out (T). Highest values for each time series are shown in bold.

Dataset	BOCPD	BOCPDMS	ECP	KCPA	RBOCPDMS	ZERO	WATCH
apple	0.916	0.445	0.745	0.634	F/T	0.594	**0.941**
bee_waggle_6	**0.929**	0.481	0.233	0.634	0.245	**0.929**	**0.929**
gas_sensor	0.619	T	**0.634**	0.444	F/T	0.091	0.604
har	0.792	F/T	0.885	0.831	F/T	0.148	**0.908**
mnist	0.533	T	0.625	0.706	F/T	0.167	**0.900**
movement	0.769	T	0.651	0.571	T	0.125	**0.833**
occupancy	0.919	0.735	0.932	0.812	F/T	0.341	**0.968**
run_log	**1.000**	0.469	0.990	0.909	0.380	0.446	0.671
traffic	0.703	0.500	0.451	0.414	0.296	0.083	**0.773**

TABLE V: Cover for multivariate datasets and methods. Values are unavailable either due to a failure (F) or the method timing out (T). Highest values for each time series are shown in bold.

Dataset	BOCPD	BOCPDMS	ECP	KCPA	RBOCPDMS	ZERO	WATCH
apple	**0.846**	0.720	0.758	0.462	F/T	0.425	0.748
bee_waggle_6	**0.891**	0.889	0.116	0.730	0.205	**0.891**	**0.891**
gas_sensor	0.742	T	**0.781**	0.435	F/T	0.094	0.743
har	0.732	F/T	**0.832**	0.729	F/T	0.084	0.802
mnist	0.464	T	0.611	**0.698**	F/T	0.123	0.697
movement	**0.766**	T	0.732	0.384	T	0.067	0.699
occupancy	0.645	0.556	0.666	0.581	F/T	0.236	**0.671**
run_log	**0.824**	0.327	0.819	0.729	0.329	0.304	0.603
traffic	0.546	0.357	0.387	0.332	0.158	0.073	**0.561**

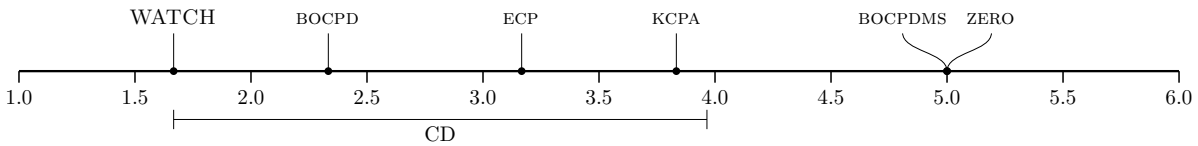


Fig. 2: F1 rank plot for multivariate data in the *best* mode.

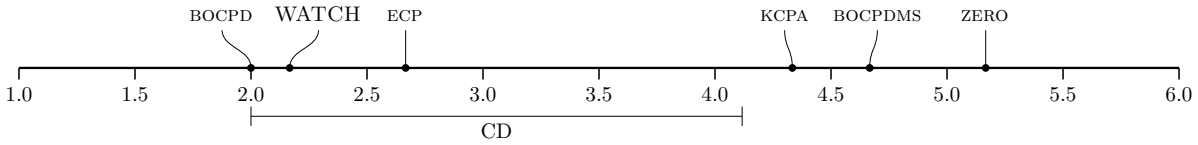


Fig. 3: Cover rank plot for multivariate data in the *best* mode.

- **run logs** - WATCH does not obtain competitive results on this dataset, presenting significantly worse results than the other methods. However, it is important to note that this dataset has only 2 dimensions. A quick inspection at the data shows that one dimension (overall *travel distance* of the runner) is monotonically increasing since it is accumulated over time, and therefore does not follow a typical data distribution. This data characteristic suggests that WATCH is unable to correctly detect changes due to the Wasserstein distance values being dominated by one of the two dimensions, yielding a large number of false positives. We argue that this dataset is atypical in nature, and is not representative of our main target, i.e. high-dimensional data. In this special case, the outcome may be completely different by just excluding the cumulative distance as a dimension, and limiting the analysis of the other dimension (*running pace*).
- **traffic** - WATCH significantly outperforms other methods, achieving an F1 score that is higher by 9.95% than the second-best performing method, BOCPD, and by 54.6% than the third-best performing method, BOCPDMS. We argue that this dataset is representative of our scope of analysis due to the strong real-world implication of traffic monitoring.

As we can see, the WATCH method achieves the best results with 7 out of 9 multivariate datasets. Most importantly, it provides consistent and robust detections with the datasets presenting the highest dimensionality (HAR and MNIST).

In order to assess the statistical significance of our results, we adopt a rank plot for both metrics (F1 and Cover) computed by averaging the ranks obtained by the algorithm over all datasets [52]. More specifically, Holm's procedure was adopted to calculate the significant differences considering all the different algorithms [53]. Figure 2 shows that WATCH strongly outperforms other methods in terms of F1. Looking at the figure, the pairwise difference between any two combina-

TABLE VI: F1 for univariate datasets (*best* mode). Values are unavailable either due to a failure (F), the method timing out (T) or missing values in the series unsupported by the method (M). Highest values for each time series are shown in bold.

Dataset	AMOC	BINSEG	BOCPD	BOCPDMS	CPNP	ECP	KCPA	PELT	PROPHET	RBOCPDMS	RFPOP	SEGNEIGH	WBS	ZERO	WATCH
bank	**1.000**	**1.000**	**1.000**	0.500	0.054	0.182	0.333	0.400	**1.000**	T	0.015	**1.000**	0.043	**1.000**	**1.000**
bitcoin	0.507	0.690	0.733	0.533	0.611	0.648	0.665	0.735	0.446	T	0.284	0.735	0.690	0.450	**0.806**
brent_spot	0.465	0.670	0.609	0.239	0.607	0.641	0.553	0.586	0.249	T	0.521	0.586	0.564	0.315	**0.677**
businv	**0.588**	**0.588**	**0.588**	0.455	0.386	0.370	0.294	0.490	0.275	0.370	0.261	**0.588**	0.289	**0.588**	**0.588**
centralia	0.909	**1.000**	**1.000**	**1.000**	**1.000**	0.909	**1.000**	**1.000**	0.763	0.846	**1.000**	**1.000**	0.556	0.763	0.974
children_per_woman	0.678	0.663	0.712	0.405	0.344	0.551	0.525	0.637	0.310	0.504	0.246	0.637	0.500	0.507	**0.844**
co2_canada	0.544	0.856	**0.924**	0.479	0.642	0.875	0.867	0.670	0.482	0.542	0.569	0.872	0.681	0.361	0.893
construction	0.696	0.709	0.709	0.410	0.602	0.709	0.634	0.709	0.324	0.340	0.185	0.709	0.523	0.696	**0.966**
debt_ireland	0.760	**1.000**	**1.000**	0.892	0.958	0.980	**1.000**	**1.000**	0.469	0.748	0.824	**1.000**	0.538	0.469	0.958
gdp_argentina	0.889	**0.947**	**0.947**	0.583	0.818	0.824	0.800	**0.947**	0.615	0.452	0.615	**0.947**	0.421	0.824	**0.947**
gdp_croatia	**1.000**	0.824	**1.000**	0.583	**1.000**	0.824	0.583	0.824	0.824	0.824	0.400	0.824	0.167	0.824	**1.000**
gdp_iran	0.696	0.652	0.862	0.492	0.620	0.824	0.734	0.808	0.652	0.737	0.636	0.808	0.576	0.652	**0.889**
gdp_japan	**1.000**	0.889	**1.000**	0.615	0.667	**1.000**	0.500	0.889	0.889	0.889	0.222	0.889	0.222	0.889	**1.000**
global_co2	0.929	0.929	0.889	0.458	0.667	**0.929**	0.667	0.929	0.463	0.547	0.293	0.929	0.250	0.846	0.846
homeruns	0.812	0.829	0.829	0.650	0.650	0.812	0.829	0.812	0.723	0.397	0.661	0.812	0.664	0.659	**0.974**
iceland_tourism	0.947	0.947	0.947	0.486	0.391	**1.000**	0.486	0.643	0.220	0.667	0.200	0.947	0.200	0.947	**1.000**
jfk_passengers	**0.776**	**0.776**	**0.776**	0.650	0.602	0.651	0.437	**0.776**	0.354	T	0.491	**0.776**	0.437	0.723	0.723
lga_passengers	0.561	0.620	0.704	0.563	0.606	**0.892**	0.526	0.537	0.366	T	0.592	0.537	0.674	0.535	0.774
measles	**0.947**	**0.947**	**0.947**	0.486	0.118	0.103	0.281	0.153	0.391	F/T	0.030	**0.947**	0.041	**0.947**	**0.947**
nile	**1.000**	**1.000**	**1.000**	0.800	**1.000**	**1.000**	0.824	**1.000**	0.824	0.667	**1.000**	**1.000**	**1.000**	0.824	**1.000**
ozone	0.776	0.723	0.857	0.778	0.750	**1.000**	0.667	**1.000**	0.723	0.651	0.429	**1.000**	0.286	0.723	**1.000**
quality_control_1	**1.000**	**1.000**	**1.000**	0.667	0.667	**1.000**	0.667	**1.000**	0.500	0.286	0.667	**1.000**	0.667	0.667	0.800
quality_control_2	**1.000**	**1.000**	**1.000**	0.667	**1.000**	**1.000**	**1.000**	**1.000**	0.750	0.429	**1.000**	**1.000**	**1.000**	0.750	**1.000**
quality_control_3	**1.000**	**1.000**	**1.000**	0.766	0.571	**1.000**	**1.000**	**1.000**	0.667	T	0.800	**1.000**	**1.000**	0.667	**1.000**
quality_control_4	0.810	0.873	0.787	0.561	0.658	0.787	0.658	0.780	0.780	T	0.241	0.780	0.608	0.780	**0.909**
quality_control_5	**1.000**	**1.000**	**1.000**	0.500	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.500	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
rail_lines	0.846	0.846	**0.966**	0.889	**0.966**	**0.966**	0.800	0.846	0.537	0.730	0.615	0.889	0.205	0.537	**0.966**
ratner_stock	0.776	0.824	**0.868**	0.559	0.396	0.776	0.754	0.824	0.280	T	0.203	0.824	0.378	0.571	0.824
robocalls	0.800	**0.966**	**0.966**	0.750	0.862	**0.966**	**0.966**	**0.966**	0.636	0.846	0.714	**0.966**	0.714	0.636	**0.966**
scanline_126007	0.710	0.920	**0.921**	0.829	0.906	0.870	0.838	0.889	0.644	T	0.649	0.889	0.818	0.644	0.833
scanline_42049	0.485	0.879	**0.962**	0.889	0.713	0.910	0.908	0.910	0.269	T	0.460	0.910	0.650	0.276	0.924
seatbelts	0.824	0.838	0.683	0.583	0.735	0.683	0.621	0.683	0.452	0.383	0.563	0.735	0.583	0.621	**0.974**
shanghai_license	**0.966**	0.868	0.868	0.605	0.600	0.868	0.465	0.868	0.532	0.389	0.357	0.868	0.385	0.636	0.778
uk_coal_employ	M	M	M	0.617	M	0.513	0.513	M	0.639	M	M	M	M	0.513	**0.941**
unemployment_nl	0.742	**0.889**	0.876	0.592	0.747	0.744	0.744	0.788	0.566	F/T	0.628	0.788	0.801	0.566	0.876
us_population	**1.000**	0.889	**1.000**	0.615	0.232	0.471	0.276	0.500	0.159	T	0.889	0.889	0.113	0.889	0.889
usd_isk	**0.785**	0.704	**0.785**	0.678	0.674	**0.785**	0.601	0.657	0.489	0.510	0.462	0.678	0.636	0.489	**0.785**
well_log	0.336	0.914	0.832	0.743	0.822	**0.928**	0.776	0.873	0.149	T	0.923	0.873	0.832	0.237	0.869

TABLE VII: Cover for univariate datasets (*best* mode). Values are unavailable either due to a failure (F), the method timing out (T) or missing values in the series unsupported by the method (M). Highest values for each time series are shown in bold.

Dataset	AMOC	BINSEG	BOCPD	BOCPDMS	CPNP	ECP	KCPA	PELT	PROPHET	RBOCPDMS	RFPOP	SEGNEIGH	WBS	ZERO	WATCH
bank	**1.000**	**1.000**	**1.000**	0.997	0.103	0.238	0.509	0.509	**1.000**	T	0.036	**1.000**	0.053	**1.000**	**1.000**
bitcoin	0.771	0.771	**0.822**	0.773	0.364	0.772	0.778	0.794	0.723	T	0.168	0.771	0.409	0.516	0.816
brent_spot	0.503	0.650	**0.667**	0.265	0.437	0.653	0.571	0.659	0.527	T	0.235	0.659	0.310	0.266	0.652
businv	0.574	0.562	**0.693**	0.459	0.402	0.690	0.405	0.647	0.539	0.459	0.123	0.647	0.154	0.461	0.554
centralia	0.675	0.675	**0.753**	0.650	0.675	**0.753**	0.675	0.675	0.675	0.624	0.568	0.675	0.253	0.675	**0.753**
children_per_woman	0.804	0.798	0.801	0.427	0.486	0.718	0.613	0.798	0.521	0.715	0.154	0.798	0.284	0.429	**0.868**
co2_canada	0.527	0.747	**0.773**	0.584	0.528	0.751	0.739	0.705	0.605	0.617	0.497	0.752	0.552	0.278	0.742
construction	0.629	0.575	0.585	0.571	0.375	0.524	0.395	0.423	0.561	0.575	0.154	0.575	0.300	0.575	**0.735**
debt_ireland	0.635	0.717	0.688	0.729	0.777	0.798	0.747	0.777	0.321	0.396	0.489	0.777	0.248	0.321	**0.832**
gdp_argentina	**0.737**	**0.737**	**0.737**	0.711	0.461	**0.737**	0.367	0.667	0.592	0.711	0.346	**0.737**	0.326	**0.737**	**0.737**
gdp_croatia	0.708	0.708	0.708	0.675	0.708	0.708	0.623	0.708	0.708	0.708	0.353	0.708	0.108	0.708	**0.773**
gdp_iran	0.583	0.583	0.583	0.611	0.448	0.583	0.505	0.505	0.583	0.692	0.248	0.583	0.295	0.583	**0.730**
gdp_japan	**0.802**	**0.802**	**0.802**	0.777	0.522	**0.802**	0.525	**0.802**	**0.802**	**0.802**	0.283	**0.802**	0.283	**0.802**	**0.802**
global_co2	**0.758**	**0.758**	**0.758**	0.745	0.381	0.665	0.602	0.743	0.284	0.745	0.368	**0.758**	0.338	**0.758**	**0.758**
homeruns	**0.694**	0.683	**0.694**	0.681	0.537	**0.694**	0.501	0.683	0.575	0.506	0.392	0.683	0.407	0.511	**0.694**
iceland_tourism	**0.946**	**0.946**	**0.946**	0.936	0.393	0.842	0.655	0.830	0.498	0.936	0.293	**0.946**	0.512	**0.946**	**0.946**
jfk_passengers	0.844	0.839	0.837	**0.855**	0.583	0.807	0.563	0.839	0.373	T	0.393	0.839	0.514	0.630	0.812
lga_passengers	0.434	0.541	0.547	0.475	0.478	**0.653**	0.536	0.534	0.446	T	0.426	0.543	0.501	0.383	0.604
measles	**0.951**	**0.951**	**0.951**	0.950	0.098	0.105	0.400	0.232	0.616	F/T	0.046	**0.951**	0.084	**0.951**	**0.951**
nile	**0.888**	0.880	**0.888**	0.870	0.880	**0.888**	0.758	0.880	0.758	0.876	0.880	0.880	0.880	0.758	**0.888**
ozone	0.721	0.600	0.602	0.735	0.458	0.676	0.451	0.635	0.574	0.755	0.358	0.627	0.309	0.574	**0.822**
quality_control_1	**0.996**	0.992	**0.996**	0.990	0.721	0.989	0.620	0.992	0.693	0.780	0.706	0.992	0.687	0.503	0.882
quality_control_2	0.927	0.922	**0.927**	0.921	0.922	0.927	**0.927**	0.922	0.723	0.637	0.922	0.922	0.922	0.638	0.923
quality_control_3	**0.997**	0.996	**0.997**	0.987	0.831	**0.997**	**0.997**	0.996	0.500	T	0.978	0.996	0.996	0.500	0.993
quality_control_4	**0.742**	0.673	0.673	0.670	0.508	0.540	0.535	0.673	0.673	T	0.077	0.673	0.506	0.673	0.673
quality_control_5	**1.000**	**1.000**	**1.000**	0.994	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.994	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
rail_lines	0.786	0.786	0.768	0.769	**0.786**	0.768	0.773	0.786	0.534	0.769	0.482	0.786	0.103	0.428	0.784
ratner_stock	0.874	0.914	0.906	0.872	0.382	0.874	0.771	**0.914**	0.444	T	0.162	**0.914**	0.182	0.450	0.906
robocalls	0.666	0.788	**0.808**	0.741	0.677	**0.808**	**0.808**	0.760	0.601	0.682	0.569	0.760	0.559	0.601	**0.808**
scanline_126007	0.634	0.633	0.631	**0.677**	0.433	0.390	0.494	0.444	0.503	T	0.316	0.633	0.329	0.503	0.523
scanline_42049	0.425	0.861	**0.892**	0.875	0.577	0.866	0.860	0.862	0.441	T	0.257	0.862	0.690	0.211	0.874
seatbelts	0.683	0.797	0.800	0.630	0.809	0.800	0.702	0.797	0.635	0.533	0.660	0.797	0.727	0.528	**0.832**
shanghai_license	**0.930**	0.920	0.920	0.911	0.474	0.920	0.497	0.920	0.804	0.763	0.381	0.920	0.351	0.547	0.847
uk_coal_employ	M	M	M	0.504	M	0.356	0.356	M	0.481	M	M	M	M	0.356	**0.650**
unemployment_nl	0.627	0.669	**0.669**	0.572	0.503	0.515	0.476	0.650	0.507	F/T	0.246	0.650	0.447	0.507	0.651
us_population	**0.803**	**0.803**	0.737	0.801	0.391	0.508	0.271	0.506	0.135	T	**0.803**	**0.803**	0.050	**0.803**	**0.803**
usd_isk	0.865	0.764	0.853	**0.866**	0.540	0.858	0.714	0.791	0.436	0.770	0.166	0.791	0.401	0.436	0.858
well_log	0.463	0.828	0.793	0.769	0.779	**0.835**	0.798	0.782	0.411	T	0.787	0.782	0.768	0.225	0.798

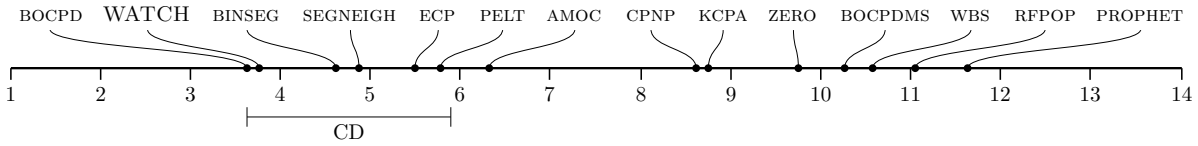


Fig. 4: F1 rank plot for univariate data in the *best* mode.

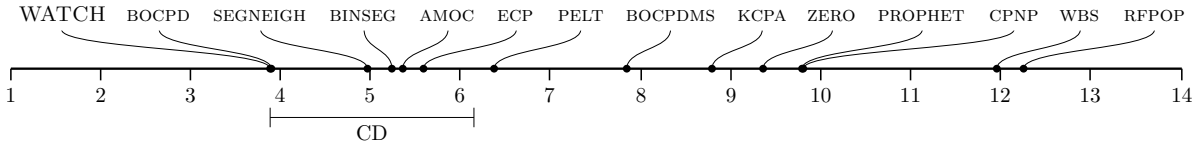


Fig. 5: Cover rank plot for univariate data in the *best* mode.

tions of algorithms is significant if their distance exceeds the critical difference (CD). In terms of Cover, looking at Figure 3 reveals a slightly lower performance for WATCH, which is not significantly worse than BOCPD. Moreover, it is noteworthy that WATCH still obtains the best rank values compared to all other algorithms considered in our study.

C. Univariate results analysis

Although this work focuses on multivariate data, it is noteworthy that WATCH presents a consistent performance also with the univariate datasets available in TCPD benchmark. Table VI and Table VII show the results in terms of F1 and Cover for all univariate datasets and algorithms. Overall, WATCH achieves the best F1 results with 23 out of 38 datasets. Considering the Cover metric, WATCH shows the highest value with 20 datasets. Moreover, Table III, shows that WATCH achieves the best average values of F1 and Cover for univariate datasets in the *best* experiment mode.

The results of WATCH for univariate datasets strongly suggest that the method may be successfully applied not only for multivariate (and particularly high-dimensional) but also for univariate time series datasets. The rank plots for univariate datasets reported in Figure 4 and 5 show that, overall, WATCH positions itself as the best (in terms of Cover) and second best (in terms of F1) method and significantly outperforms most of the other competitors. This result validates our considerations made inspecting the numerical values of the metrics in Tables VI and VII.

VI. Conclusion

In this work we addressed the change point detection problem focusing on multivariate and high-dimensional data, which represents a challenging scenario for most of state-of-the-art methods. We proposed WATCH, a novel Wasserstein distance-based change point detection approach that models the current data distribution and monitors its behavior over time. A threshold-ratio based approach is used to adaptively detect change points in dynamically changing time series data. An extensive experimental evaluation with a large number of benchmark datasets reveals the potential of our method, which outperforms all considered competitors in the majority of the cases. A statistical evaluation was performed to validate the obtained results. The results obtained validate our intuition that the Wasserstein distance and its powerful properties can be effectively applied for change point detection analysis tasks with high-dimensional data. As future work, we aim to investigate possible extensions of our method to leverage periodic behavior in time series data. Moreover, we will study procedures for automatic parameter adaptation, as well as possible applications in lifelong learning settings.

Acknowledgments

The work presented in this article was partially funded by DARPA through the project "Lifelong Streaming Anomaly Detection" (Grant N. A19-0131-003 and A21-0113-002). We also acknowledge the support of NVIDIA through the donation of a Titan V GPU.

References

[1] M. I. Baron, "Nonparametric adaptive change point estimation and on line detection," *Sequential Analysis*, vol. 19, no. 1-2, pp. 1–23, 2000.

[2] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems*, vol. 51, no. 2, pp. 339–367, 2017.

[3] J. Gama, *Knowledge discovery from data streams*. CRC Press, 2010.

[4] R. Corizzo, M. Ceci, G. Pio, P. Mignone, and N. Japkowicz, "Spatially-aware autoencoders for detecting contextual anomalies in geo-distributed data," in *Discovery Science*. Cham: Springer International Publishing, 2021, pp. 461–471.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[6] K. Faber, L. Faber, and B. Sniezynski, "Autoencoder-based ids for cloud and mobile devices," in *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE, 2021, pp. 728–736.

[7] S. Ryan, R. Corizzo, I. Kiringa, and N. Japkowicz, "Pattern and anomaly localization in complex and dynamic data," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, 2019, pp. 1756–1763.

[8] B. Krawczyk, B. Pfahringer, and M. Woźniak, "Combining active learning with concept drift detection for data stream mining," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 2239–2244.

[9] V. M. Souza, F. A. Chowdhury, and A. Mueen, "Unsupervised drift detection on high-speed data streams," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 102–111.

[10] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 964–994, 2016.

[11] S. Wares, J. P. Isaacs, and E. Elyan, "Data stream mining: methods and challenges for handling concept drift," *SN Applied Sciences*, 2019.

[12] A. Essien, I. Petrounias, P. Sampaio, and S. Sampaio, "A deep-learning model for urban traffic flow prediction with traffic events mined from twitter," *World Wide Web*, vol. 24, no. 4, pp. 1345–1368, 2021.

[13] J.-L. Reyes-Ortiz, L. Oneto, A. Sama, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, vol. 171, pp. 754–767, 2016.

[14] M. Ceci, R. Corizzo, F. Fumarola, M. Ianni, D. Malerba, G. Maria, E. Masciari, M. Oliverio, and A. Rashkovska, "Big data techniques for supporting accurate predictions of energy production from renewable sources," in *Proceedings of the 19th International Database Engineering & Applications Symposium*, 2015, pp. 62–71.

[15] G. Habault, Y. Nishimura, K. Yoshihara, and C. Ono, "Detecting errors in short-term electricity demand forecast using people dynamics," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 4163–4168.

[16] M. Ceci, R. Corizzo, N. Japkowicz, P. Mignone, and G. Pio, "Echad: embedding-based change detection from multivariate time series in smart grids," *IEEE Access*, vol. 8, pp. 156 053–156 066, 2020.

[17] L. Zhu, X. Su, and X. Tai, "A high-dimensional indexing model for multi-source remote sensing big data," *Remote Sensing*, vol. 13, no. 7, p. 1314, 2021.

[18] R. Corizzo, M. Ceci, and N. Japkowicz, "Anomaly detection and repair for accurate predictions in geo-distributed big data," *Big Data Research*, vol. 16, pp. 18–35, 2019.

[19] K. Prabuchandran, N. Singh, P. Dayama, A. Agarwal, and V. Pandit, "Change point detection for compositional multivariate data," *Applied Intelligence*, pp. 1–26, 2021.

[20] D. V. Hinkley, "Inference about the change-point in a sequence of random variables," 1970.

[21] Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 2, pp. 114–127, 2012.

[22] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application*. prentice Hall Englewood Cliffs, 1993, vol. 104.

[23] S. V. Georgakopoulos, S. K. Tasoulis, and V. P. Plagianakos, "Efficient change detection for high dimensional data streams," in *2015 IEEE International Conference on Big Data (Big Data)*, 2015, pp. 2219–2222.

[24] A. J. Scott and M. Knott, "A cluster analysis method for grouping means in the analysis of variance," *Biometrics*, pp. 507–512, 1974.

[25] I. E. Auger and C. E. Lawrence, "Algorithms for the optimal identification of segment neighborhoods," *Bulletin of mathematical biology*, vol. 51, no. 1, pp. 39–54, 1989.

[26] P. Fryzlewicz, "Wild binary segmentation for multiple change-point detection," *The Annals of Statistics*, vol. 42, no. 6, pp. 2243–2281, 2014.

[27] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal detection of changepoints with a linear computational cost," *Journal of the American Statistical Association*, vol. 107, no. 500, pp. 1590–1598, 2012.

[28] P. Fearnhead and G. Rigaill, "Changepoint detection in the presence of outliers," *Journal of the American Statistical Association*, vol. 114, no. 525, pp. 169–183, 2019.

[29] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[30] R. P. Adams and D. J. MacKay, "Bayesian online changepoint detection," *arXiv preprint arXiv:0710.3742*, 2007.

[31] J. Knoblauch, J. E. Jewson, and T. Damoulas, "Doubly robust bayesian inference for non-stationary streaming data with beta-divergences," *Advances in Neural Information Processing Systems*, vol. 31, pp. 64–75, 2018.

[32] J. Knoblauch and T. Damoulas, "Spatio-temporal bayesian on-line changepoint detection with model selection," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2718–2727.

[33] V. Chandola and R. R. Vatsavai, "Scalable time series change detection for biomass monitoring using gaussian process." in *CIDU*, 2010, pp. 69–82.

[34] S. Brahim-Belhouari and A. Bermak, "Gaussian process for nonstationary time series prediction," *Computational Statistics & Data Analysis*, vol. 47, no. 4, pp. 705–712, 2004.

[35] K. Haynes, P. Fearnhead, and I. A. Eckley, "A computationally efficient nonparametric approach for changepoint detection," *Statistics and Computing*, vol. 27, no. 5, pp. 1293–1305, 2017.

[36] Z. Harchaoui, E. Moulines, and F. R. Bach, "Kernel change-point analysis," in *Advances in neural information processing systems*, 2009, pp. 609–616.

[37] D. S. Matteson and N. A. James, "A nonparametric approach for multiple change point analysis of multivariate data," *Journal of the American Statistical Association*, vol. 109, no. 505, pp. 334–345, 2014.

[38] H. Du and Z. Duan, "Finder: A novel approach of change point detection for multivariate time series," *Applied Intelligence*, 06 2021.

[39] B. Gaujac, I. Feige, and D. Barber, "Learning disentangled representations with the wasserstein autoencoder," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 69–84.

[40] M. Hallin, G. Mordant, and J. Segers, "Multivariate goodness-of-fit tests based on wasserstein distance," 2021.

[41] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schölkopf, "Wasserstein auto-encoders," in *6th International Conference on Learning Representations (ICLR 2018)*. OpenReview. net, 2018.

[42] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multidimensional data streams," *Interfaces*, 01 2006.

[43] S. Aminikhanghahi, T. Wang, and D. J. Cook, "Real-time change point detection with application to smart home time series data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, pp. 1010–1023, 2019.

[44] G. J. J. van den Burg and C. K. I. Williams, "An evaluation of change point detection algorithms," *arXiv preprint arXiv:2003.06222*, 2020.

[45] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *ESANN*, 2013.

[46] "Gas Sensor Array Drift Dataset at Different Concentrations," UCI Machine Learning Repository, 2013.

[47] Dias, Daniel, Peres, Sarajane, Bscaro, and Helton, "Libras Movement," UCI Machine Learning Repository, 2009.

[48] "The MNIST database of handwritten digits," Online: http://yann.lecun.com/exdb/mnist/.

[49] "Behavior of the urban traffic of the city of Sao Paulo in Brazil," UCI Machine Learning Repository, 2018.

[50] D. Martin, C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 530–549, 2004.

[51] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[52] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[53] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian journal of statistics*, pp. 65–70, 1979.

Chapter 13

LIFEWATCH: Lifelong Wasserstein Change Point Detection

In this paper, we addressed a previously unexplored problem of lifelong change point detection. Our proposed method, LIFEWATCH, is a Wasserstein distance-based change point detection method. In addition to change point detection, LIFEWATCH has the capability to recognize recurring tasks, as it is extended with a memory able to model multiple data distributions in a fully unsupervised manner. The method can recognize whether a change means the appearance of a new task or the recurrence of the already encountered task.

Our experiments carried out on a large number of benchmark datasets revealed that LIFEWATCH outperforms state-of-the-art change point detection competitors. Moreover, we performed additional experiments to showcase LIFEWATCH potential in recognizing the characteristic of detected change.

Contributions of Kamil Faber: In this work, Kamil Faber co-designed and developed the LIFEWATCH method. Kamil Faber carried out the experiments and was co-author of all sections of the paper.

This paper was published at the 2022 International Joint Conference on Neural Networks (IJCNN; CORE B-Rank; 140 points).

LIFEWATCH: Lifelong Wasserstein Change Point Detection

Kamil Faber
AGH University of Science and Technology
Institute of Computer Science
Krakow, Poland
kfaber@agh.edu.pl

Roberto Corizzo
Department of Computer Science
American University
Washington, DC, USA
rcorizzo@american.edu

Bartlomiej Sniezynski
AGH University of Science and Technology
Institute of Computer Science
Krakow, Poland
bartlomiej.sniezynski@agh.edu.pl

Michael Baron
Department of Mathematics and Statistics
American University
Washington, DC, USA
baron@american.edu

Nathalie Japkowicz
Department of Computer Science
American University
Washington, DC, USA
japkowic@american.edu

2022 International Joint Conference on Neural Networks (IJCNN) | 978-1-7281-8671-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IJCNN55064.2022.9892891

***Abstract*—Change point detection methods offer a crucial capability in modern data analysis tasks characterized by evolving time series data in the form of data streams. Recent interest in lifelong learning showed the importance of acquiring knowledge and identifying new occurring tasks in a continually evolving environment. Although this setting could benefit from a timely identification of changes, existing change point detection methods are unable to recognize recurring tasks, which is a necessary condition in lifelong learning. In this paper, we attempt to fill this gap by proposing `LIFEWATCH`, a novel Wasserstein-based change point detection approach with memory capable of modeling multiple data distributions in a fully unsupervised manner. Our method does not only detect changes, but discriminates between changes characterized by the appearance of a new task and changes that rather describe a recurring or previously seen task. An extensive experimental evaluation involving a large number of benchmark datasets shows that `LIFEWATCH` outperforms state-of-the-art methods for change detection while exploiting the characterization of detected changes to correctly identify tasks occurring in complex scenarios characterized by recurrence in lifelong consolidation settings.**

***Index Terms*—Lifelong Learning, Change Point Detection, Time-Series Data**

I. Introduction

Change point detection methods provide a significant contribution for the analysis of data characterized by a dynamic and time-evolving nature. The identification of time points that define when the data distribution changes is, in fact, crucial in a variety of real-world domains characterized by multivariate time series in the form of data streams [1]. Applications include human activity recognition [2], renewable energy plant monitoring in smart grids [3], anomaly detection in geo-distributed data [4], traffic flow prediction [5], and intrusion detection [6].

Change point detection methods are typically incorporated in more complex machine learning based approaches to inform changes in the data distribution and trigger appropriate actions. In particular, lifelong learning methods are recently trying to address the challenge of acquiring new knowledge and tackling new tasks (corresponding to emerging classes, concepts, or distributions) in a continuously evolving environment. Recent methods that address this new learning paradigm are attracting increasing interest, although they are limited to image classification tasks [7]–[11]. In lifelong learning, change point detection can be used as a tool to signal changes between tasks and inform the consolidation of new tasks in memory-based approaches or, more generally, trigger updates of deep neural network-based models. However, current change point detection approaches in the literature are still not particularly suited to the needs of lifelong learning. Limitations include the inability to differentiate between changes that anticipate a new task (i.e. that never appeared before) and changes that anticipate a recurring task (i.e. the appearance of a previously seen task). As a consequence, this limitation may affect lifelong learning methods in terms of robustness and accuracy.

In this paper, we attempt to fill this gap by proposing `LIFEWATCH`, a novel change point detection approach that leverages memory to provide consolidation capabilities, particularly suited for the analysis of multivariate time series data in lifelong learning settings. Our method is able to efficiently identify changes in a multivariate time series in a fully unsupervised way while discriminating between changes characterized by the appearance of a new task and changes that rather describe a recurring or previously seen task. An extensive experimental evaluation on real-world multivariate time series datasets is conducted. The results of traditional change point detection, as well as lifelong consolidation, are presented and show that `LIFEWATCH` outperforms state-of-the-art methods for change point detection, conventional clustering, and incremental clustering in challenging experimental conditions.

The remainder of the paper is structured as follows. Section II discusses relevant change point detection works in the state-of-the-art. Section III describes the proposed method. Section

978-1-7281-8671-9/22/$31.00 ©2022 IEEE

IV defines the research questions and provides details on our experiments and their setup, followed by a discussion of the results in Section V. Finally, Section VI concludes the paper, summarizing our contributions and discussing possible future work.

II. Related Work

Change point detection methods can be characterized in three main types: Offline, Bayesian, and Non-Parametric [12].

Offline methods often incorporate likelihood ratio approaches that compare probability distributions of data before and after a candidate change point. The At Most One Change (AMOC) method [13] generalized this idea. Generally, these methods monitor the logarithm of the likelihood ratios within two consecutive intervals [14], [15]. Some variants introduce reducing data dimensionality as a preliminary stage [16]. As pointed out in [12], the main downside of these approaches is their reliance on known parametric models, which limits their flexibility in real-world domains. Some offline methods such as Binary Segmentation (BINSEG) [17] and Segment Neighborhoods (SEGNEIGH) [18] are capable of discovering more than one change point in a time series, by means of a greedy binary segmentation approach relying on a cost function. One alternative approach for binary segmentation is known as Wild Binary Segmentation and leverages the CUSUM statistic [19] (WBS). Additional state-of-the-art methods in this category include extensions that include pruning algorithms, resulting in a reduced execution time, such as Pruned Exact Linear Time [20] (PELT) and Robust Functional Prunning Optimal Partitioning (RFPOP) [21], which presents an higher robustness to outliers. Unfortunately, these methods are not capable of analyzing multivariate data.

Offline and online Bayesian approaches are included in probabilistic methods. Prophet [22] is a Bayesian forecasting method that supports time series with change points. Bayesian Online Change Point Detection [23] is another online approach that is based on two assumptions: *i)* a time series can be partitioned into non-overlapping states; *ii)* data points in each state are distributed independently and identically. Bayesian methods compute the run length distribution and update the relevant statistics using Bayes' theorem. Finally, the detection of change points is based on the comparison of probability values. Model Selection (BOCPDMS) [24] and spatio-temporal models (RBOCPDMS) [25] have been introduced in recent extensions.

Non-parametric change-point detection methods are usually based on tests of hypothesis, leveraging a test statistic, and a pre-defined threshold. Notable methods pertaining to this class include Kernel Change-Point Analysis (KCPA) [26], Nonparametric Change Point Detection (CPNP) [27], and Energy Change Point (ECP) [28]. Alternative non-parametric methods are distance-based. One representative example is the WATCH method [29] which exploits the Wasserstein distance metric. Instead of comparing two consecutive intervals, the method compares incoming data points to the stored part of the current distribution and decides whether they are from the same distribution.

III. Method

In this section, we describe LIFEWATCH, our change point detection method designed to work in lifelong settings. Building upon our preliminary efforts in [29], the method leverages the Wasserstein distance as a non-parametric distance-based approach to compare data distributions. In the following, we first introduce the Wasserstein distance, which is a building block of the method. After that, we provide a detailed algorithm and description of the method.

A. Wasserstein distance

The Wasserstein distance is a metric defined by a probability distribution on a given metric space M. The literature provides an intuitive description of the metric considering the distribution as a unit amount of soil piled on M and the metric as the minimum "cost" of turning one pile into the other. Precisely, the metric value is considered the amount of earth to move multiplied by the mean distance it has to be moved. This analogy explains why the metric is known as the earth mover's distance.

Following the definition of the Wasserstein distance from [30], let $P(\mathbb{R}^d)$ be the set of Borel probability measures on $\mathbb{R}^d$ and $P_p(\mathbb{R}^d)$ be the subset of such measures with a finite moment of order $p \in [1, \inf)$. For P, Q $\in P(\mathbb{R}^d)$, let $\Gamma(\mathrm{P}, \mathrm{Q})$ be the set of probability measures γ on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals P and Q, i.e., such that $\gamma(B \times \mathbb{R}^d) = \mathrm{P}(B)$ and $\gamma(\mathbb{R}^d \times B) = \mathrm{Q}(B)$ for Borel sets $B \subseteq \mathbb{R}^d$. $dist()$ is the Euclidean norm. The p-Wasserstein distance between P, Q $\in P_p(\mathbb{R}^d)$ can be defined as:

$$W_p(\mathrm{P}, \mathrm{Q}) = \Big(\inf_{\gamma \in \Gamma(P,Q)} \int_{\mathbb{R}^d \times \mathbb{R}^d} dist(x-y)^p \; \partial\gamma(x,y)\Big)^{\frac{1}{p}}. \quad (1)$$

In terms of random variables X and Y with laws P and Q, respectively, the p-Wasserstein distance is the smallest value of $\{E(dist(X-Y))^p\}^{\frac{1}{P}}$ over all possible joint distributions $\gamma \in \Gamma(\mathrm{P}, \mathrm{Q})$ of (X, Y). It is important to note that the Wasserstein distance computation is based on an approximated implementation, as the original formulation is computationally intractable. From an intuitive point of view, having a vector space $V \subset \mathbb{R}^k$, where k corresponds to the number of dimensions, it is possible to map the distributions (P, Q) to corresponding sets of k-dimensional data points $G, H \subseteq V$. In this paper, we leverage the SAG approximated computational approach to calculate the Wasserstein distance W_P as in [30]. The approximated version supports the analysis of data points rather than probability distributions that are computationally harder to handle. Consequently, in Algorithm 1, we refer to distributions as sets of data points representing their probability distribution. Another advantage of the approximated implementation is that it allows for a time-efficient distance computation on high-dimensional data. In the following text, we use $W_A \approx W_p$ to refer to the approximated version of the Wasserstein distance.

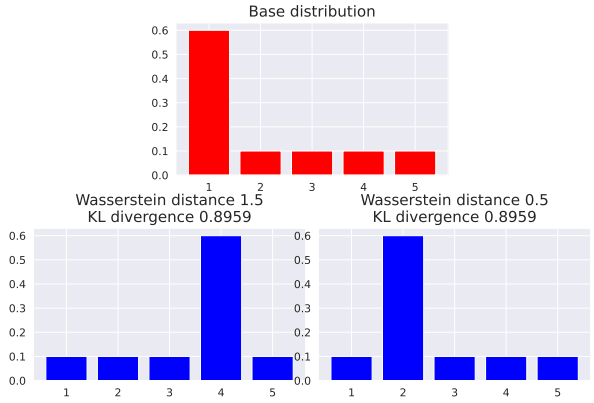


Fig. 1. Visual comparison of Wasserstein distance and Kullbach-Leibler (KL) divergence. A pairwise comparison between the base distribution (in red) and two target distributions (in blue) reveals that the KL divergence estimates the same distance between the red and blue distributions. On the other hand, the Wasserstein distance reflects a better ability to measure the actual distance between the base and the two target distributions. As a result, it is able to capture a difference between the two target distributions that is uncovered by the KL divergence.

The Wasserstein distance has some unique advantages that justifies its adoption in our method. Measuring the work required to transport the probability mass between distribution states provides a meaningful and smooth representation of the distance between distributions, especially useful in domains where evaluating outcome similarity is more important than perfect likelihood matching. Figure 1 intuitively presents the informative power carried by the Wasserstein distance when compared to the Kullback-Leibler (KL) divergence. The Wasserstein distance, which measures the work required to convey the probability mass from one distribution state to the other, recognizes the difference between the red and blue distributions even though their KL divergence is the same. Motivated by these desirable properties, our method adopts the Wasserstein distance aiming at triggering a more precise and robust detection of change points. This assumption may be especially relevant in multivariate data streams, in which high-dimensionality presents increased challenges.

B. Algorithm

The input for LIFEWATCH is a time series of d-dimensional data samples $\mathbf{I} = (I_1, I_2, \ldots, I_T)$. LIFEWATCH learns an initial representation of an unknown distribution D_C in an unsupervised manner. It processes the multivariate time series data stream in mini-batches B_i of length ω (sliding non-overlapping windows). Data points collected over time for the current distribution D_C are stored in a buffer. When the number of samples in the buffer reaches the desired capacity (κ parameter), D_C is added to the distribution pool P, which represents the memory of the LIFEWATCH method. The size of P can be customized to accommodate limited memory requirements. Moreover, the condition that triggers the addition of a new distribution to P can be easily controlled to reflect domain characteristics, such as task volatility, by tuning the ϵ parameter. All distributions added to P are subsequently exploited in the change detection process. Each distribution D_j in the pool P has an assigned threshold $E[D_j]$, which is periodically updated to reflect the characteristics of the distribution. The threshold is determined as:

$$E[D_j] = \epsilon \max_{B_i \in D_j} W_A(B_i, D_j). \quad (2)$$

Intuitively, each updated threshold $E[D_j]$ conservatively reflects the largest distance computed across all mini-batches for a given distribution D_j, scaled by an ϵ factor which allows to include data points that are reasonably beyond the core of the current distribution. For each mini-batch B_i, the method determines which is the target distribution assigned. The change point detection process starts with checking whether B_i can be qualified as belonging to the current distribution D_C, comparing the Wasserstein distance between B_i and D_C with the desired threshold $E[D_C]$ for the current distribution. If the distance is lower than $E[D_C]$, we qualify B_i as belonging to D_C. On the contrary, if the distance is higher than $E[D_C]$, a change point is detected and the nature of the change point is determined. LIFEWATCH maintains two result sets for detected change points, characterizing them in new task changes (N) and recurring task changes (R).

More specifically, in order to determine whether B_i belongs to any already known distribution $D_j \neq D_C$, corresponding to a recurring task, LIFEWATCH determines the closest distribution D_k by looking for the lowest ratio between the Wasserstein distance $W_A(B_i, D_j)$ and the specific threshold for this distribution $E[D_j]$. If the distance to the closest distribution D_k is lower then $E[D_k]$, then it qualifies B_i as belonging to D_k and shifts the current distribution, i.e. D_C becomes D_k. It also adds a change point to a set R of change points for recurring tasks. On the contrary, if the distance to the closest distribution D_k is higher than $E[D_k]$, a new distribution is created: $D_C \leftarrow B_i$ and the change point is added to a set N of change points for new tasks.

After determining the proper distribution for B_i, LIFEWATCH adds B_i to D_C to let the distribution reflect the new data points. Simultaneously, it updates the corresponding threshold $E[D_C]$. The data samples for given distribution are accommodated until D_C reaches its storage capacity controlled by μ. This choice implies that the method maintains a compact version of the data stream, resulting in a lightweight memory footprint. A pseudo-code of the algorithm is presented in Algorithm 1. We emphasize some significant advantages of LIFEWATCH over existing change point detection methods:

- It works by comparing sets of points instead of working with the parameters of specific distributions, which would imply making assumptions on the distribution type;
- It characterizes change points in two types: new tasks, and recurring tasks;
- It presents the advantage of memory approaches, i.e. the opportunity to recall prior experiences, while preserving

Input: $\mathbf{I} = (I_1, I_2, \ldots, I_T)$ Time series data

Result: N Change points (New Tasks),
R Change points (Recurring Tasks).

Parameters: κ Min points required in a distrib. before starting detection,
μ Max points stored for a distribution,
ϵ Threshold ratio for the distance value,
ω Mini-batch size.

Initialize pool of distributions $P \leftarrow \emptyset$
Initialize pool of distribution thresholds $E \leftarrow \emptyset$
Initialize current distribution $D_C \leftarrow \emptyset$
Initialize results variables: $N \leftarrow \emptyset, \quad R \leftarrow \emptyset$
$\mathbf{B} \leftarrow$ Process $\mathbf{I}$ into sequence $(B_1, B_2, \ldots, B_{\frac{T}{\omega}})$ of non-overlapping mini-batches B_i of size ω
for $B_i \in \mathbf{B}$ **do**
  if $|D_C| < \kappa$ **then**
    $D_C \leftarrow D_C \cup B_i$
    if $|D_C| >= \kappa$ **then**
      Determine $E[D_C]$ from Eq. (2) with ϵ ▷ Save threshold for D_C
      $P \leftarrow P \cup D_C$ ▷ Add D_C to the distribution pool
    end
  else
    $\upsilon \leftarrow W_A(B_i, D)$ ▷ Compute Wasserstein Distance
    if $\upsilon > E[D_C]$ **then**
      $D_k = \begin{cases} \arg\min_{D_j \in P} \frac{E[D_j]}{W_A(B_i, D_j)}, & \text{if } \exists_{D_j \in P} W_A(B_i, D_j) < E[D_j] \\ \emptyset, & \text{otherwise} \end{cases}$
      Set change point $c \leftarrow i * \omega$
      if $|D_k| = 0$ **then**
        $N \leftarrow N \cup \{c\}$ ▷ Change point (New task)
      else
        $R \leftarrow R \cup \{c\}$ ▷ Change point (Recurring task)
      end
      $D_C \leftarrow D_k$ ▷ Set D_k as a current distribution
    end
    if $|D_C| < \mu$ **then**
      $D_C \leftarrow D_C \cup B_i$
      Determine $E[D_C]$ from Eq. 2 with ϵ ▷ Update threshold for D_C
    end
  end
end
return N, R

Algorithm 1: Pseudo-code of the proposed LIFEWATCH method

a lightweight memory footprint;
- It automatically updates thresholds for each learned distribution without human intervention;
- It avoids unnecessary comparisons of new data points with distributions in the pool if they are recognized as belonging to the current distribution.

IV. Experiments

The experimental study is designed to address the following research questions:

- **RQ1:** Does LIFEWATCH offer accurate and precise detection of change points on a large number of benchmark datasets and reference competitor methods?
- **RQ2**: Can the LIFEWATCH characterization of change points be effectively leveraged in a lifelong consolidation setting to accurately identify the number of tasks in scenarios with recurrence?
- **RQ3**: Is LIFEWATCH competitive when compared to a heterogeneous set of methods for change point detection, conventional clustering, and stream clustering methods in the lifelong consolidation setting?

In the following, we describe our datasets, scenarios, and metrics. Subsequently, we discuss the results obtained in our experiments.

A. Datasets

In addition to a large number of univariate and multivariate datasets (42) in the change point detection task [31], we adopted the following multivariate time series datasets to evaluate the methods with challenging data representing real-world scenarios for change point detection and lifelong consolidation tasks:

1) **HAR** [32] (561 features): Human Activity Recognition dataset containing sequences of 6 types of human activities such as walking, sitting, standing, and laying. In our experiments, we analyze time series data from all human participants separately and report the average values of the metrics.
2) **Movement** [33] (90 features): The Libras movement dataset contains sequences of 15 hand movement types in the official Brazilian signal language.
3) **Traffic** [34] (17 features): This dataset contains sequential urban traffic measurements. We obtain a discrete attribute with 3 classes (*low*, *medium*, and *high*) by performing discretization on the *slowness* attribute.
4) **Eye** [35] (14 features): Dataset containing records of EEG measurements while observing whether the eye of the subject is opened or closed.

B. Lifelong learning scenarios

In our experiments, we are interested in reproducing a lifelong learning scenario with newly appearing tasks over time. For this reason, we adopt time series datasets presented as a stream of mini-batches. We enumerate the set of possible tasks (corresponding to the k distinct classes in each dataset) and we generate 10 random permutations of them to extract a sequential scenario, whenever possible. As for the **Traffic** and **Eye** datasets (6 and 2 classes, respectively), we preserve the full set of possible task permutations. In order to further increase the complexity and realism of the scenario, we generate a sequence of tasks with a three-time recurrence, resulting in sequential scenarios of $(k \times 3)$ tasks. We argue that recurrence increases the difficulty of the experimental conditions for each considered method, which has to determine whether a change in the data distribution corresponds to a new, never seen before task, or to an already seen task. As a result,

an incorrectly identified type of change may result in wrong inferences for new data points.

C. Metrics

Our evaluation is divided in two parts: change point detection and lifelong consolidation. Each part presents different metrics described in the following:

- **Change Point Detection** – metrics adopted in the TCPD Benchmark: [31]
  - **F1**: Harmonic mean of Precision and Recall. Recall describes the ability of the model to find all relevant change points, while Precision measures how well the model can identify only relevant change points.
  - **Cover**: Describes how well the ground truth segments are covered by the predicted segments. In the time series case, a segment corresponds to a specific subsequence of contiguous points. The details on Cover metric can be found in [31].
- **Lifelong consolidation:**[1]
  - **Number of Clusters:** Number of identified tasks.
  - **Homogeneity**: Assesses cluster labeling given ground truth. It has a perfect score (1.0) when each cluster has data points belonging to the same task.
  - **Completeness**: Proportion of points from each task assigned to the same cluster. It has perfect score (1.0) when all data points from the same task are into the same cluster.
  - **V-measure**: Harmonic mean of homogeneity and completeness.

V. Results and discussion

In this section, we describe the results obtained in two tasks: change point detection and lifelong consolidation.

A. Change point detection

To evaluate change point detection in non-recurring scenarios, we adopt the Turing Change Point Detection (TCPD) Benchmark [31]. It allows the evaluation of any new change point detection method on 42 datasets and compares its results to 13 pre-implemented algorithms. We extend the benchmark including the LIFEWATCH method and the datasets mentioned in Section IV-A. The evaluation is based on F1 and Cover metrics, described in Section IV-C. Results are reported in Table I, where each method is optimized via grid search using benchmark sets of parameter values, and the best results are reported on each line. The results show that our LIFEWATCH method achieves the best results in terms of average Cover and F1 metrics, in both univariate and multivariate datasets, also when compared to state-of-the-art algorithms such as Bayesian Online Change Point Detection (BOCPD) [23] and Energy Change Point (ECP) [28]. We experimentally observed that small values for the batch size parameter (e.g. 2, 4, 8) yield the best results in terms of F1 and Cover on all datasets analyzed. This generalized behavior demonstrates the stability and accuracy of the method when the data granularity is set to a few data points. In this setting, LIFEWATCH is able to detect changes in a timely manner while avoiding false alarms. A purely incremental setting (one sample at a time) was shown to be too sensitive to false positives, whereas a large batch size (higher powers of 2) was shown to yield false negatives. Overall, these results confirm the potential of our method in a typical change point detection task, in which we are broadly interested in identifying changes, without worrying about their characterization (**RQ1**).

TABLE I
CHANGE POINT DETECTION: AVERAGE RESULTS IN TERMS OF COVER AND F1 METRICS. BLANK VALUES INDICATE THAT THE METHOD DOES NOT SUPPORT MULTIVARIATE DATA. RESULTS FOR THE BEST PERFORMING METHOD ARE MARKED IN BOLD.

	Univariate		Multivariate	
	Cover	F1	Cover	F1
AMOC	0.726	0.778		
BINSEG	0.760	0.833		
BOCPD	0.769	0.857	0.705	0.785
BOCPDMS	0.737	0.620	0.285	0.263
CPNP	0.538	0.648		
ECP	0.711	0.789	0.616	0.665
KCPA	0.619	0.679	0.583	0.671
PELT	0.706	0.766		
PROPHET	0.574	0.537		
RBOCPDMS	0.422	0.349	0.069	0.092
RFPOP	0.403	0.517		
SEGNEIGH	0.763	0.832		
WBS	0.417	0.519		
ZERO	0.573	0.658	0.268	0.332
LIFEWATCH	**0.798**	**0.908**	**0.713**	**0.800**

We assess the statistical significance of the results by adopting a rank plot for the F1 metric computed by averaging the ranks obtained by the algorithm over all datasets [36]. We apply Holm's procedure to calculate the significant differences considering all the algorithms [37]. Figure 2 shows that LIFEWATCH strongly outperforms other methods except for BOCPD, which obtains the same rank value for multivariate datasets. If the distance between two algorithms exceeds the critical difference (CD), then the difference between them is significant. It looks clear that, from a statistical viewpoint, LIFEWATCH is on par with BOCPD, even if LIFEWATCH obtains the best average results in terms of Cover and F1 metrics. Figure 3 presents the same comparison but for univariate datasets, showing that LIFEWATCH achieves the best rank across all univariate datasets.

We compare the reported results with a variant of the method that leverages the more popular KL-divergence. As expected, the extracted results appear much worse than those extracted using the Wasserstein distance: 0.510 Cover and 0.568 F1 for univariate datasets; 0.356 Cover and 0.432 F1 for multivariate datasets. Using the Wasserstein distance yields an improvement of 0.288 in terms of Cover and 0.340 in terms of F1 for univariate datasets. Moreover, it yields an improvement

[1]We adopted clustering metrics due to the unsupervised and rather model-less nature of our method, which does not allow us to calculate metrics such as *Forward Transfer* adopted in the lifelong learning.

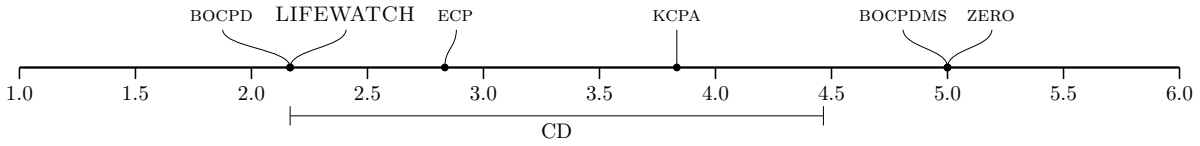


Fig. 2. Change point detection: F1 rank plot for all multivariate datasets analyzed.

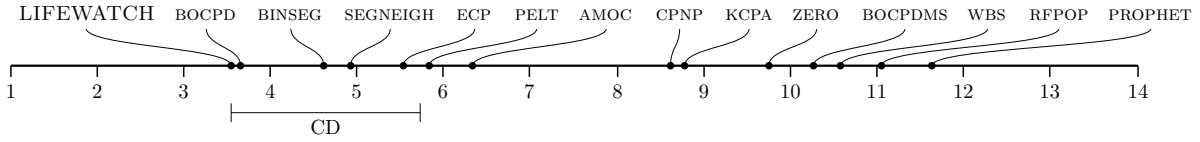


Fig. 3. Change point detection: F1 rank plot for all univariate datasets analyzed.

of 0.357 in terms of Cover and 0.232 in terms of F1 for multivariate datasets. This evaluation confirms our hypothesis about the Wasserstein distance being a more suitable and powerful metric for change point detection in the experimental setting analyzed in this study.

B. Lifelong consolidation

These experiments are focused on assessing the potential of our method in correctly identifying tasks in a consolidation experiment, leveraging its ability to differentiate between two types of change points as a feedback in a lifelong learning setting. We compare our results with three classes of methods:

- **Change point detectors**: We devise a simple procedure to create a clustering model, in order to compare LIFEWATCH with other change point detectors on a common ground: when a change point is detected, we create a new cluster and assign data points from the current mini-batch to it. Otherwise, we assign them to the closest cluster in the model. We selected BOCPD [23] and ECP [28] since they showed to be the best two performing competitor methods in the change point detection task on our datasets (see Table I).
- **Conventional clustering**: We consider three methods, each of them representative of different clustering paradigms: hierarchical (BIRCH [38]), density-based (OPTICS [39]) and partitional (MEAN-SHIFT [40]). Since these methods are not incremental, we execute experiments providing them with the full dataset as input, thus giving them a strong advantage for the identification of the target number of clusters.
- **Stream clustering**: We consider state-of-the-art methods in incremental/stream clustering: DENSTREAM [41], HDBSCAN [42], and TRESTLE [43]. In order to setup a good value for their parameters (variants of the *eps, minPts* parameters in the well-known DBSCAN algorithm), we apply the Elbow rule [44] considering the distances on a first batch of data consisting of 100 points.

Results in Table II show that LIFEWATCH presents the best results in terms of the number of identified clusters (NC) with respect to the actual number of tasks in the scenarios for all datasets except for **Eye**. A first inspection shows that MEAN-SHIFT extracts, on average, a number of clusters that is closer to the ground truth of that specific dataset when compared to LIFEWATCH (difference of 0.5). However, a more detailed analysis reveals that MEAN-SHIFT mostly extracts the same number of clusters for all datasets. It is also important to remark that clustering algorithms have the advantage of seeing the full datasets, whereas LIFEWATCH accomplishes this task in a incremental way, without any prior knowledge (**RQ2**). Moreover, the average number of clusters extracted by LIFEWATCH is very close to the optimal results ($2.50 \approx 2$), which is a remarkable result. A visual perspective of lifelong consolidation results is shown in Figure 4, which allows to easily grasp the best performing methods (in terms of number of identified tasks) by comparing bars with the dashed line (number of actual tasks).

Overall, these results show that, even if LIFEWATCH is not strictly a consolidation method resembling prior attempts in lifelong learning, it can be fruitfully exploited for this task, without the burden of training complex deep neural networks (**RQ2**). Moreover, results show that other change point detectors are not able to provide high-quality results for this task, since they can only detect generic changes and cannot detect a recurrence of tasks due to their lack of memory. Finally, while LIFEWATCH identifies a near-optimal number of tasks, classical clustering algorithms and stream clustering algorithms are not able to extract a precise representation of the data in the lifelong scenario (**RQ3**).

VI. Conclusions and Future Work

In this work, we have tackled a challenging and previously unaddressed problem of lifelong change point detection with multivariate time series data in the form of data streams. We proposed LIFEWATCH, a novel Wasserstein distance-based change point detection approach with memory capable of modeling multiple data distributions in a fully unsupervised manner, and discriminating between changes characterized by the appearance of a new task and changes that rather describe a recurring or previously seen task. An extensive experimental evaluation with a large number of benchmark datasets reveals

TABLE II

LIFELONG CONSOLIDATION: RESULTS IN TERMS OF *Homogeneity* (H), *Completeness* (C), *V-Measure* (VM), AND *Number of Clusters* (NC) WITH FOUR MULTIVARIATE TIME SERIES DATASETS. THE ACTUAL NUMBER OF TASKS FOR EACH DATASET (GROUND TRUTH) IS REPORTED IN PARENTHESIS. THREE CLASSES OF METHODS: *i)* CHANGE POINT DETECTORS COMBINED WITH A UNIFIED CLUSTERING STRATEGY (TOP), *ii)* CONVENTIONAL CLUSTERING (CENTER), AND *iii)* STREAM CLUSTERING (BOTTOM). THE BEST RESULTS IN TERMS OF NC ARE MARKED IN BOLD.

	Movement				**HAR**			
	H	**C**	**VM**	**NC (15)**	**H**	**C**	**VM**	**NC (6)**
LIFEWATCH	0.26±0.05	0.30±0.02	0.28±0.04	**15.20±5.53**	0.42±0.05	0.60±0.14	0.48±0.04	**6.40±3.32**
BOCPD	0.41±0.04	0.33±0.02	0.37±0.03	60.30±2.15	0.55±0.07	0.40±0.04	0.46±0.05	20.40±2.15
ECP	0.21±0.04	0.25±0.04	0.23±0.04	19.30±1.49	0.56±0.05	0.42±0.04	0.48±0.04	16.20±0.87
BIRCH	0.75±0.01	0.56±0.01	0.64±0.01	46.60±4.39	0.93±0.02	0.45±0.03	0.61±0.03	87.20±12.20
MEAN-SHIFT	0.00±0.00	1.00±0.00	0.00±0.00	1.00±0.00	0.39±0.00	0.98±0.02	0.55±0.01	2.80±0.87
OPTICS	0.41±0.01	0.51±0.00	0.46±0.01	29.50±0.67	0.17±0.03	0.37±0.01	0.23±0.03	10.50±1.69
DENSTREAM	0.42±0.24	0.23±0.13	0.30±0.17	4.30±0.64	0.64±0.17	0.44±0.15	0.52±0.15	4.40±1.50
HDBSCAN	0.78±0.21	0.39±0.12	0.52±0.15	3.80±0.40	0.90±0.14	0.71±0.34	0.76±0.28	4.10±3.39
TRESTLE	0.25±0.02	0.55±0.04	0.35±0.02	3.70±0.46	0.38±0.01	0.97±0.03	0.54±0.02	2.00±0.00
	Traffic				**Eye**			
	H	**C**	**V**	**NC (3)**	**H**	**C**	**V**	**NC (2)**
LIFEWATCH	0.16±0.13	0.25±0.21	0.19±0.16	**3.17±1.34**	0.11±0.04	0.10±0.04	0.10±0.04	2.50±0.50
BOCPD	0.26±0.05	0.16±0.02	0.20±0.03	13.00±1.53	0.77±0.00	0.22±0.02	0.35±0.03	15.50±2.50
ECP	0.19±0.06	0.16±0.06	0.17±0.06	7.00±1.29	0.72±0.05	0.25±0.03	0.37±0.04	9.00±0.00
BIRCH	0.61±0.02	0.25±0.01	0.36±0.01	21.17±0.69	0.85±0.05	0.30±0.01	0.44±0.01	7.50±0.50
MEAN-SHIFT	0.00±0.00	1.00±0.00	0.00±0.00	1.00±0.00	0.17±0.00	0.16±0.00	0.17±0.00	**2.00±0.00**
OPTICS	0.70±0.00	0.28±0.00	0.40±0.00	20.00±0.00	0.58±0.00	0.18±0.00	0.27±0.00	16.00±0.00
DENSTREAM	0.30±0.30	0.50±0.37	0.21±0.20	3.17±2.11	0.10±0.01	0.19±0.01	0.13±0.00	4.00±1.00
HDBSCAN	0.69±0.44	0.20±0.13	0.31±0.20	9.17±1.77	0.90±0.00	0.32±0.00	0.47±0.00	9.00±0.00
TRESTLE	0.07±0.04	0.13±0.02	0.09±0.03	2.17±0.37	0.64±0.12	0.40±0.07	0.49±0.09	3.00±0.00

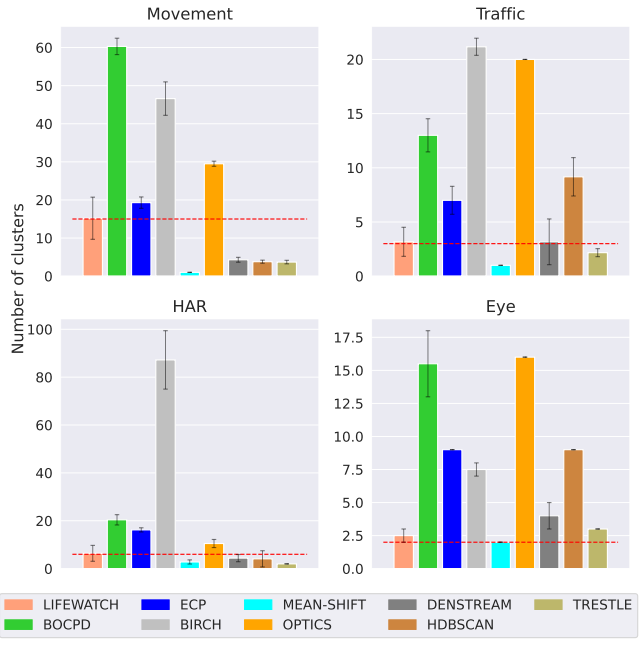


Fig. 4. Lifelong consolidation: Number of clusters extracted (average and standard deviation) for all methods and datasets considered. The dashed line is the ground truth, i.e. the number of actual tasks in each dataset.

that our method is capable of accurately and precisely detecting changes in multivariate time series data, outperforming all considered change point detection competitor methods. Additional results reveal the potential of our method when exploiting the characterization of detected changes in two different types to use in lifelong consolidation. Our method outperforms all considered change point detection, clustering, and stream clustering competitor methods considered in the identification of the correct number of tasks in complex lifelong learning scenarios characterized by recurrence. As future work, we aim to investigate extensions of our method to support lifelong learning settings, as well as improvements in the change point detection procedure to identify periodic behavior in time series data. Moreover, we envision the opportunity to investigate extensions including a systematic evaluation and compression of the learned distributions, which could overcome possible limitations of the method in fast evolving scenarios with a large number of distributions.

ACKNOWLEDGMENTS

The work presented in this article was partially funded by DARPA through the project "Lifelong Streaming Anomaly Detection" (Grant N. A19-0131-003 and A21-0113-002). The authors also acknowledge the support of the Polish Ministry of Education and Science assigned to AGH University of Science and Technology in Kraków, Poland. The authors also acknowledge the support of NVIDIA through the donation of a Titan V GPU.

REFERENCES

[1] M. Bahri, A. Bifet, J. Gama, H. M. Gomes, and S. Maniu, "Data stream analysis: Foundations, major tasks and tools," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 3, p. e1405, 2021.

[2] J.-L. Reyes-Ortiz, L. Oneto, A. Sama, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, vol. 171, pp. 754–767, 2016.

[3] M. Ceci, R. Corizzo, N. Japkowicz, P. Mignone, and G. Pio, "Echad: embedding-based change detection from multivariate time series in smart grids," *IEEE Access*, vol. 8, pp. 156 053–156 066, 2020.

[4] R. Corizzo, M. Ceci, G. Pio, P. Mignone, and N. Japkowicz, "Spatially-aware autoencoders for detecting contextual anomalies in geo-distributed data," in *International Conference on Discovery Science*. Springer, 2021, pp. 461–471.

[5] A. Essien, I. Petrounias, P. Sampaio, and S. Sampaio, "A deep-learning model for urban traffic flow prediction with traffic events mined from twitter," *World Wide Web*, vol. 24, no. 4, pp. 1345–1368, 2021.

[6] R. Corizzo, M. Baron, and N. Japkowicz, "Cpdga: Change point driven growing auto-encoder for lifelong anomaly detection," *Knowledge-Based Systems*, p. 108756, 2022.

[7] Z. Mai, R. Li, H. Kim, and S. Sanner, "Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 3584–3594.

[8] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio, "Gradient based sample selection for online continual learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[9] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-GEM," in *International Conference on Learning Representations*, 2019.

[10] Z. Li and D. Hoiem, "Learning without forgetting," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 614–629.

[11] D. Shim, Z. Mai, J. Jeong, S. Sanner, H. Kim, and J. Jang, "Online class-incremental continual learning with adversarial shapley value," in *AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9630–9638.

[12] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems*, vol. 51, no. 2, pp. 339–367, 2017.

[13] D. V. Hinkley, "Inference about the change-point in a sequence of random variables," 1970.

[14] Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 2, pp. 114–127, 2012.

[15] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application*. Prentice Hall Englewood Cliffs, 1993, vol. 104.

[16] S. V. Georgakopoulos, S. K. Tasoulis, and V. P. Plagianakos, "Efficient change detection for high dimensional data streams," in *2015 IEEE International Conference on Big Data (Big Data)*, 2015, pp. 2219–2222.

[17] A. J. Scott and M. Knott, "A cluster analysis method for grouping means in the analysis of variance," *Biometrics*, pp. 507–512, 1974.

[18] I. E. Auger and C. E. Lawrence, "Algorithms for the optimal identification of segment neighborhoods," *Bulletin of mathematical biology*, vol. 51, no. 1, pp. 39–54, 1989.

[19] P. Fryzlewicz, "Wild binary segmentation for multiple change-point detection," *The Annals of Statistics*, vol. 42, no. 6, pp. 2243–2281, 2014.

[20] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal detection of changepoints with a linear computational cost," *Journal of the American Statistical Association*, vol. 107, no. 500, pp. 1590–1598, 2012.

[21] P. Fearnhead and G. Rigaill, "Changepoint detection in the presence of outliers," *Journal of the American Statistical Association*, vol. 114, no. 525, pp. 169–183, 2019.

[22] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[23] R. P. Adams and D. J. MacKay, "Bayesian online changepoint detection," *arXiv preprint arXiv:0710.3742*, 2007.

[24] J. Knoblauch, J. E. Jewson, and T. Damoulas, "Doubly robust bayesian inference for non-stationary streaming data with beta-divergences," *Advances in Neural Information Processing Systems*, vol. 31, pp. 64–75, 2018.

[25] J. Knoblauch and T. Damoulas, "Spatio-temporal bayesian on-line changepoint detection with model selection," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2718–2727.

[26] Z. Harchaoui, E. Moulines, and F. R. Bach, "Kernel change-point analysis," in *Advances in Neural Information Processing Systems*, 2009, pp. 609–616.

[27] K. Haynes, P. Fearnhead, and I. A. Eckley, "A computationally efficient nonparametric approach for changepoint detection," *Statistics and Computing*, vol. 27, no. 5, pp. 1293–1305, 2017.

[28] D. S. Matteson and N. A. James, "A nonparametric approach for multiple change point analysis of multivariate data," *Journal of the American Statistical Association*, vol. 109, no. 505, pp. 334–345, 2014.

[29] K. Faber, R. Corizzo, B. Sniezynski, M. Baron, and N. Japkowicz, "Watch: Wasserstein change point detection for high-dimensional time series data," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 4450–4459.

[30] M. Hallin, G. Mordant, and J. Segers, "Multivariate goodness-of-fit tests based on wasserstein distance," *Electronic Journal of Statistics*, vol. 15, no. 1, pp. 1328–1371, 2021.

[31] G. J. J. van den Burg and C. K. I. Williams, "An evaluation of change point detection algorithms," *arXiv preprint arXiv:2003.06222*, 2020.

[32] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *ESANN*, 2013.

[33] Dias, Daniel, Peres, Sarajane, Bscaro, and Helton, "Libras Movement," UCI Machine Learning Repository, 2009.

[34] R. Pinto Ferreira, "Combination of artificial intelligence techniques for prediction the behavior of urban vehicular traffic in the city of são paulo," 03 2016, pp. 1–5.

[35] O. Roesler, "EEG Eye State," UCI Machine Learning Repository, 2013.

[36] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[37] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian journal of statistics*, pp. 65–70, 1979.

[38] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: An efficient data clustering method for very large databases," in *1996 ACM SIGMOD International Conference on Management of Data*, 1996, p. 103–114.

[39] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," *ACM Sigmod record*, vol. 28, no. 2, pp. 49–60, 1999.

[40] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[41] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *SDM*, 2006.

[42] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 160–172.

[43] C. J. MacLellan, E. Harpstead, V. Aleven, K. R. Koedinger *et al.*, "Trestle: a model of concept formation in structured domains," *Advances in Cognitive Systems*, vol. 4, pp. 131–150, 2016.

[44] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms," in *16th IEEE International Conference on Tools with Artificial Intelligence*, 2004, pp. 576–584.

Chapter 14

VLAD: Task-agnostic VAE-based lifelong anomaly detection

In this work, we devised VLAD – a novel concept-agnostic lifelong anomaly detection method that leverages a replay-based strategy to provide knowledge retention while adapting to an evolving environment. VLAD utilizes a lifelong change point detection along with hierarchically organized memory to overcome the challenges of concept-agnostic scenarios. Moreover, this memory serves as a replay buffer utilized during model updates, which take place according to a devised strategy.

Our extensive experimental evaluation was conducted on multiple real-world datasets in the concept-agnostic scenario in which the model has no information about the concept being currently processed. The results showcased that VLAD can achieve better anomaly detection performance than non-lifelong state-of-the-art anomaly detection methods in such a challenging environment. Moreover, we observed that a combination of hierarchical memory with lifelong change point detection and memory summarization significantly reduces forgetting. Noteworthy, VLAD achieves good results with a small memory footprint.

Contributions of Kamil Faber: In this work, Kamil Faber co-designed and developed the VLAD method, designed and developed the evaluation framework. Kamil Faber carried out the experiments and was co-author of all sections of the paper.

This paper was published in the Neural Networks journal (IF 7.8; 200 points).

Note: In this paper, we use the term "task" instead of "concept" as it was published before the term "concept" was created.

Neural Networks 165 (2023) 248–273

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

VLAD: Task-agnostic VAE-based lifelong anomaly detection

Kamil Faber [a,*], Roberto Corizzo [b,*], Bartlomiej Sniezynski [a], Nathalie Japkowicz [b]

[a] AGH University of Science and Technology, Institute of Computer Science, Adama Mickiewicza 30, Krakow, 30-059, Poland
[b] American University, Department of Computer Science, 4400 Massachusetts Ave NW, Washington, 20016, DC, United States

ARTICLE INFO

Article history:
Available online 27 May 2023

Keywords:
Lifelong learning
Anomaly detection
Lifelong anomaly detection
Continual learning
Neural networks

ABSTRACT

Lifelong learning represents an emerging machine learning paradigm that aims at designing new methods providing accurate analyses in complex and dynamic real-world environments. Although a significant amount of research has been conducted in image classification and reinforcement learning, very limited work has been done to solve lifelong anomaly detection problems. In this context, a successful method has to detect anomalies while adapting to changing environments and preserving knowledge to avoid catastrophic forgetting. While state-of-the-art online anomaly detection methods are able to detect anomalies and adapt to a changing environment, they are not designed to preserve past knowledge. On the other hand, while lifelong learning methods are focused on adapting to changing environments and preserving knowledge, they are not tailored for detecting anomalies, and often require task labels or task boundaries which are not available in task-agnostic lifelong anomaly detection scenarios. This paper proposes VLAD, a novel **V**AE-based **L**ifelong **A**nomaly **D**etection method addressing all these challenges simultaneously in complex task-agnostic scenarios. VLAD leverages the combination of lifelong change point detection and an effective model update strategy supported by experience replay with a hierarchical memory maintained by means of consolidation and summarization. An extensive quantitative evaluation showcases the merit of the proposed method in a variety of applied settings. VLAD outperforms state-of-the-art methods for anomaly detection, presenting increased robustness and performance in complex lifelong settings.

1. Introduction

Lifelong learning represents an emerging machine learning paradigm that involves cross-disciplinary challenges in the fields of biology, neuroscience, and computer science (Parisi, Kemker, Part, Kanan, & Wermter, 2019). Discoveries in these fields, such as the Complementary Learning Systems (CLS) theory (Kumaran, Hassabis, & McClelland, 2016), the stability–plasticity dilemma (Grossberg, 1982) and Adaptive Resonance Theory (Grossberg, 2013) inspired the design and implementation of new models and methods that provide accurate analyses in complex and dynamic real-world environments. In such environments, data characteristics change over time, presenting new distributions and tasks that are not known beforehand. Although this phenomenon is commonly observed in anomaly detection applications, most lifelong learning works so far have been directed towards image classification and reinforcement learning (Abel, Arumugam, Lehnert and Littman, 2018; Abel, Jinnai, Guo, Konidaris and Littman, 2018; Parisi, Tani, Weber, & Wermter, 2017). Emerging trends include lifelong learning in the context of data streams (Korycki & Krawczyk, 2021b), learning in the presence of concept drift (Korycki & Krawczyk, 2021a), and 3D object recognition with grasp synthesis (Santhakumar & Kasaei, 2022). However, the scarce availability of research works in lifelong anomaly detection (Corizzo, Baron, & Japkowicz, 2022; Frikha, Krompaß, & Tresp, 2021) significantly limits the learning capabilities of currently available anomaly detection methods in lifelong scenarios. Lifelong learning could provide a valuable contribution in the anomaly detection problem, allowing to retain and reuse past knowledge in recurring scenarios that can be frequently observed in most time-evolving real-life domains. Examples include human behavior and weather phenomena, which are recurring in nature, and allow the exploitation of past knowledge to anticipate and provide timely and accurate detection of anomalies.

A successful lifelong anomaly detection method requires the ability to: (i) detect anomalies; (ii) adapt to changing environments; and (iii) preserve knowledge in order to avoid catastrophic forgetting. While state-of-the-art online anomaly detection methods provide the ability to detect anomalies and adapt to a changing environment, they are not designed to preserve past knowledge. Instead, efforts are primarily directed towards online learning capabilities, which yield adaptive models that present high plasticity and limited stability, which makes them unable

* Corresponding authors.
E-mail addresses: kfaber@agh.edu.pl (K. Faber), rcorizzo@american.edu (R. Corizzo), bartlomiej.sniezynski@agh.edu.pl (B. Sniezynski), japkowic@american.edu (N. Japkowicz).

https://doi.org/10.1016/j.neunet.2023.05.032
0893-6080/

to preserve past knowledge (Khan & Madden, 2014; Li, Zhao, Botta, Ionescu, & Hu, 2020; Zhao et al., 2021). On the other hand, lifelong learning methods are focused on adapting to changing environments while preserving knowledge, but are not designed to provide specialized anomaly detection capabilities. Another shortcoming of lifelong learning methods is that emphasis is put on finding a stability–plasticity trade-off in class-incremental (or task-incremental) scenarios that assume the availability of task boundaries (and task labels) (Van de Ven & Tolias, 2019). The availability of task boundaries implies that the model is aware when the task has changed, whereas the availability of task labels implies that the model is aware of the actual task for currently processed .[1] However, this assumption is often too strong in anomaly detection problems, where there is no explicit definition of tasks. In this context, a successful model should be able to detect transitions between tasks and recognize the current task without explicit knowledge, which characterizes task-agnostic scenarios. Some preliminary attempts in the field of anomaly detection with knowledge retention have been made (Corizzo et al., 2022; Frikha et al., 2021; Wiewel & Yang, 2019), although none of them addresses these three challenges simultaneously.

This paper proposes VLAD, a novel a **V**AE-based **L**ifelong **A**nomaly **D**etection method that simultaneously addresses the desired properties for task-agnostic lifelong anomaly detection, i.e., providing the ability to detect anomalies, while adapting to changing environments, and preserving past knowledge in challenging scenarios without clear task boundaries.

Recalling the three challenges defined above, VLAD leverages the combination of lifelong change point detection with an effective VAE model update strategy in order to accomplish adaptation to changing environments in task-agnostic scenarios. To preserve past knowledge, VLAD features an experience replay approach that leverages an efficient and accurate organization of experiences in a continually maintained memory. Finally, for anomaly detection, VLAD relies on the capabilities of the VAE model and efficient update strategy enhanced by experience replay.

Specifically, VLAD discovers changes by means of a *Lifelong Change Point Detection* component that processes new data in an unsupervised manner, and characterizes changes in different types, supporting an informed learning workflow of the model. Knowledge is continually acquired and organized in memory. Inspired by the CLS theory, which stipulates the importance of the interaction between the hippocampus and the neocortex in human beings to consolidate experiences in the memory, we propose a hybrid short and long-term memory managed by a *Consolidation* component. Experiences in long-term memory are organized hierarchically, reflecting the natural taxonomy of knowledge in concepts and sub-concepts. Memory is further refined by means of a *Memory Summarization* component that accomplishes the goal of retaining relevant experiences, while yielding a compact representation that guarantees constrained upper bound storage requirements. A custom *Experience Replay* component completes the workflow, being responsible for triggering model updates and leveraging the most relevant experiences available. An extensive quantitative evaluation showcases the merit of the proposed method in a variety of applied settings, including host-based intrusion detection, network intrusion detection, and renewable energy domains. Experimental results show that VLAD outperforms state-of-the-art methods for anomaly detection, presenting increased robustness and performance in complex lifelong settings.

In summary, the key contributions of this work can be summarized as:

- A task-agnostic lifelong anomaly detection method that solves three crucial challenges for the anomaly detection problem in lifelong learning scenarios: detect anomalies, adapt to changing environments, and preserve knowledge in order to avoid catastrophic forgetting.
- A lifelong change point detection approach that identifies task changes, supporting adaptation in task-agnostic scenarios. The approach does not only discover generic changes, but also characterizes them in new and recurring changes, which have impact on the learning workflow of the method.
- High-quality experience replay based on memory consolidation and summarization algorithms, providing knowledge retention while yielding an improved anomaly detection performance on previously seen tasks.
- An extensive quantitative evaluation and analysis that shows the potential of the proposed method on a variety of applied real-world evolving domains, including host-based intrusion detection, network-based intrusion detection, and renewable energy.
- A new evaluation scenario for lifelong anomaly detection which features multiple concepts of the normal class and anomaly types presented over time, which effectively bridges the fields of anomaly detection and lifelong learning.

The paper is structured as follows. Section 2 formalizes our proposed analytical scenario tailored for lifelong anomaly detection. Section 3 summarizes relevant works in the field of lifelong learning and anomaly detection. Section 4 provides an overview of the method, explaining its workflow and input parameters. Section 5 describes our method in detail, in terms of its workflow and components. Section 6 describes the scenarios and the experimental settings used in our study. Section 7 presents and discusses the results of the conducted experiments. Finally, Section 8 concludes the paper, summarizing the key getaways and outlining ideas for future work.

2. Problem definition

A lifelong learning setting consists of a continuous process where a sequence of tasks $T_1, T_2, \ldots, T_N$ are presented to a learning method. In this setting, the learner should be able to acquire new skills and adapt to newly presented tasks while leveraging previously learned knowledge to jointly address new challenges and the recurrence of previously observed tasks.

In image classification works, a task typically corresponds to a subset of k classes ($T_i = \{c_{i_1}, c_{i_2}, \ldots, c_{i_k}\}$) among the available classes $C = \{c_1, c_2, \ldots, c_n\}$, $T_i \subset C$. For instance, in handwritten digit classification, it is possible to split 10 classes (0 to 9) into 5 binary tasks containing 2 digits each, as in the Split-MNIST dataset (Zenke, Poole, & Ganguli, 2017). The accuracy of the model is assessed on its ability to exhibit satisfactory classification performance on all classes learned so far.

Different types of scenarios are possible based on some key assumptions. While supervised machine learning settings require data features and class labels as in the conventional machine learning setting, lifelong learning scenarios can also be categorized based on task labels, task boundaries, and assumptions on their availability.[2] For instance, task-incremental scenarios assume the availability of both task labels (i.e., the model is aware of the set of classes involved in the current task) and task boundaries (i.e the model is aware when a task transition occur). Class-incremental scenarios do not require the knowledge of task

[1] A more detailed discussion with examples is provided in Section 2.

[2] For simplicity, in this discussion we focus on these two properties. A complete lifelong learning setting could also be described by the experience content (new classes *vs.* new instances) and problem (unique vs partitioned).

labels, but they require task boundaries as detailed in Lomonaco (2019) and Van de Ven and Tolias (2019). Task-agnostic scenarios appear more challenging since they do not assume the availability of both task labels and task boundaries. Recalling the example of hand-written digits classification mentioned above, the scenario can be either formulated as task-incremental or class-incremental, depending on whether the information about which digits are provided in the current task is available, or not, respectively. Task-agnostic scenarios are less frequent in this context, given the nature of image classification.

In this paper, we identify the corresponding counterparts of the scenarios devised in image classification for the lifelong anomaly detection task. A mapping effort is required since anomaly detection features a normal class and an anomaly class that changes over time, instead of having new and independent classes learned over time. In this context, a lifelong anomaly detector must adapt to the new behavior of the normal class while retaining past knowledge. We consider an example where multiple servers generate data to be analyzed by a single anomaly detection method that needs to monitor new servers being added over time. The example is illustrated in Fig. 1. The three scenarios differentiate themselves based on the availability of task labels and task boundaries. In this context, task labels correspond to information about which server is the current data source, while task boundaries correspond to information about whether the currently analyzed server has changed.

In a *task-incremental* scenario, the model has availability of both task labels and task boundaries, meaning that it has knowledge of which data comes from which server at all times. This information is available both during the learning and inference phases. A more challenging scenario is regarded as *concept-incremental*, where the model knows that the data source has changed (task boundaries) but does not know which is the current source (no task labels), both during learning and inference phases. The main difference compared to the class-incremental scenario in image classification is that the new behavior of the normal class (which in this example corresponds to different servers) is presented over time, instead of new classes. Finally, in a *task-agnostic* scenario the model does not know that the current source of data has changed (no task boundaries) and does not know which server is currently acting as the data source (no task labels). Therefore, the model has to adapt and detect anomalies without knowing which task does the data belong to, and whether the task has changed.

We argue that the task-agnostic scenario more closely resembles many real-world characteristics of anomaly detection problems. First, domains are mainly characterized by multiple concepts of the normal and anomaly classes that are unknown at the beginning instead of well-defined sets of classes that are known beforehand. Second, in most scenarios it is unrealistic to assume the availability of task labels and task boundaries. Third, anomaly detection fundamentally requires tools and methods that operate in an environment with limited supervision or complete lack of supervision. For instance, in the context of network intrusion detection, a realistic scenario with network traffic processed in real-time may present transitions from web browsing to file uploads on cloud storage, with several hours of inactivity in between.

For these reasons we focus on lifelong anomaly detection in a task-agnostic scenario. The scenario does not require strict task boundaries and does not provide task label information. More specifically, in the specific scenario addressed in this study, data is processed sequentially in multiple learning batches $S^{(t)}, S^{(t+1)}, \ldots, S^{(t+k)}$. Each batch contains a set of data points (also referred as experiences in the remainder of the paper) represented as multidimensional feature vectors. During the *learning phase*, the presentation of each task $T = (T_1, T_2, \ldots, T_N)$ may span across multiple batches, i.e., $T_i = (S^{i^1}, S^{i^2}, \ldots, S^{i^k})$.

It is worth noting that the number of batches presented for each task are not known in advance, and the information about transitions between tasks is not provided to the learning models. Tasks in this context represent different characteristics of the normal class, which is supposed to change over time.

The *evaluation phase* takes place each time the presentation of a task is completed during the *learning phase*. During this phase, a set of batches $E = \{E_1, E_2, \ldots, E_N\}$, $|E| = |T|$ is used to evaluate the model on all tasks. Each batch E_i contains both normal data and anomalies from a given task T_i and has an arbitrary size. It is important to note that data points in the evaluation batches in E are put aside from the batches used in the learning batches and are never seen by the model during the *learning phase*. The scenario described so far is represented in Fig. 2. In regard to the type of anomalies of interest in this study, their connotation may change as the learning scenario progresses. More specifically, from the perspective of a single task, anomalies may be regarded as global, although when more tasks are considered in combination, anomalies can become local to a specific task.

While the ability of the model to discriminate between normal data and anomalies is evaluated in terms of anomaly detection performance, our proposed scenario also allows us to assess models based on their ability to adapt to new tasks, while retaining knowledge of previously observed tasks. Therefore, the performance across all learned tasks is taken into account to assess the ability of the model to incorporate new knowledge while avoiding forgetting.

This scenario, which takes inspiration from the evaluation protocol used in lifelong image classification, presents advantages for anomaly detection problems. In particular, it allows to assess models in more complex circumstances and from a broader perspective, taking into account forgetting and knowledge transfer across tasks (typically measured using forward and backward transfer), instead of relying solely on the anomaly detection performance.

3. Related works

In this section, we describe related works in the literature in relationship with the three key challenges of lifelong anomaly detection problem. Our proposed method aims to address these three challenges simultaneously, overcoming the limitations of most currently available works, which tackle two out of three key challenges.

3.1. Lifelong learning methods

Lifelong learning approaches are typically neural network-based. Following the characterization in Joseph and Balasubramanian (2020) and Parisi et al. (2019), they can be characterized in the following three types:

- Regularization-based
- Replay-based
- Dynamic architectures

All these methods aim at adapting to changing environments while preserving knowledge in order to avoid catastrophic forgetting. They do not detect anomalies and they do not support task-agnostic scenarios where task labels and boundaries are not available. We describe them in the following subsections.

Our proposed method is most closely related to replay-based methods and expands on their existing capabilities, being specifically tailored for task-agnostic anomaly detection, where two classes (normal and anomaly) change over time, instead of multi-class problems typical of class-incremental and task-incremental scenarios in image classification problems.

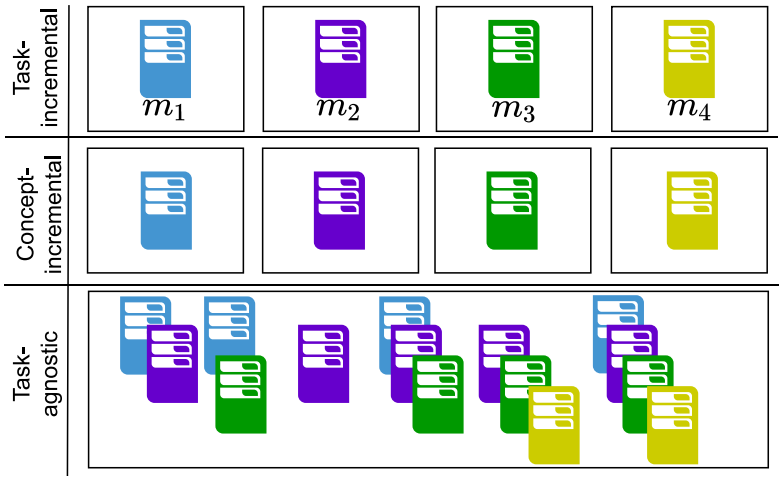


Fig. 1. Proposed scenario characterization for lifelong anomaly detection. Data is generated by multiple servers that are added over time. A *task-incremental* scenario assumes the availability of both task labels (indication about which server is the current data source, i.e., m_1, m_2, m_3, m_4, identified by different colors) and task boundaries (an indication that the current data source has changed). A *concept-incremental* scenario only assumes the availability of task boundaries. In that scenario, the server representing the data source is not indicated. A *task-agnostic* scenario assumes the availability of neither task labels nor task boundaries. The three identified scenarios are presented in increasing order of complexity.

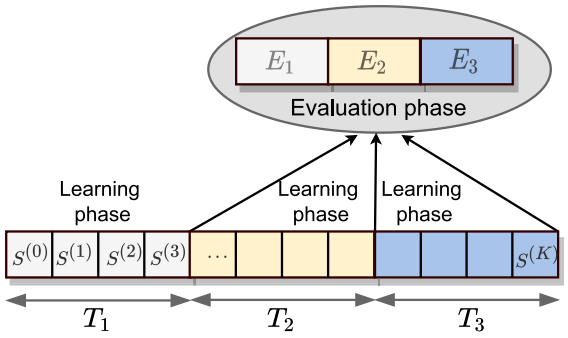


Fig. 2. Example of data batches $(S^{(0)}, S^{(1)}, \ldots, S^{(K)})$, representing a changing data distribution with tasks T_1, T_2, T_3 and corresponding evaluation batches E_1, E_2, E_3.

3.1.1. Regularization-based methods

These methods define constraints on the weights update of neural network models, in order to overcome catastrophic forgetting. The Elastic Weight Consolidation (EWC) method in Sharif Razavian, Azizpour, Sullivan, and Carlsson (2014) leverages the OverFeat network (Sermanet et al., 2014) as a feature extractor and tackles its ability to perform a number of recognition tasks including image classification, scene recognition, and image retrieval. Results show that using pre-trained off-the-shelf features provided by OverFeat in combination with simple classifiers, yields a higher accuracy than models optimized on specific datasets. A similar approach is used in Inter-Task Synaptic Mapping (ISYANA) (Mao, Weng, Pratama, & Yee, 2021), which combines the notion of task-to-neuron relationship with task-to-task relationship, constraining different neurons to focus on different tasks. While the importance of a neuron for a given task is determined using mutual information, task-to-task mapping is computed using the Kullback–Leibler (KL) divergence of class distributions for the current tasks with respect to old tasks. On the other hand, Kirkpatrick et al. (2017) trains and updates the model in a lifelong setting, but identifies which parameters are the most important for the current task and reduces their ability to change using a penalty factor. This mechanism is inspired by the concept of synaptic consolidation in the human brain, which allows to accomplish lifelong learning by reducing the plasticity of synapses that are vital to previously learned tasks.

Other approaches include the AR1 method, which (Maltoni & Lomonaco, 2019) dynamically updates the amount of change allowed for weights across different batches. Alternative solutions are proposed in the context of Bayesian learning. For instance, Chen, Diethe, and Lawrence (2019) describe that, in this context, retaining knowledge from previous tasks is possible through posterior distributions over the parameters (knowledge from inference in previous tasks), as well as coresets (knowledge of data distributions of previous tasks). Another approach (Zhao, Wang, Masoomi, & Dy, 2022) proposed Deep Bayesian Unsupervised Lifelong Learning (DBULL), a Bayesian framework that incorporates past knowledge while sequentially updating the belief with new data. New clusters are discovered progressively from unlabeled data, while past knowledge is retained by means of statistics of the latent representation for raw data. A similar idea is proposed in Kurle, Cseke, Klushyn, van der Smagt, and Günnemann (2019), which combined Bayesian neural networks with memory-based online variational Bayes. Authors in Titsias, Schwarz, Matthews, Pascanu, and Teh (2019) propose a Gaussian process-based method that computes summaries of sequentially encountered tasks that are used to regularize learning of future tasks through KL-divergence.

Another method in the context of reinforcement learning is proposed in the Progress and Compress (Schwarz et al., 2018) method, which builds a knowledge base in combination with a neural network. The recently learned knowledge is distilled in the knowledge base while the previous knowledge is preserved adopting the EWC method. The proposed model is shown to improve model performance in Atari and 3D maze navigation domains.

3.1.2. Replay-based methods

Replay-based methods rehearse previously experienced data from already learned tasks. Replaying past experiences has been shown to reduce forgetting in the brain Parisi et al. (2019), and it can be realized adopting different strategies that depend on the tasks, data, and scenarios addressed.

A replay mechanism named flashcards is proposed in Gopalakrishnan, Singh, Fayek, Ramasamy, and Ambikapathi (2022). The method defines and extracts flashcards as reconstructions of random images via recursive passes of an auto-encoder model. Replay is conducted using flashcards on a new network

to remember previously observed tasks. Experiments are performed on different image-based tasks including reconstruction, denoising, task-incremental learning, and new-instance learning classification.

The authors in de Masson D'Autume, Ruder, Kong, and Yogatama (2019) propose a sparse experience replay mechanism with local adaptation. The model interacts with a key–value memory module, that is leveraged both during training and inference. In the training stage, new samples in combination with existing samples from the memory are used to update the model by performing gradient updates. During inference, the most similar samples given the current context are retrieved to fine-tune a local and temporary model in order to extract more accurate predictions. A model called Gradient Episodic Memory (GEM) is proposed in Lopez-Paz and Ranzato (2017) which leverages an episodic memory storing a subset of the observed examples for each observed task. One limitation of the proposed method is that it becomes prohibitive in terms of memory and compute time when the size of the episodic memory and the number of tasks is large. An improved version of the method that alleviates this computational burden that leverages loss averaging is Averaged Gradient Episodic Memory (A-GEM) (Chaudhry, Ranzato, Rohrbach, & Elhoseiny, 2018).

Generative replay is proposed in Shin, Lee, Kim, and Kim (2017), where the generator capability of deep generative models is used to build replay buffers sampling from previously learned tasks. Replay buffers are then used in combination with new data to update a solver model that learns new tasks. The authors in van de Ven and Tolias (2018) proposed a more efficient replay approach that embeds the generative capabilities in the solver model. This is possible through the adoption of feedback connections that are trained to reconstruct inputs from their hidden representations, and a layer of stochastic latent variables that follow a known distribution which allows sampling.

The study in Buzzega, Boschini, Porrello, and Calderara (2021) proposes five optimization tricks for known experience replay-based approaches: independent buffer augmentation, bias control, exponential learning rate decay, balanced reservoir sampling, and loss-aware reservoir sampling. These approaches are shown to improve results over their baseline counterpart. The study in Isele and Cosgun (2018) explores four strategies for selecting which experiences should be stored: favoring surprise, favoring reward, matching the global training distribution, and maximizing coverage of the state space. Experiments show that the global training distribution is often the best performing strategy.

3.1.3. Dynamic architectures

Neural network models with dynamic architectures were explored in artificial intelligence and related sub-fields before the recent formulation of the lifelong machine learning paradigm (Ünal & Başçiftçi, 2021). In neurocomputing, methods such as NeuroEvolution of Augmenting Topologies (NEAT) (Stanley & Miikkulainen, 2002), Growing Hierarchical Self-Organizing Maps (GHSOM) (Dittenbach, Merkl, & Rauber, 2000), and more recent variants (Faber, Pietron and Zurek, 2021; Malondkar, Corizzo, Kiringa, Ceci, & Japkowicz, 2019; Miikkulainen et al., 2019) represented initial attempts towards allowing a neural network model architecture to change and evolve, also by adding more neurons and layers autonomously, adapting to data characteristics and specific application domains. Optimization methods such as Particle Swarm Optimization (PSO) have also been used to automatically evaluate and tweak neural network architectures that are best suited for a problem at hand Junior and Yen (2019).

Similarly, in the context of lifelong learning, dynamic architectures have impact in the expansion and the evolution of a model architecture. However, differently than methods proposed in past literature, lifelong learning methods explicitly address learning in a continual setting while ensuring a proper stability–plasticity tradeoff. One example is that of Evolved Plastic Artificial Neural Networks (EPANNs), which have the ability of learning from scratch and recovering performance in unseen conditions, potentially supporting a large variety of different neuron types, network architectures, and plasticity rules (Soltoggio, Stanley, & Risi, 2018).

Some methods consist in the adoption of multiple models. The Continual Neural Dirichlet Process Mixture (CN-DPM) method (Lee, Ha, Zhang, & Kim, 2020) trains a new model for a new task and leaves the existing models intact, which act as a group of experts and retain the knowledge of the past tasks. Each expert has a classifier and a generative model such as VAE, is responsible for a subset of the data. The pool of experts can be expanded based on a Dirichlet Process Mixture with Sequential Variational Approximation. A similar pooling approach is followed in Expert Gate (Aljundi, Chakravarty, & Tuytelaars, 2017), where multiple autoencoder models are trained and combined via a gating approach using a softmax layer, that takes as input the reconstruction errors from the different autoencoders and assigns a test sample to the most confident model. A different approach is proposed by the Meta-Consolidation for Continual Learning (MERLIN) method (Joseph & Balasubramanian, 2020), where multiple models are trained to match a task-specific distribution as new tasks occur. The models are consolidated by replaying parameters generated using earlier task-specific priors, yielding a model ensemble that is used for inference. An eigentask-based approach that trains multiple models with different skills and provides a gating mechanism to select the most relevant model for the current task in the context of reinforcement learning is proposed in Raghavan, Hostetler, Sur, Rahman, and Divakaran (2020).

Other models add new neurons or layers to a single model to handle new tasks while retaining knowledge of previous tasks. The Dynamically Expandable Network (DEN) method (Yoon, Yang, Lee, & Hwang, 2018) dynamically adapts network capacity upon arrival of each task, adding in or splitting/duplicating neurons when necessary. The method identifies neurons that are relevant to the new tasks, and selectively retrains the network parameters associated with them. Subsequently, if the selective retraining is not successful according to a desired loss threshold, the network capacity is expanded, and unnecessary neurons are eliminated via group-sparsity regularization. The Learning without Forgetting (LwF) (Li & Hoiem, 2017) method consists of a CNN with shared parameters and task-specific parameters, which are added as new tasks are observed. Nodes for new classes are added to the output layer, fully connected to the previous layer. These connections are first trained using new data, while the rest of the network is frozen. Then, the whole network is jointly trained until convergence.

Another category of approach is that of selective training and inference. The PackNet method sequentially packs multiple tasks into a single network leveraging network pruning to free some parameters that are dedicated to learn a new task (Mallya & Lazebnik, 2018). This process is performed iteratively to update the network without increasing its capacity. A random path selection method called RPS-Net is proposed in Rajasegaran, Hayat, Khan, Khan, and Shao (2019). Based on a parallel residual neural network design, the method selects the optimal paths for new tasks among a set of randomly sampled candidate paths. Only one path at a time is trained with new data, while previous paths are reused to balance stability and plasticity. A gating mechanism to form hard attention masks that activate or deactivate the output of the units of every layer is proposed in Hard Attention to the Task (HAT) (Serra, Suris, Miron, & Karatzoglou, 2018). Paths are dynamically created and destroyed across layers and they can be preserved when learning a new task.

3.2. Anomaly detection methods

Anomalies can be characterized in two taxonomies (Goldstein & Uchida, 2016). The former differentiates between local and global anomalies. While local anomalies are only anomalous when compared with its close-by neighborhood, global anomalies are always anomalous when the entire dataset is considered. The latter differentiates between point, collective, and contextual anomalies. While point anomalies are single anomalous data points in a larger dataset, collective anomalies are represented as a set of many data points, and contextual anomalies are data points that are anomalous only taking a certain context into account, and are normal data points otherwise.

Anomaly detection methods can characterized in supervised, semi-supervised, and unsupervised (Goldstein & Uchida, 2016; Pang, Shen, Cao, & Hengel, 2021). Supervised methods have strong assumptions about labels availability for both normal and anomaly classes. These approaches are often impractical since the anomaly class represents the minority of the data, and it is hard to acquire a sufficient amount of anomalies to learn a robust model.

Semi-supervised methods train models using exclusively normal data and try to identify anomalies in unseen data based on their difference with respect to the learned data distribution of the normal class. This idea is also known as one-class learning (Goldstein & Uchida, 2016). These methods provide a good trade-off between data availability and robustness, since usually a large amount of normal data is available, whereas anomaly data is scarce (Pang et al., 2021).

Unsupervised methods are the most flexible in that no assumptions on labels are provided, work solely based on intrinsic properties of the dataset, and they do not present a strict separation between training and prediction phases. Moreover, studies suggest to rely on semi-supervised methods if enough labeled normal data is available, in order to achieve more robust models. It is interesting to note that some unsupervised methods support semi-supervised learning settings and vice-versa (Goldstein & Uchida, 2016).

In our study, we put emphasis on methods that can work in a semi-supervised setting, since our scenarios do not assume the availability of anomalies. Moreover, the learning setting consisting of two phases (learning and evaluation) supported by these methods matches with the lifelong anomaly detection problem addressed in this work. Further details are provided in Section 2.

Prominent examples of solid and well-established anomaly detection methods are One-Class Support Vector Machines (OC-SVM), Local Outlier Factor (LOF), Isolation Forests (IF), and auto-encoders.

Isolation Forest leverages an ensemble of tree-based models in order to calculate an isolation score for all data samples (Liu, Ting, & Zhou, 2008). The isolation score is the average path length from the root of the tree to a single node representing a sample. Shorter paths indicate samples that are significantly different than the most represented data distribution, and therefore are easier to isolate. The model is learned by creating trees that can isolate any data object from the rest of the data using recursive random splits on feature values.

By executing dot products between data points from the input space, OC-SVM learns a separating hyperplane in a high-dimensional space with kernels (Schölkopf, Williamson, Smola, Shawe-Taylor, & Platt, 2000). OC-SVM classifies new data as similar or different in terms of the training distribution and its position inside the decision boundary during the prediction phase.

LOF measures the local density deviation of a given data sample with respect to its neighbors (Breunig, Kriegel, Ng, & Sander, 2000). The ratio of a sample's own local density to the average local density of k-nearest neighbors determines its score. The samples that are similar to the training data distribution have a density that is similar to one of their neighbors. The samples outside the training data distribution, on the other hand, have a substantially lower local density.

Other prevalent one-class approaches are auto-encoders. They are neural networks designed to reconstruct the input as the output. Auto-encoders learn the data distribution in a one-class manner by minimizing the difference between the reconstructed and original data (so called reconstruction error) (Sengupta et al., 2020). Therefore, during the detection phase, how well the auto-encoder can reconstruct new samples determines whether they belong to the same distribution as the training data.

Two recent state-of-the-art anomaly detection methods that are quickly gaining recognition are Copula-Based Outlier Detection (COPOD) (Li et al., 2020) and Scalable Unsupervised Outlier Detection (SUOD) (Zhao et al., 2021). COPOD is an anomaly detector that models data by constructing an empirical copula, i.e., a multivariate cumulative distribution function for which the marginal probability distribution of each variable is uniform in the interval [0, 1]. In the detection phase, the method leverages this copula to predict the tail probabilities of each data sample to determine the level of its "extremeness". SUOD combines a large number of unsupervised, heterogeneous anomaly detection models. It focuses on accelerating the three complementary aspects, i.e., dimensionality reduction for high-dimensional data, model approximation for complex models, and execution efficiency improvement for task load imbalance within distributed systems.

Although most of these methods are well suited for detecting anomalies, but do not directly support changing environments, they can be easily adapted by means of model update using most recent data, which brings them closer to methods that are designed for online anomaly detection methods like COPOD and SUOD. However, despite detecting anomalies while adapting to changing environments, these methods are unable to preserve knowledge in order to avoid catastrophic forgetting.

Anomaly detection methods that are more specifically oriented to lifelong learning settings are just starting to emerge in recent literature (Faber, Corizzo, Sniezynski and Japkowicz, 2022). A transfer learning approach tailored to anomaly detection in surveillance videos is proposed in Doshi and Yilmaz (2020). A generative anomaly detection approach with experience replay that incorporates data from the previous tasks for model retraining is proposed in Wiewel and Yang (2019). The ARCADe method presents a meta-learning approach for the identification of the most appropriate parameter values for each task in one-class image anomaly detection (Frikha et al., 2021). However, these methods are primarily focused on the analysis of image and videos, and present the key limitation of requiring explicit task boundaries, which makes them unsuitable for the settings addressed in this paper.

A different attempt to address task-agnostic scenarios is taken by CPDGA, Corizzo et al. (2022) an autoencoder-based approach for anomaly detection involving statistical change point detection and a growing memory of embedding data objects representing the normal class. However, the method focuses on transferring knowledge from one task to the following one (forward transfer), without explicit evaluation on previously learned tasks (backward transfer). Moreover, the adoption of non-lifelong statistical change point detection makes the method prone to false detections as recurring tasks are presented, which in turn limits anomaly detection robustness due to growing memory requirements. Finally, the method does not explicitly address forgetting, since the memory is used exclusively for the anomaly detection stage and not for model update.

Unlike other current work, our proposed method tackles all the limitations identified in one-class learning methods discussed

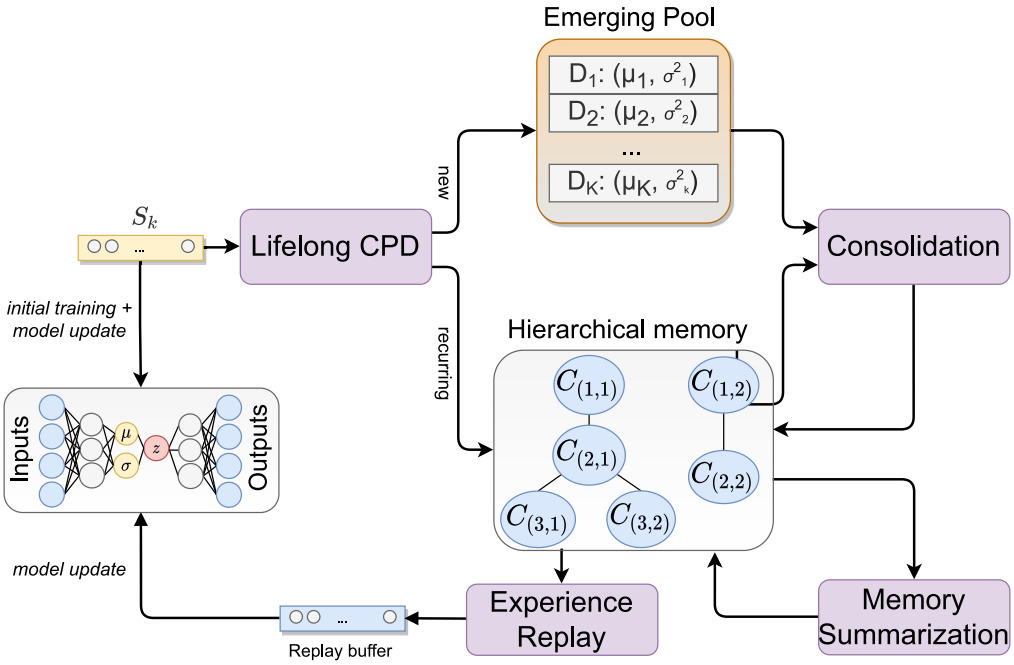


Fig. 3. VLAD: Learning stage, featuring the combined effort of Lifelong CPD, Memory Summarization, Consolidation, and Experience Replay. The VAE model is trained and updated with learning data in combination with replay buffers provided by the Experience Replay component.

above, by simultaneously taking into account anomaly detection, adaptation in changing environments, and knowledge retention to avoid catastrophic forgetting.

4. Method overview

In this section, we describe the preliminaries of our method, the general lifelong learning workflow of the method, and its input parameters.

4.1. Method workflow

A graphical representation of the learning stage is presented in Fig. 3. *Lifelong Change Point Detection* (LCPD) takes place in order to identify possible changes in the data presented by the environment. The component has the ability to discriminate between changes that describe the appearance of recurring distributions and changes that describe the appearance of new distributions. Moreover, the component adds newly encountered distributions to the Emerging Pool (P), which conceptually represents a short-term memory. Incoming data representing recurring distributions is used to update matched concepts in the hierarchical memory, which conceptually represents a long-term memory. The newly observed distributions stored in P are then incorporated in the hierarchical memory ($\mathcal{M}$) through the *Consolidation* component, in the form of concepts, either root concepts or sub-concepts. Conceptually, a sub-concept is a concept associated to a parent concept.

Over time, the *Experience Replay* component recalls prior experiences from the memory and generates a replay buffer that is used in combination with the newly available data to update the Variational Auto-Encoder model in order to prevent forgetting. When the desired memory bound is exceeded, the *Memory Summarization* component is triggered to maintain the memory occupation stable by means of Pyramidal Summarization followed by within concept summarization. More details for each component are provided in the following section.

Recalling the three challenges in lifelong anomaly detection mentioned in Section 1, this learning workflow allows VLAD to address them in a combined manner. Specifically, it allows to maintain an updated model over time, incorporating new distributions of the normal class leveraging an effective model update strategy supported by the Lifelong Change Point Detection component. Moreover, it allows to retain representations of previously learned distributions with the Experience Replay component, which in turn relies on Consolidation and Memory Summarization components. Together with the VAE model, which provides anomaly scores for new data points, the cooperation of all components allows VLAD to perform task-agnostic lifelong anomaly detection.

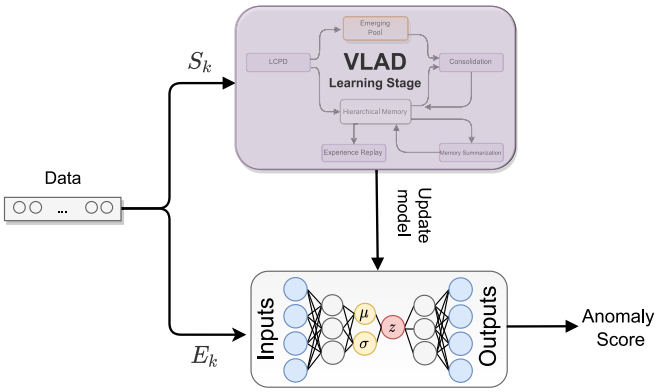


Fig. 4. VLAD: Full method workflow. The model trained during the learning stage predicts anomaly scores on newly incoming data. The learning stage of VLAD is shown in Fig. 3.

Once the VAE has been trained using normal data in a semi-supervised learning setting, evaluation data is processed by the model, and an anomaly score is provided by the model. The full workflow is presented in Fig. 4.

4.2. Input parameters

In this section, we summarize the main parameters of the method to simplify its discussion in the following subsections.

Parameters are discussed separately for each component. A study on their sensitivity is discussed in Section 7.6.2.

- *Lifelong Change Point Detection*: This component is controlled by three main parameters:
  - C_ϵ: it represents the ratio defining how distant the samples can be from the reference distribution to be still considered a part of the same. A low value will make the method more sensitive to detecting new distributions that are close to the current one, whereas a high value will make the method more conservative to the detection of new distributions. The method offers the possibility to automatically tune this parameter according to the internal cohesion of the first task by identifying the minimum value that does not lead to the creation of more than one task on the initial data.
  - C_ω: it controls the size of the mini-batch in which the data points are processed. A small value will allow to detect frequent changes, whereas a large value should be preferred if less frequent changes are expected. Similar to batch size in neural network models, common sense heuristics (such as positive powers of two) can be adopted to facilitate the setup of this parameter. A conservative setting is to set it to a small power of two, which will allow to detect frequent changes using more computational resources.
  - C_κ: it determines the minimum number of points that can be considered a sufficient representation of the distribution. It should be sufficiently high to yield valid distribution estimates, e.g. a multiplicative factor of C_ω.
- *Consolidation*: The W_ϵ parameter represents the ratio for adaptive threshold of Wasserstein distance between two distributions, and influences the creation of root concepts and sub-concepts in the memory. A low value implies that root concepts are more likely to be created than sub-concepts, whereas a large value implies that sub-concepts are more likely to be created than root concepts. It is defined by the user based on domain knowledge and likelihood of sub-concepts in the data.
- *Memory Summarization*: This component is controlled by two main parameters:
  - M_B: it defines the *soft* upper bound for the model's memory size. It is expressed as the number of samples that should be stored in the whole memory. This bound is used in the memory summarization process in order to ensure the compactness of the memory while retaining the most relevant samples. While a low value may lead to a faster forgetting, a high value may prevent the model from learning new concepts in an efficient manner.
  - M_f: it defines the frequency by which the summarization process is triggered, and defines the *hard* upper bound of the memory. A low value may lead to a more frequent execution of the process, whereas a high value may lead to a less frequent execution, which in turn determines a temporarily higher memory occupation.
  - M_k: it defines the number of centroids for the within concept summarization process, which is carried out separately for each concept. Centroids are regarded as meta-experiences that summarize concepts. For a given concept, a high value will store a lower number of similar experiences, whereas a low value will store a higher number of similar experiences.

  All Memory Summarization parameters can be set depending on the available memory resources.
- *Experience Replay*: The R_ψ parameter controls the maximum time between model updates, defined in terms of the processed number of samples. It is used to force model update in the situations described in Section 5.5. Its value is directly related to the expected frequency of changes in incoming data which is domain specific, and should consider the computational cost of model update. A conservative approach taking into account frequent data shifts could be to set this parameter to at least twice the number of data samples observed in the first task.

5. Method

In this section, we describe our proposed method in detail, focusing on the role and the salient aspects of each component. The main algorithm referencing all components is presented in Algorithm 1.

Input: S - input batches

Parameters: C_ϵ, C_ω, C_κ, W_ϵ, M_B, M_f, M_k, R_ψ – described in 4.2

Initialize the Hierarchical Memory: $\mathcal{M} = \emptyset$
Initialize the Emerging Pool: $P = \emptyset$
Steps from the last model update: $t = 0$
Initialize the model: $VAE^{(0)}$

for $S^{(i)} \in S$ **do**
  $\lambda_S \leftarrow False$ // Indication if summarization was carried out
  λ_N, λ_R, P, $\mathcal{M}$ = $LCPD(C_\epsilon, C_\omega, C_\kappa, P, \mathcal{M})$ following Alg. 2
  if λ_N **then** // Consolidation process
    for $D \in P$ **do**
      Determine parent distribution CD using Eq. (4)
      if $CD \neq \emptyset$ **then**
        Add D as sub-concept of CD to $\mathcal{M}$
      else
        Add D as root concept to $\mathcal{M}$
      end
    end
    $P \leftarrow \emptyset$ // Reset Emerging Pool
  end
  if $size(P) + size(\mathcal{M}) > M_B \cdot M_f$ **then** // Summarization process
    for $C_{(l,l')} \in \mathcal{M}$ **do**
      Determine new size for the concept $M(C_{(l,l')})$ using Eq. (5)
      Perform summarization on $C_{(l,l')}$ using Eq: (6), (7), (8)
    end
    $\lambda_S \leftarrow True$ // Set flag that the summarization was carried out
  end
  if λ_N *or* λ_R *or* $t > R_\psi$ *or* λ_S **then** // Update process
    Create replay buffer R using Eq: (9)
    $VAE^{(i)} \leftarrow$ update $VAE^{(i-1)}$ using $S^{(i)}$ and $R^{(i)}$
    $t \leftarrow 0$
  end
end

Algorithm 1: Pseudo-code of the learning process of the VLAD

5.1. Variational auto-encoder

In our method, we adopt auto-encoders to model the distribution of the normal data class. Reconstruction models based on neural networks have the advantage of leveraging non-linear interactions in the feature space, which are typically neglected by simpler one-class learning models (such as LOF) and data compression methods (such as PCA).

In particular, we adopt Variational Auto-Encoders (VAE) since they allow to learn reliable disentangled representations. The prior distribution imposed on the latent representation in a VAE contributes to model fitting due to the KL divergence term. In particular, the disentanglement in VAE models is achieved by storing a unique latent Gaussian for each feature in the embedding space (Higgins et al., 2016; Mathieu, Rainforth, Siddharth, & Teh, 2019). This characteristic leads to statistical implications that make VAE a reliable model for most data sources. We argue that, in the context of lifelong anomaly detection, this characteristic is desirable to effectively recognize whether newly incoming data points belong to one of the previously learned concepts. An accurate representation should increase the probability of yielding a high anomaly score for data points not belonging to concepts describing the normal class. At the same time, it should increase the probability of yielding a low anomaly score for data points belonging to the normal class. As a consequence, the model should provide a more robust anomaly detection performance. Moreover, recent studies showed their reliability in image classification (Pu et al., 2016) and anomaly detection (An & Cho, 2015; Xu et al., 2018), and their ability to outperform Vanilla Auto-encoders and other simpler reconstruction models. In our method, a VAE model is trained with a semi-supervised one-class workflow (Goldstein & Uchida, 2016), exclusively using normal data representing the majority class, and therefore free of anomalies.

The VAE model is trained by optimizing the evidence lower bound (ELBO) as in the following equation:

$$L_{(\theta,\phi)} = \min E_{q_\phi(z|x)}[\ln p_\theta(x|z)] - D_{KL}[q_\phi(z|x) \parallel p(z)], \tag{1}$$

where $q_\phi(z|x)$ is the encoder, and $p_\theta(x|z)$ is the decoder. Maximizing the log-likelihood $\ln p(x|z)$ is equivalent to minimizing the mean squared error between the input x and its reconstruction $\hat{x}$ output by the decoder, and $D_{KL}[q(z|x) \parallel p(z)]$ is the KL-divergence between the distribution of the encoder and the prior on z. Due to the non-negativity of the KL divergence, the ELBO is a lower bound on the log-likelihood of the data (Kingma, Welling, et al., 2019). The training process allows to identify the optimized set of parameters ϕ and θ, which takes place through minimization of the loss function defined in Eq. (1) using Stochastic Gradient Descent (SGD).

In our method, our goal is to provide VAE with lifelong learning capabilities, adopting a custom workflow that features *Change Point Detection, Consolidation, Memory Summarization,* and *Experience Replay* mechanisms. We explain each of these mechanisms in the following subsections.

5.2. Lifelong change point detection

One limitation of existing lifelong learning methods in the classification setting is that the notion of changes between concepts (typically known as tasks or classes) is given as the ground truth in the learning setting. However, this represents a significant simplification in many real-world domains of interest for lifelong anomaly detection (see Section 2 for a more detailed discussion).

On the other hand, anomaly detection methods for continual learning settings are unaware of changes between concepts, and therefore they undergo a continuous update process as new data becomes available. This characteristic may expose them to catastrophic forgetting. For this reason, we argue that detecting changes between concepts is relevant to trigger model updates.

The ability to detect changes in a lifelong setting empowers our method with task-agnostic learning capabilities (meaning that task boundaries are not available). Therefore, the model can learn in a fully unsupervised manner, relying on the feedback provided by the change point detection method. However, using conventional change point detection approaches in lifelong scenarios would not provide a precise characterization of the type of change (e.g., new *vs.* recurring concept), but only that a generic change has occurred. Even assuming that the existing change point detection could leverage a memory data structure, they would be unable to differentiate between different types of changes. As a consequence, memory representations would not reflect the actual concept taxonomy, with some extremely underrepresented concepts and some overly represented concepts. Therefore, the model may suffer from forgetting when a previously observed concept is not observed for a long time or when certain tasks appear more frequently than others.

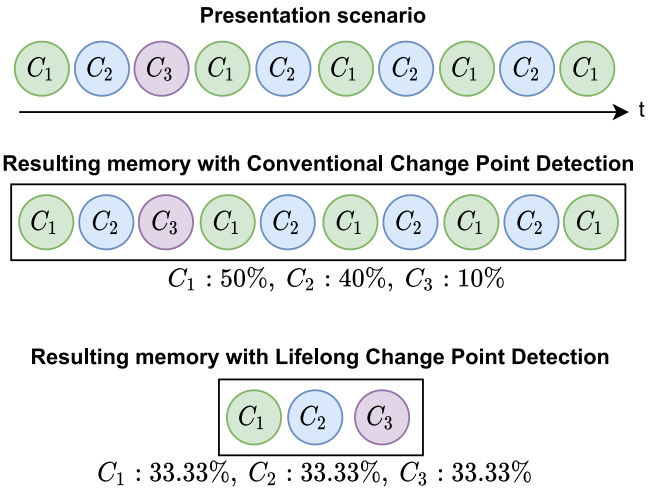


Fig. 5. Comparison of conventional change point detection and lifelong anomaly detection in terms of memory storage. The ability to characterize different types of changes in our lifelong change detection approach yields a balanced taxonomy for concepts observed over time. For each concept, we report their percentage of representation in memory.

In this work, we adopt lifelong change point detection as a particular form of change point detection where we are not simply interested in detecting time points where a generic change has occurred, but we need to discriminate between changes that describe transitions to new concepts and transitions to already seen concepts (Faber, Corizzo, Sniezynski, Baron and Japkowicz, 2022). Moreover, it is essential to leverage this feedback to recognize the newly observed data points as belonging to previously observed distributions. In principle, this capability should allow to overcome the issue of imbalanced memory representations highlighted above, which in turn may lead to model degradation and forgetting. A conceptual comparison of the two change point detection approaches in terms of their ability to handle this scenario is shown in Fig. 5. To the best of our knowledge, change point detection methods for lifelong learning settings have never been leveraged for anomaly detection before, and our `VLAD` represents the first effort in this respect.

Our component leverages the Wasserstein distance to compare the representation of newly processed data points with respect to a set of known distributions without making assumptions about any particular type of distribution in the analyzed data. For this reason, instead of modeling distributions in terms of their parameters (such as mean and variance) and distribution type, we focus on non-parametric change point detection using generic sets of points.[3] Some works in the literature provide a

[3] From now on, the term *distribution* is used to describe the change point detection process and the short-term memory, while the term *concept* is used

conceptual explanation of the Wasserstein distance by regarding the distribution as a unit amount of piled soil. In this visualization, the metric is considered the minimum "cost" of shifting one pile into another. This cost represents the amount of earth to be moved multiplied by the mean distance between the piles. A preliminary study (Faber, Corizzo, Sniezynski, Baron and Japkowicz, 2021) showed the potential of its adoption for timely and accurate identification of change points in high-dimensional datasets, and its advantages over other standard measures such as Kullback–Leibler divergence. A more detailed description of the Wasserstein distance is available in Faber, Corizzo et al. (2021) and Hallin, Mordant, and Segers (2021).

Our method learns an initial representation of an unknown distribution D_C in an unsupervised manner. Multivariate data is processed in mini-batches B_i of length C_ω. The algorithm stores data points that are collected over time (possibly from multiple batches) for the currently analyzed distribution in a buffer D_C. For new distributions, as soon as the buffer reaches the desired capacity C_κ, D_C is included in the emerging pool P, which defines a short-term memory. The condition that triggers this action can be controlled to align to domain specifics, such as task heterogeneity, by changing the value of the C_ϵ parameter.

A long-term hierarchical memory $\mathcal{M}$ is also accessed and updated by the component. All distributions existing in P and $\mathcal{M}$ are then leveraged in the change detection procedure. A threshold $E[D_j]$ for each distribution D_j (assigned to P or representing a concept C_j in $\mathcal{M}$) is calculated as:

$$E[D_j] = C_\epsilon \max_{B_i \in D_j} W_A(B_i, D_j). \tag{2}$$

From an intuitive standpoint, $E[D_j]$ allows to decide which data points should be included by comparing them with the representation of the current distribution. It is defined to resemble the largest distance computed across all mini-batches for a given distribution D_j, scaled by an C_ϵ factor. The threshold is periodically updated to reflect the characteristics of the distribution.

Initially, the change point detection algorithm compares the Wasserstein distance between B_i and D_C with the previously defined threshold $E[D_C]$. It decides if B_i can be considered as part of the current distribution D_C. If the distance is smaller than $E[D_C]$, the algorithm considers B_i as a part of D_C. Otherwise, when the distance is greater than $E[D_C]$, the algorithm detects a change point and proceeds to determine its nature (new or recurring).

If a change point has been detected, the next step is determining whether B_i qualifies as part of any of the already known distributions $D_j \neq D_C$. If this condition holds true, the change point is of recurring nature. The algorithm looks for the smallest ratio between the Wasserstein distance $W_A(B_i, D_j)$ and the defined threshold for the candidate distribution $E[D_j]$ in order to identify the nearest distribution D_k. When the distance from B_i to the nearest distribution D_k is smaller than $E[D_k]$, the algorithm considers B_i as part of D_k and shifts the reference to the current distribution, i.e., $D_C \leftarrow D_k$. Otherwise, if the distance from B_i to the nearest distribution D_k is greater than $E[D_k]$, the algorithm detects a new distribution, assigns it as the current, and adds it to the emerging pool ($P = P \cup D_C$). Following this process, the method determines which is the target distribution assigned for each mini-batch B_i.

Then, the algorithm extends the identified distribution with data from B_i, so that the freshly acquired data may be reflected.

Input: $S^{(t)} = (x_1, x_2, \ldots, x_{|S^{(t)}|})$ Batch of the experiences
P Set of the distributions from Emerging Pool
$\mathcal{M}$ Set of the distributions from Hierarchical Memory

Result: λ_N Flag informing whether any new tasks were detected,
λ_R Flag informing whether any recurring tasks were detected,
P Updated Emerging Pool,
$\mathcal{M}$ Updated Hierarchical Memory,

Parameters: C_κ Min points required in a distribution before starting detection,
C_ϵ Threshold ratio for the distance value,
C_ω Mini-batch size.

An operational pool of distributions: $D = P \cup \mathcal{M}$
Results variables: $\lambda_N \leftarrow False$, $\lambda_R \leftarrow False$
$D_C \leftarrow$ The reference to the current distribution or $\emptyset$
$B \leftarrow$ Process $S^{(t)}$ into sequence $(B_1, B_2, \ldots, B_{\frac{T}{C_\omega}})$ of non-overlapping mini-batches B_i of size C_ω
for $B_i \in B$ **do**
  if $|D_C| < C_\kappa$ **then** // Building new distribution
    $D_C \leftarrow D_C \cup B_i$
    if $|D_C| >= C_\kappa$ **then**
      Compute $E[D_C]$ from Eq. (2) with C_ϵ // Compute threshold for D_C
      $P \leftarrow P \cup D_C$ // Add D_C to the Emerging Pool
      $D \leftarrow D \cup D_C$ // Add D_C to the operational pool
    end
  else // Detection phase
    $\upsilon \leftarrow W_A(B_i, D_C)$ // Compute Wasserstein Distance
    if $\upsilon > E[D_C]$ **then**
      $D_k = \begin{cases} \operatorname{argmin}_{D_j \in D} \frac{E[D_j]}{W_A(B_i, D_j)}, & \text{if } \exists_{D_j \in D} W_A(B_i, D_j) < E[D_j] \\ \emptyset, & \text{otherwise} \end{cases}$
      if $D_k = \emptyset$ **then**
        $\lambda_N \leftarrow True$ // New task detected
      else
        $\lambda_R \leftarrow True$ // Recurring task detected)
      end
      $D_C \leftarrow D_k$ // Assign D_k as a current distribution D_C
    end
    $D_C \leftarrow D_C \cup B_i$ // Update current distribution from P or $\mathcal{M}$
    Periodically compute $E[D_C]$ from Eq. (2) with C_ϵ // Update threshold
  end
end
return λ_N, λ_R, P, $\mathcal{M}$

Algorithm 2: Pseudo-code of the Lifelong Change Point Detection

At the same time, the algorithm recalculates $E[D_C]$. Algorithm 2 presents the pseudo-code of the component.

It is worth noting that D_C can correspond to a recently observed distribution in the short-term memory P or a previously consolidated concept C_j in the long-term memory $\mathcal{M}$. This flexibility allows us to maintain an updated representation of the concepts in both P and $\mathcal{M}$. A schematic representation of the impact of lifelong change point detection workflow in memory is represented in Fig. 6.

The component also keeps track of whether at least one new distribution and whether at least one recurring distribution has been detected in the current batch $S^{(t)}$. This information is useful to trigger model updates and consolidation process.

We highlight a few critical and peculiar characteristics of the method that differentiates it from non-lifelong change point detection methods. First, it characterizes changes in two types of changes or transitions: new distributions, and recurring distributions, assigning incoming data points to distributions already in the memory. Second, it does not imply any assumptions on the distribution types since it leverages comparisons among sets of data points. Third, the method avoids excessive comparisons

to describe experiences stored in the long-term memory. While the two terms both refer to sets of experiences that are stored in the same way, they are both used to avoid abuse of notation, and to better showcase their conceptual meaning.

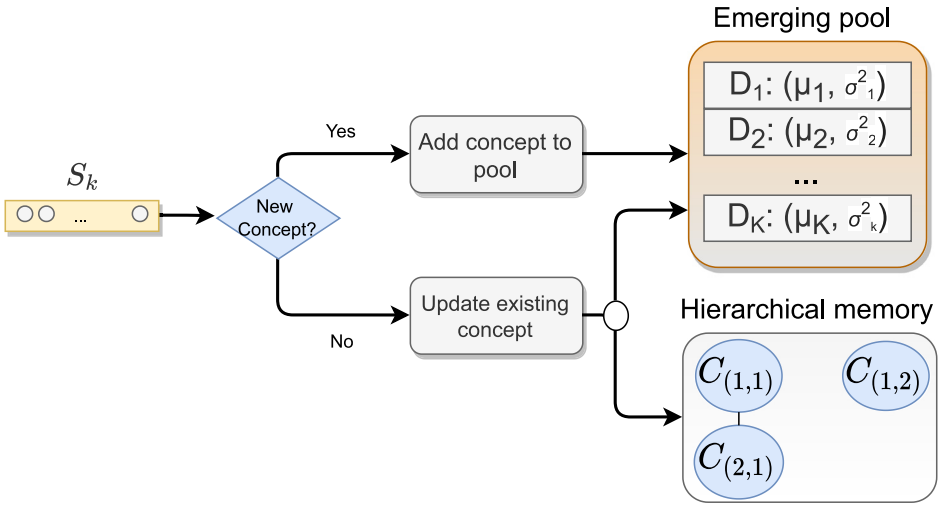


Fig. 6. The *Lifelong Change Point Detection* component adds concepts to the short-term memory (Emerging pool) and updates existing concepts in the short-term and in the long-term memory (Hierarchical memory).

of the distributions in the pool with freshly obtained data if it determines that new data belongs to the current distribution.

5.3. Consolidation

The Consolidation component is responsible for building, organizing, and retaining knowledge in the method. This capability is necessary for lifelong learning methods to retain past knowledge (stability) while incorporating new knowledge (plasticity). In our method, the knowledge is stored in a long-term hierarchical memory $\mathcal{M}$ inspired by the taxonomy hierarchy occurring in many real-world settings. In particular, our memory stores a forest of root level concepts and sub-concepts. Conceptually, the connotation of concepts and sub-concepts relates to the domain characteristics of the analyzed data. For example, in solar energy plants, different concepts may correspond to different production profiles (e.g., *day*: production - *night*: no production), whereas different sub-concepts may correspond to multiple conditions of day, which affect production (e.g., *sunny*: maximum production, *cloudy/rainy*: reduced production). In the network traffic domain, concepts may describe network traffic in *working days* vs. *weekends*, whereas sub-concepts may describe specific activities, such as *web browsing*, and *data transfers*. Our intuition is that properly structuring the memory in terms of concepts and sub-concepts as a hierarchy should yield a better use of its capacity, which in turn could reduce forgetting when exploiting experiences stored in memory to update models.

The hierarchical structure $\mathcal{M}$ organizes data points into a forest of concepts or sub-concepts. More formally, $\mathcal{M} = \{C_{(l,l')}\}$, where (l, l') represents the two-dimensional integer coordinates of the root concept or sub-concept in the hierarchy (l indicates the level of the hierarchy, and l' indicates a sequentially increasing concept identifier at that level). Intuitively, $C_{(1,*)}$ are root-level concepts, whereas $C_{(i,*)}, i > 1$ are sub-concepts. Fig. 7 shows a graphical representation of the two-dimensional mapping.

The concept consolidation strategy exploits the Wasserstein distance among data points. The rationale behind this strategy is that the Wasserstein distance margin between distributions corresponds to different semantic types of newly appearing concepts. New sub-concepts are identified by significant but modest variations from the known beginning distribution, whereas new concepts are identified by bigger distances.

This component takes $\mathcal{M}$ and P in their current state as inputs, and returns an updated version of $\mathcal{M}$ where new concepts from P are matched as new sub-concepts or new root level concepts in $\mathcal{M}$. For each distribution $C_{(l,l')} \in \mathcal{M}$, we determine the threshold $E'[C_{(l,l')}]$ using the previously calculated threshold $E[C_{(l,l')}]$ from the Lifelong Change Point Detection component (as described in Section 5.2), and the sub-concept threshold ratio W_ϵ, as in the following:

$$E'[C_{(l,l')}] = E[C_{(l,l')}] \cdot W_\epsilon \tag{3}$$

Then, we can use this threshold to identify the parent concept in the hierarchical memory $CD(D_k)$ for each distribution $D_k \in P$, as in the following formula:

$$CD(D_k) = \begin{cases} \arg\min_{C_{(l,l')} \in \mathcal{M}} \dfrac{E'[C_{(l,l')}]}{W_A(D_k, C_{(l,l')})}, & \text{if } \exists_{C_{(l,l')} \in \mathcal{M}} W_A(D_k, C_{(l,l')}) < E'[C_{(l,l')}] \\ \emptyset, & \text{otherwise} \end{cases} \tag{4}$$

This assignment criterion operates on a notion of relative distance, according to which a distribution with a small variance will require a lower threshold than a distribution with a larger variance to be assigned as the closest to the distribution currently being considered. If the first condition in Eq. (4) holds, D_k becomes a new sub-concept for $C_{(l,l')}$. Otherwise, if no match is found, D_k becomes a new root concept.

As soon as a distribution $D_k \in P$ is matched with a concept in $\mathcal{M}$, its representation is removed from P. Since we are iterating through the set of distributions learned so far, the *Consolidation* workflow leads to the removal of all distributions from P, since they are all identified as sub-concepts or root level concepts in the hierarchical memory. This process is regarded as Pool Reset.

The memory consolidation process is triggered if at least one change describing a transition to a new distribution has been identified by the *Lifelong Change Point Detection* component. A more detailed discussion about the impact of *Lifelong Change Point Detection* on the *Consolidation* component is presented in Appendix A.1.

5.4. Memory summarization

The fine granularity of data collected over time may lead to a quick saturation of memory resources due to the large number of experiences learned and stored in the hierarchical memory. Storing systematically all experiences in memory is infeasible and creates the need for memory summarization mechanisms

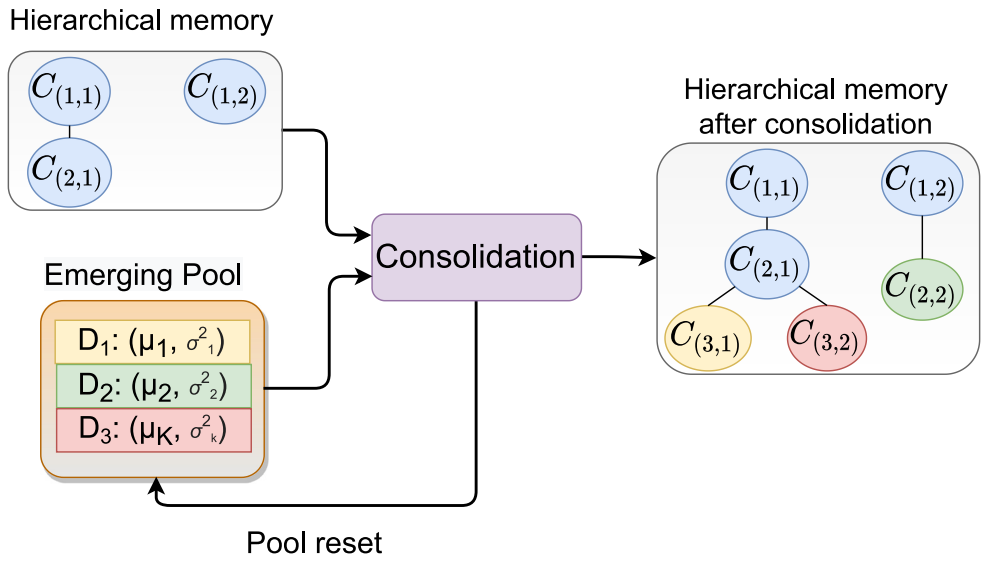


Fig. 7. The *Consolidation* component is responsible for organizing concepts and sub-concepts in the Hierarchical Memory, and for triggering the Pool Reset mechanism. It takes the Emerging Pool and the Hierarchical Memory in their current state, and returns an updated version of the Hierarchical Memory.

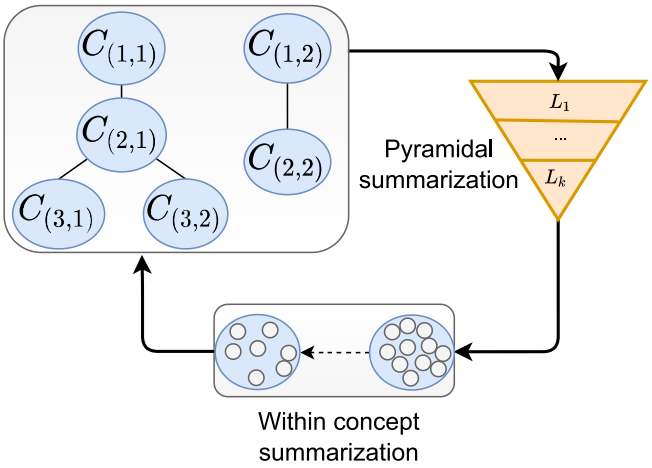


Fig. 8. The *Memory Summarization* component balances the granularity level of concepts through pyramidal summarization. This granularity is exploited during the within concept summarization process, that generalizes experiences in each concept. The resulting memory maintains the same hierarchy but balances the number of samples of each concept based on their level in the hierarchy.

to maintain lightweight data structures, so that they can be efficiently updated. In addition, generalization is necessary to summarize episodic experiences and generalize them, which is a crucial need for model longevity in lifelong learning settings.

This component is responsible for summarizing experiences stored in memory and works in synergy with the *Consolidation* component to maintain a lightweight and accurate knowledge representation. The memory summarization process also allows to prevent forgetting and model overfitting, which may lead to model degradation over time. A discussion about the interaction between the *Consolidation* and *Memory Summarization* components is provided in Appendix A.2.

In our method, the memory summarization process is triggered when the memory exceeds the desired upper bound M_B capacity by a factor $M_f > 1.0$, provided as an input parameter. A schematic representation of the component and its workflow is represented in Fig. 8. Inspired by the recent success of pyramid-based networks in object detection (Lin et al., 2017), a novel Pyramidal Summarization strategy is proposed to generalize stored experiences in the hierarchical memory. Specifically, the strategy guarantees an upper bound M_B that corresponds to the number of stored experiences. The rationale is that, given the hierarchical nature of our memory, root level concepts should be represented more broadly, i.e., with more samples, than sub-concepts. For this reason, a pyramid-based memory should allow to represent multi-scale information in the hierarchy, facilitating the retention of salient knowledge at different levels of granularity. An elastic memory adaptation is provided by shrinking the number of samples for each concept and sub-concept over time, as new experiences are collected. More specifically, the number of samples $M(C_{(l,l')})$ assigned to a given concept $C_{(l,l')}$ decreases by a factor of 2 for each subsequent level of the hierarchy, according to the following equation:

$$M(C_{(l,l')}) = \frac{1}{2^{(l-1)}} \cdot \frac{M_B}{\sum_{C_{(l,l')} \in \mathcal{M}} \frac{1}{2^{(l-1)}}}, \tag{5}$$

where M_B is the upper bound for the memory (maximum number of samples) defined in Section 4.2. We point out that since this component leverages what is stored in the hierarchical memory, an accurate hierarchy extracted by the consolidation process is crucial to ensure high-quality output for the memory summarization process, and that their successful interaction in `VLAD` is fruitful to reach the defined learning goals of the model.

Once the size of each concept $M(C_{(l,l')})$ is known, the actual within-concept summarization process takes place resorting to centroid-based summarization. This stage allows to detect sub-groups of similar experiences within each concept based on their similarity and, consequently, to store a limited number of prototypes.

The k-Means algorithm is used to partition a given set of experiences into a predefined amount of M_k prototypes for each concept $C_{(l,l')}$. The algorithm starts with a random set K of center points (k). During each update step, all experiences $x \in C_{(l,l')}$ are assigned to their nearest center-point In the standard algorithm, only one assignment to one center is possible. If multiple centers have the same distance to the experience, a random one would be chosen. Afterwards, the center-points are repositioned by calculating the mean of the experiences X_i assigned to the respective center points (see Eq. (6)).

$$k'_i = \frac{1}{|X_i|} \sum_{x_j \in X_i} x_j \tag{6}$$

The process tries to optimize the objective function in Eq. (7). As there is only a finite number of possible assignments for the number of centroids and experiences available, the algorithm always ends in a local minimum (Bottou & Bengio, 1994).

$$J = \sum_{n=1}^{|C_{(l,l')}|} \sum_{m=1}^{|K|} r_{nm} \|x_n - k_m\|^2 \tag{7}$$

$$\text{with } r_{nm} = \begin{cases} 1 & x_n \in X_m \\ 0 & \text{otherwise,} \end{cases}$$

where $X_m \subset C_{(l,l')}$ is a set of data points assigned to the cluster m. Once cluster assignments for experiences are known, a subset of experiences is retained for each cluster. In particular, we retain the set E_{k_i} of the closest n experiences to each centroid $k_i \in K$ for each cluster, formalized in a declarative definition as:

$$E_{k_i} = \{x \mid dist(x, k_i) \leq \min_{x' \in X_i \setminus E_{k_i}} dist(x', k_i)\} \tag{8}$$

The size of each cluster is obtained as the ratio between $M_{C_{(l,l')}}$ calculated according to Formula (5) and the number of centroids M_k. This approach will avoid storing experiences that are too similar and ensure the diversity of experiences in each concept.

5.5. Experience replay

Although periodic updates are necessary to allow the VAE model to adapt as new data is observed, another desired characteristic is to retain knowledge acquired in the past. One typical drawback of model updates is being prone to the forgetting phenomenon, which implies a decrease in the model performance for prior seen concepts as new knowledge is acquired. Our method aims to prevent this phenomenon by leveraging an Experience Replay component, allowing to achieve a stability–plasticity trade-off. An intuitive explanation with a visual demonstration of this behavior is shown in Appendix A.3. In our method, instead of adopting a flat replay buffer, as commonly done in the literature, the adoption of a hierarchical memory and Pyramidal Summarization ensures that the importance of experiences is re-scaled as new tasks are processed, and properly represented by the model, as shown in Fig. 9. This idea is inspired by known results in the fields of neuroscience and biology, which highlight hierarchical principles of neural organization (Berntson, Boysen, & Cacioppo, 1993; Sherrington, 1952; Tinbergen, 1950).

This component takes the most updated state of the memory $\mathcal{M}$ and creates a new replay buffer R each time it is required. The replay buffer consists of all experiences from all concepts stored in the memory according to the following equation:

$$R = \bigcup_{C_{(l,l')} \in \mathcal{M}} \{x \mid x \in C_{(l,l')}\} \tag{9}$$

The VAE model is updated resorting to R, as well as the most recently observed batch of data $S^{(t)}$, by means of concatenation. The model is updated in the following three circumstances:

- *A change is detected*: New samples starting from the change point are stored in the Emerging Pool provided by the *Lifelong CPD* component. Once a sufficient number of data points C_κ is available, the model is retrained. This aspect is important since it helps to mitigate overfitting issues in the VAE model.
- *No update in a long time*: Two cases are possible: there was actually no change in the data distribution, or a change occurred and its detection was missed. In both circumstances, model update is triggered after a desired number of data points are processed (considering the last time the model was updated), as it is beneficial to reflect the new data characteristics. This behavior is controlled by a R_ψ parameter.
- *Memory summarization was performed*: This behavior takes place when the hard upper bound of memory is reached. Once the pyramidal summarization process is done, model update is explicitly triggered to reflect the most recent characterization of experiences in terms of concepts and sub-concepts in the memory.

5.6. Anomaly detection

After training the VAE model using normal data, newly available data x is processed by the model, which provides a reconstruction $\hat{x}$, as shown in Fig. 4. An anomaly score $a(x)$ is provided by the model as follows:

$$a(x) = \frac{1}{F} \sum_{i=1}^{F} (x_i - \hat{x}_i)^2, \tag{10}$$

where F is the number of input features (dimensions).

A static threshold to anomaly scores is usually applied to determine whether x belongs to the normal or the anomaly class. Instead, our method provides an anomaly score as a more flexible solution without information loss, allowing the user the discretion to analyze cases separately and make a final decision. A specific threshold can be applied as a post-processing step according to the domain characteristics of data and to match application requirements.

For instance, maximizing True Positive Ratio (TPR) could be more important than having a balanced setting in domains where false positives can be easily discarded by a human expert, or where the cost of a false negative would be too high. A balance between TPR and False Positive Ratio (FPR) may be desired in automated settings where the model takes decisions on its own. Finally, a low desired value for FPR might be suitable in scenarios such as data cleaning, where obvious outliers or rare data points, as well as scenarios where a large number of false positives would not be manageable for domain experts involved in the post-prediction decision process.

6. Experimental setup

Our experiments aim to answer the following research questions:

- **RQ1:** Does our method achieve a **higher anomaly detection performance on all learned tasks than state-of-the-art anomaly detection methods** in task-agnostic scenarios where multiple concepts of the normal class are presented over time?
- **RQ2:** Does our method **mitigate forgetting in task-agnostic scenarios** by improving backward transfer while preserving a robust anomaly detection performance on previously learned tasks when compared to other methods?
- **RQ3:** Does our method provide an accurate memory representation in terms of the **number of concepts extracted in long scenarios with recurring tasks**?
- **RQ4:** Is it possible to achieve a **high anomaly detection performance with a low memory footprint** in terms of the number of stored experiences?
- **RQ5:** Does the synergic combination of all components contribute to **achieving a better anomaly detection performance** than that provided by any of the components in isolation?

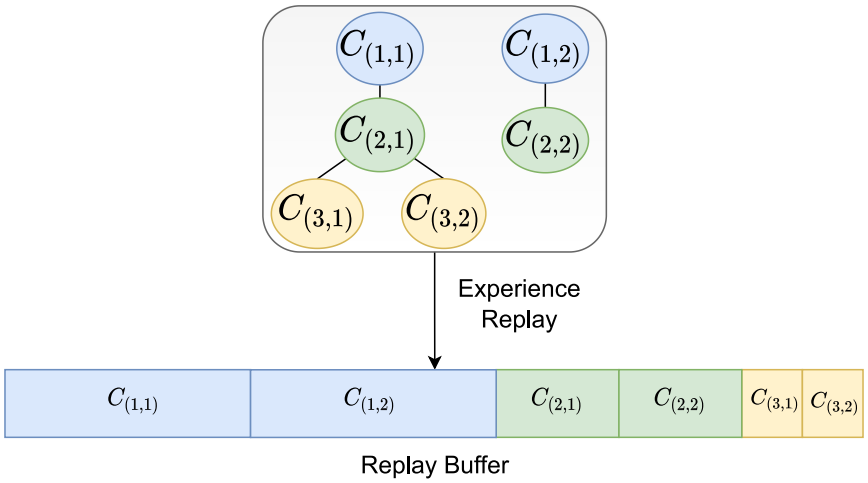


Fig. 9. The *Experience Replay* component takes the most updated state of the memory $\mathcal{M}$ and creates a new replay buffer R each time it is required. The memory structure is built upon consolidation, and concepts at different levels are represented with a different number of samples according to the outcome of pyramidal summarization, which provides a proper balance of concepts in the replay buffer.

To answer these questions, we perform a set of quantitative and qualitative experiments that assess the performance of VLAD from different perspectives. To explicitly link our research questions to our paper contributions devised in the Introduction section, **RQ1**, **RQ2**, and **RQ5** support us in the assessment of our method's effectiveness when solving crucial challenges for the lifelong anomaly detection problem, as well as its ability to timely identify task changes. **RQ3** and **RQ4** give us the opportunity to assess the quality of our proposed experience replay-based consolidation and summarization algorithms. All RQs encompass the evaluation and analysis of the proposed method on real-world domains, whereas the newly proposed evaluation scenario is directly used in all our experiments and analyses.

In the following subsections, we describe the evaluation protocol followed in our study, including metrics used for anomaly detection and lifelong learning, datasets adopted and learning scenarios addressed, as well as competitor methods considered and their configuration. Results are discussed in Section 7.

6.1. Evaluation protocol

In our experiments, we follow the standard evaluation protocol used in many lifelong learning studies (Díaz-Rodríguez, Lomonaco, Filliat, & Maltoni, 2018), which assumes access to a set of training and testing tasks. However, one key difference is that the model is not provided with task label information, nor task boundaries, resulting in a more challenging task-agnostic learning scenario as described in Section 2. In our experiments, the sequence of training tasks T is presented to the model in the form of batches $S^{(t)}, S^{(t+1)}, \ldots, S^{(t+k)}$ of fixed length. To ensure a fair comparison, all models are continuously updated each time a new batch is presented. After the model processes all batches from each learning task $T_i \in T$, it is evaluated on all testing batches ($E = \{E_1, E_2, \ldots, E_N\}$), $|E| = |T|$ which allow us to assess the model's ability to handle the last learned task, as well as previously learned tasks, and tasks observed in the future.

In this way, we build a matrix $R_{N\times N}, N = |T|$ containing evaluation results. $R_{i,j}$ contains results in terms of the ROC–AUC metric obtained on task j after learning task i. This process is formalized in Algorithm 3. Once the matrix with results has been filled, it is possible to compute lifelong learning metrics. More details on our rationale for the choice of these metrics, as well as the process of computing lifelong learning metrics, are provided in the following subsection. It is noteworthy that, even if the evaluation part is carried out after processing all batches from each task, the models are not informed about the transition between tasks.

Input: **T** – a sequence of N training tasks
Input: **E** – a sequence of N testing batches
Input: L – learning model under evaluation
Initialize results matrix $R_{N\times N}$
for $T_i \in$ **T do**
  for $S^{(k)} \in T_i$ **do**
    Train L on $S^{(k)}$
  end
  for $E_j \in$ **E do**
    Compute ROC–AUC for model L given data E_j
    Save results to $R_{i,j}$
  end
end
return R

Algorithm 3: Pseudo-code of the Evaluation Protocol followed in our experiments

6.2. Metrics

Anomaly detection

The most common metric adopted in lifelong learning works is Accuracy. However, even if accuracy is a good choice for image classification, it is not well-suited for many anomaly detection applications due to class imbalance (Branco, Torgo, & Ribeiro, 2016; Chawla, Japkowicz, & Kotcz, 2004; Provost & Fawcett, 2001). Metrics that are commonly used for anomaly detection include Precision, Recall, and F1. While Precision emphasizes the number of correctly detected anomalies over the number of detected anomalies, Recall measures the number of correctly detected anomalies over the total number of anomalies existing in the data. F1 is the harmonic mean of Precision and Recall and takes into consideration both false positives and false negatives, providing an easy way to compare the models. They are all well suited to answer essential questions about the performance of the anomaly detector. However, these metrics depend on single thresholds, which force the user to choose a specific value that divides the dataset into positively and negatively predicted classes. This decision reflects a trade-off that is difficult for the user to estimate beforehand and is subject to change over time (Fourure, Javaid, Posocco, & Tihon, 2021). A more feasible and robust solution is

using threshold-free measures that assess model-wide performance, such as Receiver Operating Characteristic Curve (ROC). Threshold-free metrics require the anomaly detector to provide an anomaly score, which is universal and can represent different observable properties, varying from the probability of classification to the reconstruction error. This characteristic is in line with the great majority of recent anomaly detectors, which can provide such values. A ROC plot shows the trade-off between True Negative Rate (TNR) and True Positive Rate (TPR) at all possible threshold values. In this work, we adopt the Area Under the ROC curve (ROC–AUC) as our primary anomaly detection performance metric. It is noteworthy that ROC–AUC has a clear value for a random classifier: 0.5, making it easier to assess the abilities of a model to perform better than a random guess.

Lifelong learning

Previous works in the literature focused on the definition of lifelong learning metrics that properly describe the ability of a learner to handle various key challenges (Díaz-Rodríguez et al., 2018; Lopez-Paz & Ranzato, 2017; New, Baker, Nguyen, & Vallabha, 2022). Appropriate metrics should be able to assess models on the following relevant properties:

- Ability to continuously learn from incoming data;
- Retaining knowledge of previously learned tasks;
- Transferring knowledge from previously learned task to new ones;
- Keeping a low memory footprint.

The results matrix $R_{N\times N}$ mentioned above is built during the evaluation phase. Its entries contain ROC–AUC values. We follow the definition in Díaz-Rodríguez et al. (2018), as it is widely adopted and presents the important advantage of considering results obtained during the whole learner's lifetime.

Lifelong ROC–AUC (*ROC-AUC*)[4] – It measures the average anomaly detection performance of the model after learning each task, for all previously learned tasks as well as the current task.

$$ROC\text{-}AUC = \frac{\sum_{i\geq j}^{N} R_{i,j}}{\frac{N(N+1)}{2}} \tag{11}$$

Backward Transfer for ROC–AUC (*BWT*) – It measures the influence of learning new tasks on the performance on all previously learned tasks. Positive backward transfer indicates that learning new tasks improves performance on previously learned tasks. On the other hand, a negative value for backward transfer is known as forgetting. A strongly negative value is also sometimes regarded as catastrophic forgetting. Backward transfer is computed over all tasks, as in the following equation:

$$BWT = \frac{\sum_{i=2}^{N}\sum_{j=1}^{i-1} R_{i,j} - R_{j,j}}{\frac{N(N-1)}{2}} \tag{12}$$

Forward Transfer for ROC–AUC (*FWT*) – It measures the overall influence that learning each task has on the performance of tasks learned in the future. The forward transfer is computed over all tasks, as in the following equation:

$$FWT = \frac{\sum_{i<j}^{N} R_{i,j}}{\frac{N(N-1)}{2}} \tag{13}$$

6.3. Datasets and scenarios

Our experiments feature the following datasets:

- **NSL-KDD.** Tavallaee, Bagheri, Lu, and Ghorbani (2009) The dataset is an improved version of the commonly used network-based intrusion dataset KDD-99, proposed to solve some of the inherent problems of its predecessor. It contains records of network traffic gathered during the DARPA Intrusion Detection Systems evaluation program. The categorical features from the original features (such as protocol name) were encoded to ordinal values.
- **UNSW-NB15.** Moustafa and Slay (2015) The dataset contains records of network traffic describing hybrid of the real modern normal and the contemporary synthesized attack activities. The categorical features from the features provided by the authors of the dataset were encoded using as an integer array leveraging ordinal encoder.
- **ADFA-LD.** Creech and Hu (2013) The dataset contains data gathered from normal operations of an Ubuntu Linux system (version 11.0). Anomalies belong to six attacks families (Adduser, Hydra-FTP, Hydra-SSH, Java Meterpreter, Meterpreter, Web-Shell). The dataset is designed to be evaluated for host-based intrusion detection.
- **NGIDS-DS.** Haider, Hu, Slay, Turnbull, and Xie (2017) The dataset was designed to reflect the contemporary security infrastructure and challenges. Anomalies belong to seven attack families, including exploits, worms and backdoors. Data covers a time frame of over 5 days and includes various enterprises challenges.
- **WWW2019.** Li et al. (2019) This dataset contains system-call traces from Windows and Linux host machines. Anomalies belong to multiple families of attacks, including malicious exfiltration of data, password theft and automated scans. Normal data includes various user activities.
- **WIND.** Corizzo, Ceci, Pio, Mignone, and Japkowicz (2021) This dataset was modeled using the Weather Research & Forecasting (WRF) model.[5] Five plants with the highest rated production have been selected, obtaining the time series of wind speed and production observed every 10 min, for a time period of two years (from January 1st, 2005 to December 31st, 2006). Hourly aggregation was performed.

In order to analyze lifelong learning capabilities of the models, we cannot simply use the available data in the provided format. On the other hand, simply partitioning it using a random split procedure would lead to tasks that are extremely similar between each other and contain heterogeneous data that describe the whole dataset, which defies the scope of the problem of interest in this study. For this reason, we devise scenarios that provide a clear and meaningful separation into tasks, and comply with the characteristics defined in Section 2. Therefore, we adopt the following procedure:

1. Data from the normal class is clustered into the desired number of tasks N using a centroid-based clustering algorithm such as k-means.
2. Anomalies are clustered into the desired number of micro-clusters of anomalies N, which corresponds to the number of tasks.
3. Each cluster of anomalies is incorporated to the closest task from the normal class.
4. Each task is separated into training and testing batches.

The procedure allows us to provide a partial initial knowledge to the model, and present new tasks over time, assessing model's ability to adapt and incorporate new tasks while retaining knowledge of previously learned tasks. We clarify that data is presented sequentially to the model without information about task boundaries and task labels (task-agnostic scenario), which

[4] For simplicity, in our experimental results, we refer to the Lifelong *ROC-AUC* metric as *ROC-AUC*.

[5] https://www.nrel.gov/wind/.

means that methods do not exploit the clustering information defined in scenarios.

We created scenarios for NSL-KDD, UNSW, NGIDS, and WIND following the specified procedure. In order to create a very challenging scenario, we combined three different host-based intrusion detection data sets together (NGIDS, WWW, and ADFA-LD) in a new data set called 3IDS. For this purpose, we created tasks for each dataset independently, and then combined them to create the final scenario. We consider 3IDS the most challenging scenario in our evaluation, as it presents a large heterogeneity of tasks originating from multiple datasets.

6.4. Methods and configuration

The rationale for the selection of competitor methods included standard methods known in the anomaly detection community and extensively used in previous works, the diversity of the methods, and their high potential in anomaly detection performance shown in recent studies with complex data. To guarantee a fair comparison, all models are updated each time a new batch is presented.

- *VAE*: We consider a basic VAE since it constitutes the core inference model of VLAD. Considering it in our experiments allows us to assess the merit of the lifelong learning-oriented components and their added value in terms of model performance over the baseline model. To fine tune the VAE, we set the number of neurons in the hidden layer and the dimensionality of z to the following possible pairs: $(64-16, 48-8, 32-8, 16-8, 16-4, 8-4)$.
- *IF*: Its tree and ensemble-based modeling capabilities showed great potential in anomaly detection works. For its optimization, we consider two high values for the number of base estimators in the ensemble $\{n\_estimators = (100, 200)\}$.
- *OC-SVM*: A well-recognized hyperplane-based method in anomaly detection due to its ability to detect out-of-distribution data points considering their distance from the decision boundary. We set the coefficient of the Radial Basis Function (RBF) kernel $gamma = (0.1, 0.001)$ and $nu = (0.1, 0.01)$ corresponding to the upper bound of the fraction of training errors and the lower bound of the fraction of support vectors, respectively.
- *LOF*: Its simple nearest-neighbor-based approach was shown to outperform more complex methods in the literature, making it a proper baseline worth consideration. We chose the number of neighbors to use for k-neighbors queries $n\_neighbors = (2, 5, 10)$; Minkowski measure as a distance measure.
- *COPOD*: The Copula-based capabilities of the method allow to predict the degree of "extremeness" of data samples based on tail probabilities. We consider it in the experimental pool since it has shown potential in recent research works on online anomaly detection in challenging domains. Its optimization process is straightforward, given the rather parameter-less nature of the method.
- *SUOD*: The heterogeneity of anomaly detectors in the method ensemble provides a competitive edge for complex data characterized by high-dimensionality and noise. Its optimization process is straightforward, given the rather parameter-less nature of the method.

It is worth noting that, since we use ROC–AUC as an evaluation metric, we overcome the burden of optimizing the value for the *contamination* parameter for methods that require it (i.e., IF, COPOD, and SUOD), since no decision function based on a threshold is used to extract predictions. For all methods, for all other parameters not mentioned above, we follow the recommended guidelines or default values provided in the SKLearn (Pedregosa et al., 2011) and PyOD (Zhao, Nasrullah, & Li, 2019) documentation, or recommended in reference papers of the methods, where suitable.

The execution of the experiments features the hyperparameter optimization process on the first task T_1 for all methods. Specifically, we train the model on T_1 and validate the performance of the models on the corresponding batch E_1. The rationale for this choice follows the idea of assessing models in the lifelong learning setting, where models are required to incrementally learn and adapt to new tasks that were previously unknown. However, it is fair to assume partial knowledge of the domain, i.e., the first task. This assumption is in line with the conventional model training procedure for anomaly detection, which requires the availability of a portion of normal data for model training, and a set of anomalies for validation.

For VLAD, we use the same VAE architecture and method configuration described above. For the Lifelong Change Point Detection component, we adjust the threshold ratio C_ϵ according to the internal cohesion of the distribution from the first task. The soft upper bound M_B is chosen from the following list of values: (250, 500, 1000, 2000, 4000). We present a detailed analysis of the impact of memory size on the anomaly detection performance in Section 7.4. For the Experience Replay component, we select R_ψ from (10 000, 15 000, 30 000) following the idea that this size should be at least twice the number of data samples observed in the first task. As for the other parameters, we used insights from a separate analysis of the hyperparameter values impact on performance presented in Section 7.6.2 to guide our experimental setting. Therefore, we leverage values of $C_\kappa = 1024$, $C_\omega = 4$, $M_f = 5$, $M_k = 5$.

7. Experimental results and discussion

In our analysis, we consider ROC–AUC as a reference metric for anomaly detection performance, and BWT as a reference metric for knowledge retention. In particular, negative backward transfer is an indication that the model presents forgetting. A strongly negative value is an indication of catastrophic forgetting. Among the two methods, we attribute the largest forgetting to the method with the lowest value of BWT. In the following subsections, we use these metrics to compare the performance of all methods in all scenarios. An important consideration is that, while some methods achieve higher values of BWT and FWT, these results should be put in relationship with the base metric (ROC–AUC). For instance, while it is true that other methods can achieve moderate or high values of BWT and FWT, its very low and close-to-random ROC–AUC performance invalidates the usefulness of these metric values. Therefore, high values of BWT and FWT are significant only when paired with high values of ROC–AUC.

7.1. Anomaly detection performance

The results in Table 1 show the performance in terms of ROC–AUC, BWT, and FWT (average and standard deviation) for all 5 analyzed scenarios. In general, it is possible to observe that VLAD achieves the highest performance in terms of ROC–AUC.

In the following analysis, we focus on comparing the anomaly detection performance of VLAD on all learned tasks in task-agnostic scenarios where multiple concepts of the normal class are presented over time (**RQ1**). We compare VLAD with the core model counterpart used in our method (VAE) to show that the lifelong learning capabilities proposed in VLAD are able to improve the anomaly detection performance by a significant

Table 1
Anomaly detection results in terms of Area Under the ROC Curve (ROC–AUC), Backward Transfer (BWT), and Forward Transfer (FWT) for all methods with all datasets. The results are averaged across different folds (*K-Folds*).

	NGIDS			3IDS		
	ROC–AUC	BWT	FWT	ROC–AUC	BWT	FWT
IF	0.615 ± 0.032	−0.489 ± 0.048	0.389 ± 0.025	0.628 ± 0.010	−0.433 ± 0.012	0.469 ± 0.038
LOF	0.634 ± 0.015	−0.417 ± 0.022	0.367 ± 0.021	0.530 ± 0.019	−0.483 ± 0.035	0.451 ± 0.158
OC-SVM	0.599 ± 0.004	−0.526 ± 0.006	0.227 ± 0.007	0.594 ± 0.003	−0.350 ± 0.004	0.158 ± 0.008
SUOD	0.583 ± 0.026	−0.541 ± 0.037	0.373 ± 0.030	0.618 ± 0.010	−0.426 ± 0.019	0.416 ± 0.080
COPOD	0.685 ± 0.004	−0.171 ± 0.017	0.432 ± 0.002	0.337 ± 0.001	−0.103 ± 0.001	0.390 ± 0.002
VAE	0.583 ± 0.003	−0.558 ± 0.010	0.231 ± 0.010	0.597 ± 0.007	−0.464 ± 0.011	0.312 ± 0.014
VLAD-NoCPD	0.749 ± 0.004	−0.033 ± 0.006	0.182 ± 0.005	0.569 ± 0.009	0.001 ± 0.003	0.456 ± 0.018
VLAD-NoReplay	0.591 ± 0.009	−0.546 ± 0.011	0.232 ± 0.004	0.564 ± 0.060	−0.330 ± 0.198	0.355 ± 0.052
VLAD	0.777 ± 0.009	−0.163 ± 0.006	0.179 ± 0.007	0.779 ± 0.024	−0.108 ± 0.021	0.426 ± 0.021
	NSL-KDD			UNSW		
	ROC–AUC	BWT	FWT	ROC–AUC	BWT	FWT
IF	0.745 ± 0.008	−0.260 ± 0.008	0.809 ± 0.033	0.533 ± 0.020	−0.423 ± 0.021	0.536 ± 0.035
LOF	0.610 ± 0.010	−0.323 ± 0.032	0.545 ± 0.058	0.719 ± 0.048	−0.292 ± 0.060	0.678 ± 0.016
OC-SVM	0.722 ± 0.002	−0.265 ± 0.004	0.858 ± 0.008	0.740 ± 0.016	−0.288 ± 0.019	0.722 ± 0.004
SUOD	0.683 ± 0.020	−0.341 ± 0.029	0.701 ± 0.032	0.581 ± 0.016	−0.403 ± 0.022	0.479 ± 0.013
COPOD	0.452 ± 0.002	−0.128 ± 0.001	0.268 ± 0.001	0.361 ± 0.001	−0.115 ± 0.002	0.309 ± 0.004
VAE	0.700 ± 0.016	−0.311 ± 0.031	0.877 ± 0.023	0.668 ± 0.018	−0.359 ± 0.022	0.689 ± 0.010
VLAD-NoCPD	0.818 ± 0.010	−0.040 ± 0.023	0.883 ± 0.018	0.802 ± 0.033	0.113 ± 0.016	0.552 ± 0.044
VLAD-NoReplay	0.693 ± 0.019	−0.312 ± 0.025	0.865 ± 0.014	0.664 ± 0.010	−0.363 ± 0.010	0.693 ± 0.008
VLAD	0.880 ± 0.013	−0.056 ± 0.010	0.915 ± 0.017	0.910 ± 0.015	−0.037 ± 0.011	0.539 ± 0.028
	WIND					
	ROC–AUC	BWT	FWT			
IF	0.842 ± 0.006	−0.147 ± 0.009	0.748 ± 0.010			
LOF	0.802 ± 0.007	−0.268 ± 0.012	0.808 ± 0.019			
OC-SVM	0.876 ± 0.000	−0.161 ± 0.001	0.874 ± 0.000			
SUOD	0.802 ± 0.015	−0.239 ± 0.024	0.696 ± 0.014			
COPOD	0.927 ± 0.001	−0.012 ± 0.001	0.935 ± 0.002			
VAE	0.858 ± 0.008	−0.167 ± 0.011	0.847 ± 0.009			
VLAD-NoCPD	0.896 ± 0.015	−0.111 ± 0.023	0.852 ± 0.006			
VLAD-NoReplay	0.858 ± 0.012	−0.166 ± 0.018	0.848 ± 0.006			
VLAD	0.959 ± 0.001	−0.014 ± 0.003	0.843 ± 0.008			

amount, allowing the model to perform successfully in a task-agnostic lifelong anomaly detection problem. We also compare VLAD to the best competitor method for each analyzed scenario in order to measure the minimum amount of improvement achieved by VLAD over competitor methods considered in our experiments.

- *NGIDS*: VLAD achieves an anomaly detection performance of 0.777, presenting a 33.28% improvement over VAE (0.583), and a 13.43% improvement over COPOD (0.685), the best competitor method. Other methods perform slightly above random guess, showing that having a changing normal class represents a challenging scenario that leads to forgetting and overall affects negatively anomaly detection performance.
- *NSL-KDD*: VLAD achieves an anomaly detection performance of 0.880, presenting a 25.71% improvement over VAE (0.700), and an 18.12% improvement over IF (0.745), the best competitor method. Other methods, such as SUOD and OC-SVM, are slightly worse than IF, whereas COPOD performs below random.
- *UNSW*: VLAD achieves an anomaly detection performance of 0.910, presenting a 36.23% improvement over VAE (0.668), and a 22.97% improvement over OC-SVM (0.740), the best competitor method. Overall, all other methods except IF perform either slightly above or below random, which shows the inability of other methods to adapt and evolve in long scenarios where the normal class changes its characteristics over time, resulting in very low anomaly detection performance.
- *3IDS*: VLAD achieves an anomaly detection performance of 0.779, presenting a 30.49% improvement over VAE (0.597), and a 24.04% improvement over IF (0.628), the best competitor method. We consider this scenario the most challenging addressed by our experiments since it entails the largest diversity of system calls, with complex evolution between tasks that encompass multiple datasets. Is it noteworthy that, in these conditions, the ensemble approach followed by SUOD achieves very close results to IF and OC-SVM, while other methods, such as COPOD and LOF, perform substantially poorly.
- *WIND*: VLAD achieves an anomaly detection performance of 0.959, presenting an 11.77% improvement over VAE (0.858), and a 3.45% improvement over COPOD (0.927), the best competitor method. It is noteworthy that, even if the VAE base model performs worse than COPOD, the combined effort of our components allows us to outperform all other competitor methods. In this scenario, although almost all methods are able to partially handle the evolving weather conditions of the wind plants (possibly due to the high similarity between tasks), achieving ROC–AUC values around 0.8, VLAD is able to yield a significant improvement over all other methods.

The results in Table 2 are obtained with 5 different randomly sampled task orderings for all considered scenarios. The goal of these experiments is to offer an additional perspective on the stability of the results for the different methods. In fact, we are particularly interested in assessing the average performance of models not only on different folds (as seen in Table 1, denoted as *K-Folds*), but also in terms of different presentation orderings (denoted as *Orderings*). The rationale is that since lifelong learning systems can be affected by the presentation order of

Table 2
Anomaly detection results in terms of Area Under the ROC Curve (ROC–AUC), Backward Transfer (BWT), and Forward Transfer (FWT) for all methods on all datasets. The results are averaged across different randomly sampled task orderings (*Orderings*).

	NGIDS			3IDS		
	ROC–AUC	BWT	FWT	ROC–AUC	BWT	FWT
IF	0.616 ± 0.025	−0.481 ± 0.047	0.487 ± 0.049	0.611 ± 0.032	−0.480 ± 0.055	0.493 ± 0.080
LOF	0.606 ± 0.042	−0.465 ± 0.070	0.406 ± 0.062	0.626 ± 0.069	−0.328 ± 0.114	0.552 ± 0.103
OC-SVM	0.545 ± 0.055	−0.594 ± 0.075	0.315 ± 0.082	0.485 ± 0.082	−0.382 ± 0.379	0.380 ± 0.251
SUOD	0.566 ± 0.036	−0.570 ± 0.061	0.404 ± 0.026	0.590 ± 0.032	−0.485 ± 0.061	0.495 ± 0.064
COPOD	0.630 ± 0.063	−0.241 ± 0.062	0.517 ± 0.093	0.365 ± 0.116	−0.138 ± 0.038	0.346 ± 0.174
VAE	0.536 ± 0.053	−0.611 ± 0.071	0.339 ± 0.078	0.569 ± 0.057	−0.535 ± 0.053	0.359 ± 0.085
VLAD-NoCPD	0.720 ± 0.053	−0.094 ± 0.052	0.302 ± 0.109	0.577 ± 0.075	0.001 ± 0.004	0.447 ± 0.081
VLAD-NoReplay	0.529 ± 0.058	−0.623 ± 0.078	0.325 ± 0.083	0.561 ± 0.045	−0.528 ± 0.045	0.370 ± 0.076
VLAD	0.779 ± 0.026	−0.184 ± 0.030	0.307 ± 0.107	0.791 ± 0.045	−0.086 ± 0.044	0.443 ± 0.117
	NSL-KDD			UNSW		
	ROC–AUC	BWT	FWT	ROC–AUC	BWT	FWT
IF	0.771 ± 0.027	−0.253 ± 0.028	0.776 ± 0.018	0.559 ± 0.043	−0.361 ± 0.087	0.464 ± 0.031
LOF	0.585 ± 0.041	−0.365 ± 0.070	0.525 ± 0.052	0.751 ± 0.054	−0.259 ± 0.075	0.741 ± 0.066
OC-SVM	0.774 ± 0.044	−0.221 ± 0.033	0.803 ± 0.057	0.749 ± 0.052	−0.282 ± 0.070	0.743 ± 0.064
SUOD	0.756 ± 0.025	−0.270 ± 0.026	0.709 ± 0.016	0.564 ± 0.048	−0.411 ± 0.081	0.479 ± 0.041
COPOD	0.423 ± 0.050	−0.099 ± 0.031	0.310 ± 0.064	0.380 ± 0.025	−0.078 ± 0.052	0.291 ± 0.031
VAE	0.757 ± 0.081	−0.256 ± 0.084	0.786 ± 0.087	0.687 ± 0.044	−0.342 ± 0.067	0.664 ± 0.073
VLAD-NoCPD	0.885 ± 0.049	0.015 ± 0.064	0.768 ± 0.075	0.776 ± 0.036	0.166 ± 0.056	0.516 ± 0.135
VLAD-NoReplay	0.757 ± 0.068	−0.258 ± 0.070	0.775 ± 0.071	0.686 ± 0.055	−0.348 ± 0.075	0.688 ± 0.046
VLAD	0.917 ± 0.027	−0.012 ± 0.028	0.772 ± 0.114	0.892 ± 0.023	−0.031 ± 0.059	0.498 ± 0.169
	WIND					
	ROC–AUC	BWT	FWT			
IF	0.835 ± 0.013	−0.167 ± 0.018	0.765 ± 0.031			
LOF	0.834 ± 0.029	−0.218 ± 0.045	0.755 ± 0.044			
OC-SVM	0.896 ± 0.014	−0.132 ± 0.020	0.845 ± 0.021			
SUOD	0.809 ± 0.036	−0.228 ± 0.052	0.724 ± 0.025			
COPOD	0.930 ± 0.004	−0.009 ± 0.006	0.932 ± 0.006			
VAE	0.873 ± 0.019	−0.142 ± 0.031	0.814 ± 0.022			
VLAD-NoCPD	0.920 ± 0.012	−0.071 ± 0.021	0.823 ± 0.023			
VLAD-NoReplay	0.877 ± 0.009	−0.137 ± 0.015	0.823 ± 0.019			
VLAD	0.958 ± 0.002	−0.013 ± 0.006	0.817 ± 0.030			

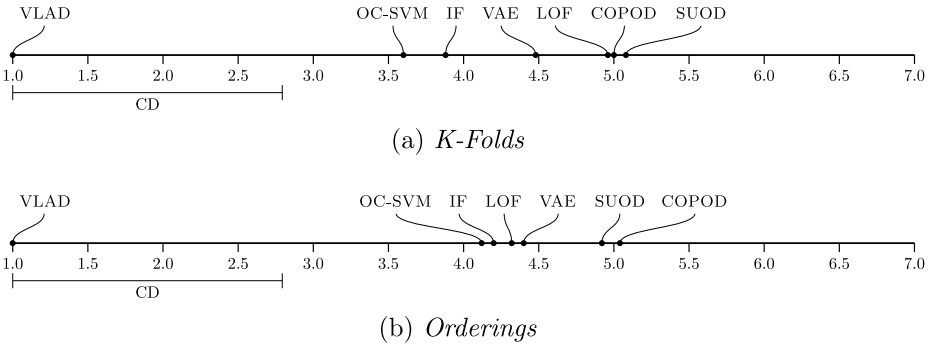


Fig. 10. Rank plot for all datasets (ROC–AUC criterion). The algorithms positioned at the leftmost side are the best performing.

tasks (Graves, Bellemare, Menick, Munos, & Kavukcuoglu, 2017), we want to assess their robustness in this context. Overall, results on different task orderings show similar trends and patterns as those observed on different executions. As expected, a higher standard deviation can be observed in the case of different task orderings. However, it does not increase by a large margin, and it still remains reasonably low. It is noteworthy that VLAD presents the overall best performance across all scenarios.

In order to validate the statistical significance of the results, we perform two different analyses. A Signed-Rank Wilcoxon test with Holm–Bonferroni correction is performed to compare all methods across all datasets. The results shown in the rank plot in Fig. 10 highlight that VLAD achieves the best performance according to the ROC–AUC metric for both modalities of experiments: different folds and different randomly sampled task orderings. All other competitor methods appear visually very distant from VLAD due to their poor performance, which implies significantly lower average ranks.

The superiority of VLAD is confirmed by a pairwise analysis of p-values with Bonferroni correction for all pairwise combinations of competitor methods with VLAD, as shown in Table 3. These results show that, in terms of ROC–AUC, VLAD significantly outperforms all the competitors taken independently.

For completeness, our experimental results include values for all metrics, even though our main interest is directed to ROC–AUC and BWT, which are direct indicators of anomaly detection performance and the model's ability to prevent forgetting, respectively. Results in terms of FWT are also reported, but not analyzed in detail, since its interpretation is not straightforward. Intuitively, if two tasks are significantly different, it is likely that

Table 3
Adjusted *p*-values of the Signed-Rank Wilcoxon test with Holm–Bonferroni correction for all pairwise combinations of competitor methods to the reference method (VLAD). Statistically significant values are marked in bold (confidence = 0.01).

Pairwise comparison	Adjusted *p*-value	Winner
ROC–AUC criterion, *K-Folds*		
VLAD *vs.* OC-SVM	**2.09E−05**	VLAD
VLAD *vs.* IF	**3.65E−06**	VLAD
VLAD *vs.* LOF	**2.28E−10**	VLAD
VLAD *vs.* VAE	**2.46E−08**	VLAD
VLAD *vs.* SUOD	**8.51E−11**	VLAD
VLAD *vs.* COPOD	**1.77E−10**	VLAD
ROC–AUC criterion, *Orderings*		
VLAD *vs.* OC-SVM	**3.29E−07**	VLAD
VLAD *vs.* IF	**2.45E−07**	VLAD
VLAD *vs.* LOF	**1.04E−07**	VLAD
VLAD *vs.* VAE	**6.57E−08**	VLAD
VLAD *vs.* SUOD	**4.21E−10**	VLAD
VLAD *vs.* COPOD	**1.33E−10**	VLAD

learning the proper boundary between normal and anomalies in the first task will lead to the classification of all normal data points from the second task as anomalies, resulting in a low FWT score. In fact, it is more intuitive to discuss FWT in the context of image classification, where the conceptual characterization of tasks is semantically clear (e.g., learning to recognize cats may be helpful to recognize tigers). Although FWT may be a relevant indicator in the context of anomaly detection, its adoption requires further investigation that is outside the scope of this paper.

7.2. Measuring forgetting

In this section, we assess VLAD's ability to mitigate forgetting in task-agnostic scenarios while preserving a robust anomaly detection performance on previously learned tasks when compared to other methods (**RQ2**). By analyzing the average values of BWT in Tables 1 and 2, it is possible to observe that all methods are exposed to a degree of forgetting in all scenarios. However, VLAD presents the least amount of forgetting in all cases by maintaining a very low memory footprint while preserving ROC–AUC performance (we refer the reader to Section 7.4 for a more detailed discussion on memory size impact on model performance).

For brevity, in this section, we are particularly interested in comparing VLAD with VAE, since it represents the core model being extended by our method with lifelong learning capabilities. Our goal is to show that the lifelong learning capabilities proposed in VLAD are able to reduce forgetting by a significant amount, allowing the model to perform successfully in a task-agnostic lifelong anomaly detection problem. In particular, looking at Table 1, we identify three types of scenarios:

- *High degree of forgetting* (NGIDS, 3IDS): VLAD reduces the forgetting presented by VAE by 70.78% (from −0.558 to −0.163) and 76.72% (from −0.464 to −0.108), respectively.
- *Average forgetting* (NSL-KDD, UNSW), VLAD reduces the forgetting presented by VAE by 81.99% (from −0.311 to −0.056) and 89.69% (from −0.359 to −0.037), respectively.
- *Limited forgetting* (WIND), VLAD reduces the forgetting presented by VAE by 91.61% (from −0.167 to −0.014).

In addition to aggregating BWT values in Tables 1 and 2, we measure ROC–AUC evaluated on each task after learning different tasks. In this way, we decompose the average ROC–AUC performance focusing on its evolution during the learning lifespan of the model. We present these results in Fig. 11, focusing on the most challenging 3IDS scenario. It is possible to observe that, overall, VAE presents a large amount of forgetting by dropping its ROC–AUC performance for tasks that are different than the last learned one. VLAD overcomes this problem presenting a significantly more stable performance across tasks.

For instance, after learning Task 0, VAE presents a ROC–AUC performance of 0.99 on the same task, which catastrophically drops to 0.13 after learning Task 1. After learning Task 2, the performance on Task 0 recoups to 0.61 (which is slightly better than random), to drop again to 0.019 after learning Task 3 and recouping to 0.43 (below random) after learning Task 4. On the other hand, VLAD is able to retain much more of the ROC–AUC performance on Task 0 after learning Task 1 (0.72, compared to 0.13 for VAE), Task 2 (0.85, compared to 0.61 for VAE), Task 3 (0.82, compared to 0.019 for VAE), and Task 4 (0.95, compared to 0.43 for VAE). Even if VAE presents a better performance on single tasks, usually right after they are learned, its overall performance on all tasks is severely impacted by forgetting. Similar observations can be drawn analyzing results for VAE, choosing different columns in the heatmap and considering the entries on and below the main diagonal from top to bottom. These results lead us to confirm our hypothesis devised in **RQ2**.

7.3. Memory representation in long scenarios with recurring tasks

In this section, we evaluate VLAD's ability to extract an accurate memory representation in terms of the number of discovered concepts in long scenarios with recurring tasks (**RQ3).** More specifically, the natural sequence of tasks is repeated multiple times, for a maximum of 6 times. This experiment is aimed at verifying VLAD's ability to properly differentiate between new and recurring tasks and perform accurate memory updates that reflect the real characterization of tasks presented by the data.

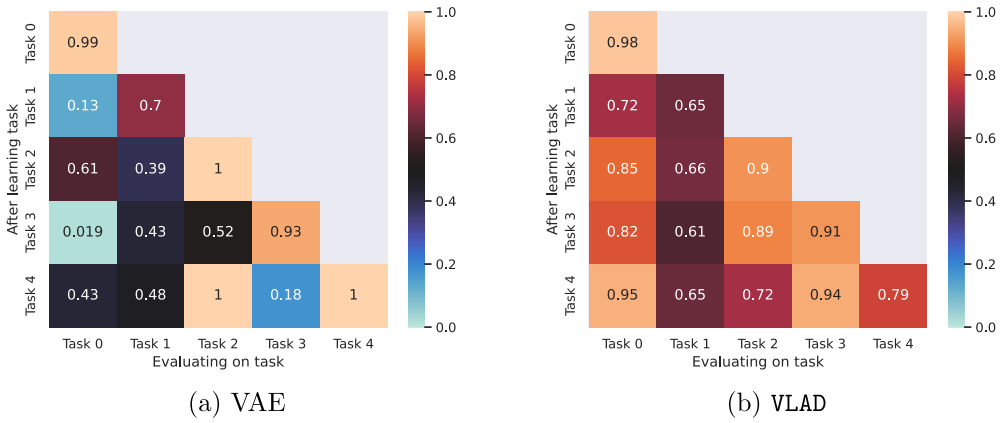
The results in Fig. 12 show the number of concepts stored in VLAD's memory over time, as new tasks are presented. The results highlight three main patterns: *(i)* A gradually increasing number of concepts that stabilizes at a certain time and does not increase further (UNSW, NGIDS, and WIND); *(ii)* A rapidly increasing number of concepts that stabilizes or keeps increasing slowly and occasionally (NSL-KDD); *(iii)* A slowly increasing number of concepts, which becomes close to the actual number of tasks after recurrence (3IDS), possibly due to the added complexity deriving from overlapping concepts.

In general, it can be observed that VLAD is capable of extracting an accurate memory representation with a number of concepts that closely approximates the actual number of concepts presented by the data, if required by the learning setting. Its robustness is remarkable since the scenarios in this experiment present recurring tasks multiple times, which represent challenging circumstances with many transitions between tasks.

It is worth noting that without Lifelong Change Point Detection capabilities, this recurring scenario would imply the creation of a new concept at each detected task transition, taking place due to memory fragmentation, and a degradation in terms of the generalization capabilities of the model. This behavior is illustrated in Fig. 5 and further discussed in the Appendix (Appendix A.1). This result allows us to answer positively to **RQ3**.

7.4. Memory size impact on model performance

In this section, we assess VLAD's ability to achieve a high anomaly detection performance with a low memory footprint in terms of the number of stored experiences (**RQ4**). To achieve this goal, we show experiments with different configurations for memory size that show the correlation between memory size and model performance in terms of ROC–AUC and BWT. The results in Fig. 13 show averaged ROC–AUC and BWT results for multiple executions of VLAD (k-folds) on all scenarios. The main takeaways from our experiments are summarized in the following:



(a) VAE (b) VLAD

Fig. 11. Comparison of ROC–AUC values obtained for VAE and VLAD on each task after learning different tasks (Multiple Intrusion Detection Datasets scenario). Only entries below the main diagonal are deemed relevant for analyzing forgetting, since values above the main diagonal indicate performance on not yet seen tasks.

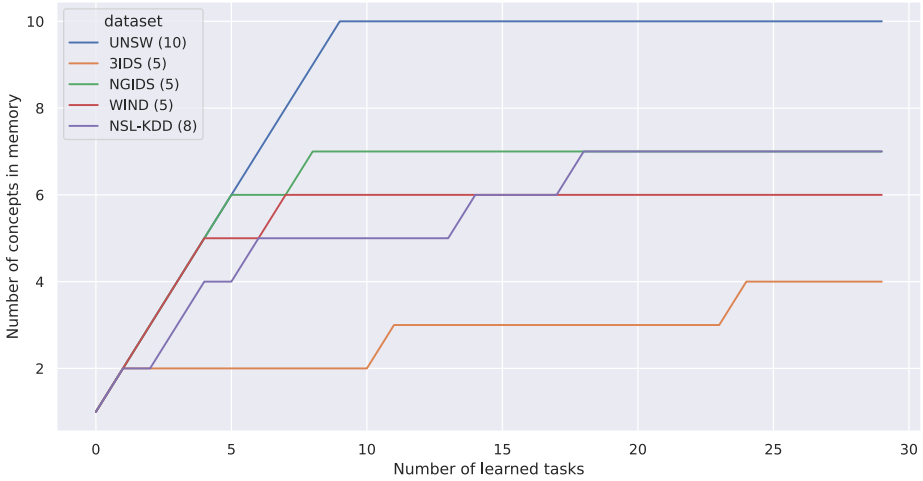


Fig. 12. Number of concepts stored in memory in scenarios with recurrence, where the full task sequence is presented multiple times. The actual number of distinct tasks for each scenario is reported in parenthesis (legend).

- *NGIDS:* The highest ROC–AUC performance is obtained with 750 samples, which represents an acceptable memory occupation. It is possible to further reduce it to 500 samples (or 400 samples), which results in a ROC–AUC decrease of 1.01% (or 1.87%) in comparison to the highest performance achieved with varying memory size configurations, while saving 33.33% (or 46.67%) in terms of memory occupation.
- *NSL-KDD:* The best ROC–AUC performance is obtained with 10 000 samples. However, the second best performance is achieved with a memory occupation of 300 samples, which results in a ROC–AUC decrease of 0.20% in comparison to the highest performance, while reducing memory occupation by a large margin (97%).
- *UNSW:* The highest ROC–AUC performance is obtained with 2000 samples, which represents a moderate memory occupation. It is possible to reduce it to 750 samples, which results in a decrease of 1.25% of the highest ROC–AUC, while reducing memory occupation by 65.50%.
- *3IDS:* The best ROC–AUC performance is obtained with 7500 samples. A viable option would be to reduce memory occupation to 2000 samples, which results in a ROC–AUC decrease of 1.64% in comparison to the highest performance, while reducing memory occupation by a large margin (73.33%). Furthermore, it would be possible to further reduce memory occupation to 500 samples, resulting in a ROC–AUC decrease of 7.83% while saving 93.33% in memory occupation. Even though the decrease in performance appears significant, the ROC–AUC of VLAD would still be higher than that achieved by all competitor methods considered in our experiments (see Table 1), which fundamentally perform close to a random classifier given the complexity of this scenario.
- *WIND:* The highest ROC–AUC performance is achieved with 7500 samples. However, the best trade-off between memory and model performance can be obtained with a very low memory size of 200 samples (reduction of 97.33%), which results in a decrease of only 0.21% of the highest ROC–AUC.

In general, it can be observed that it is possible to achieve a good trade-off between anomaly detection performance and memory size in all scenarios, keeping a low resource footprint. This result allows to answer positively to **RQ4**.

7.5. Ablation study

In this section, we assess whether the synergic combination of all components in VLAD contributes to achieving a better anomaly detection performance than that provided by any of the components in isolation (**RQ5**). The results in Table 1 show averaged results on multiple executions of the methods (k-folds) on all scenarios. The configurations of interest in this subsection are:

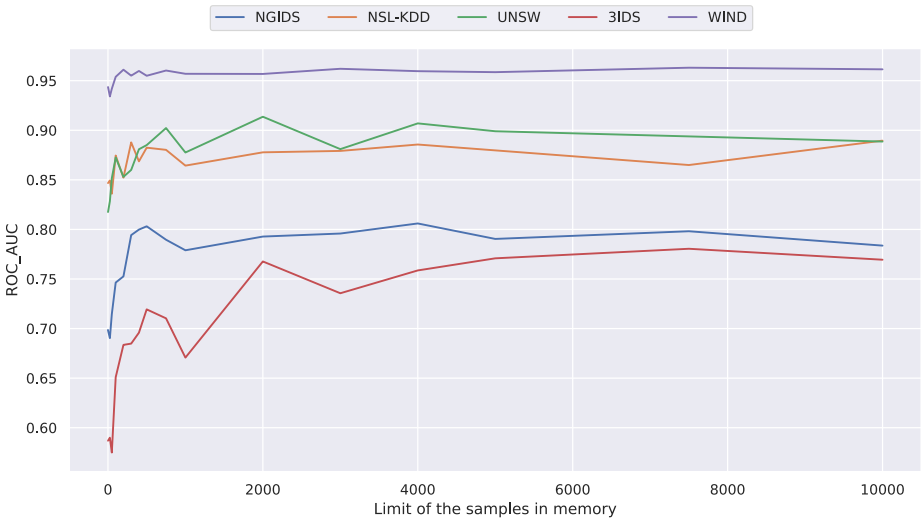


Fig. 13. Anomaly detection performance in terms of ROC–AUC with varying soft upper memory bound M_B configurations (number of samples stored) in all scenarios.

- *VAE*: The base model upon which VLAD is built. It is important since it provides a starting point to subsequently assess a possible impact of our components on anomaly detection performance.
- VLAD−NoReplay: This method simplifies VLAD by deactivating the Experience Replay component. This deactivation implies that the VAE model is aware of task changes and it is updated accordingly as an incremental learner, adapting to new tasks.
- VLAD−NoCPD: This method simplifies VLAD by deactivating the Lifelong Change Point Detection component. This deactivation implies that the VAE model is enhanced by Consolidation, Memory Summarization, and Experience Replay, but in a simplified manner, i.e., the method is able to store experiences but without the fine-grained characterization and recognition of concepts, which requires information provided by the Lifelong Change Point Detection component.
- VLAD: It represents the full method including Lifelong Change Point Detection, Consolidation, Memory Summarization, and Experience Replay.

The results on individual scenarios can be summarized as follows:

- *NGIDS*: VAE presents an unsatisfactory anomaly detection performance (0.583) and backward transfer (−0.558). VLAD−NoReplay slightly improves this result, but not significantly (0.591 and −0.546, respectively). On the other hand, VLAD−NoCPD significantly improves the anomaly detection performance (0.749) while reducing forgetting (−0.033). Even if VLAD presents a lower backward transfer than VLAD−NoCPD (−0.165), it obtains the best anomaly detection performance (0.786), improving the second best ablation method by 3.74%.
- *NSL-KDD*: VAE presents a moderate anomaly detection performance (0.700) and backward transfer (−0.311). VLAD−NoReplay slightly decreases this performance (0.693 and −0.312, respectively). Similarly to NGIDS, VLAD−NoCPD significantly improves the anomaly detection performance (0.818) while improving backward transfer (−0.040). Analogously, VLAD presents a slightly lower backward transfer than VLAD−NoCPD (−0.056), but it achieves the highest anomaly detection performance (0.880), improving the second best ablation method by 7.58%.
- *UNSW*: VAE presents a sub-optimal anomaly detection performance (0.668) and backward transfer (−0.359). VLAD−NoReplay presents a very small decrease in performance (0.664 and −0.363, respectively). VLAD−NoCPD significantly improves the anomaly detection performance (0.802), and a remarkable positive backward transfer (0.113). Nevertheless, VLAD presents the highest anomaly detection performance (0.910) improving the second best ablation method by 13.47%, while almost completely overcoming negative backward transfer (−0.037).
- *3IDS*: VAE presents a poor anomaly detection performance (0.597) and backward transfer (−0.464). Both VLAD−NoReplay and VLAD−NoCPD are unable to improve this performance. VLAD introduces a large improvement, achieving the best anomaly detection performance (0.779), improving the second best ablation method by 30.49%, while preserving a reasonable value for backward transfer (−0.108).
- *WIND*: All methods present a reasonable performance in this scenario. The base model VAE presents a satisfactory anomaly detection performance (0.858) and moderate forgetting (−0.167). VLAD−NoReplay presents similar results (0.858 and −0.166, respectively), while VLAD−NoCPD yields a small improvement (0.896 and −0.111, respectively). Also, in this scenario, VLAD achieves the highest anomaly detection performance (0.959), improving the second best ablation method by 7.03%, while almost completely mitigating forgetting (−0.014).

In summary, it can be observed that lifelong change point detection (in VLAD−NoReplay) does not improve the performance of the base model on its own. This result is expected since VAE is constantly updated as new data is observed. Therefore, even if lifelong change point detection in VLAD−NoReplay reduces the frequency of model updates, the model is very prone to forgetting. In other words, VLAD−NoReplay is affected by its inability to retain previously learned concepts as new concepts are learned via model update. On the other hand, VLAD−NoCPD presents a significant improvement over the base model (VAE), but it may be affected by its inability to timely update the model when required. This behavior is particularly visible in the 3IDS scenario, where VLAD−NoCPD is unable to provide a significant improvement over VAE.

It is remarkable to observe that the synergic combination of all components in VLAD yields the largest improvement over all

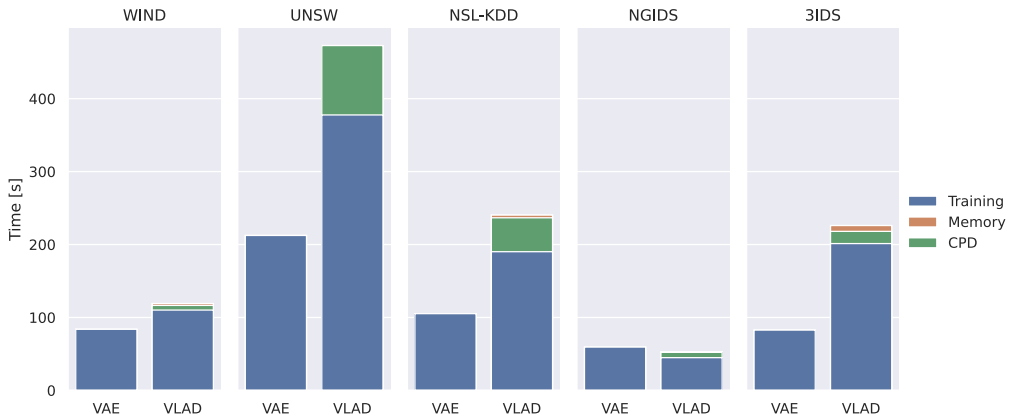


Fig. 14. Empirical time complexity for VLAD disaggregated by stages (Training, Memory, CPD) and comparison with its base model (VAE) for all analyzed datasets (WIND, UNSW, NSL-KDD, NGIDS, 3IDS). The reported results are averaged across 10 executions.

scenarios, positively answering to the research question elicited in **RQ5**. We argue that 3IDS is a clear demonstration of the capabilities of the method in complex settings, since this scenario presents a large heterogeneity of tasks, originating from multiple datasets.

7.6. Additional analysis

In this section, we provide an additional analysis on the time complexity of VLAD, also in comparison with the base model, VAE. Moreover, we discuss the impact of different hyperparameter values on performance.

7.6.1. Time complexity

To evaluate the overhead of the lifelong learning components in VLAD, we conducted an empirical time complexity analysis comparing its execution time with that of its base model, VAE. Results in Fig. 14 highlight that model training dominates the execution time, whereas Memory Management (Memory) and Lifelong Change Point Detection (CPD) require a fraction of the overall execution time, which is more visible in UNSW and NSL-KDD, possibly due to their higher data dimensionality in comparison with other datasets. A higher training time for VLAD is expected, due to data accumulation for Experience Replay, unlike VAE, which undergoes training with constant data size. One exception is observed with the NGIDS dataset, where VLAD presents a lower execution time than VAE. This result is explained by the ability of CPD to avoid unnecessary model updates, unlike VAE, which does not provide any mechanism to detect changes, and requires the adoption of a constant model update strategy.

Computing the ratio between the execution time of VLAD and VAE results in factors of 1.42 (WIND), 2.23 (UNSW), 2.28 (NSL-KDD), 0.89 (NGIDS), and 2.73 (3IDS). As a general result, we observe that the execution time in VLAD is usually higher than that of its base model. We argue that this increase in the execution time is still reasonable, and it is justified by the powerful lifelong learning capabilities offered by VLAD. Moreover, from a more general perspective, the time factors highlight that VLAD and VAE operate in the same class of time complexity.

7.6.2. Impact of hyperparameter values on the performance

Moreover, we performed additional experiments to study the impact of different hyperparameter values on the lifelong anomaly detection performance. From preliminary experiments, we observed no significant difference in Lifelong ROC–AUC results with heuristically defined configurations of C_κ, C_ω, M_f, M_k. Consequently, we fixed their configuration to those defined for the main experiments described in Section 6, and focused on varying configurations for the remainder of the parameters. Specifically, the search space included the following list of values: *(i)* Soft upper bound for Memory Summarization $M_B = \{3000, 5000, 10000\}$; *(ii)* maximum internal distribution distance ratio $C_\epsilon = \{1.5, 2, 2.5, 3\}$; *(iii)* adaptive consolidation threshold ratio $W_\epsilon = \{2, 5\}$; *(iv)* maximum time between model updates $R_\psi = \{5000, 15000\}$. Our experiments involving 48 hyperparameter configurations revealed that the impact of their values in terms of standard deviation of Lifelong ROC–AUC with all datasets was negligible. Specifically, values of 0.032, 0.024, 0.027, 0.034, and 0.022 were observed with NGIDS, 3IDS, NSL-KDD, UNSW, and WIND datasets, respectively. These results show that VLAD yields a robust lifelong anomaly detection regardless of the specific hyperparameter values, without being overly sensitive to their setting. At the same time, we would like to point out that using smaller values of memory size would have yielded a significantly larger standard deviation, as we highlighted in Section 7.4. In this analysis, we selected high values of soft memory upper bound to focus on a realistic scenario where memory size is large enough to support relevant method capabilities, such as consolidation and experience replay.

8. Conclusions and future work

In this paper, we proposed a novel task-agnostic lifelong learning method that detects anomalies, adapts to changing environments as new data is available, and preserves knowledge from previously learned experiences. By addressing these three challenges in a combined way, the method overcomes limitations of both anomaly detection methods (which are prone to forgetting) and lifelong learning methods (which are not suitable for solving the anomaly detection problem in task-agnostic scenarios). Our method leverages a novel lifelong change point detection approach connected with hierarchical memory consolidation and pyramidal summarization, as well as an effective model update strategy that involves experience replay.

An extensive quantitative evaluation was conducted on a variety of applied real-world scenarios, including network intrusion detection, host-based intrusion detection, and renewable energy. Results showed that our method is capable of achieving a higher anomaly detection performance than non-lifelong state-of-the-art methods for anomaly detection, in task sequential scenarios where the normal class changes over time. Moreover, we showed that high-quality experience replay supported by lifelong change point detection, memory management, and pyramidal summarization, significantly mitigates forgetting while yielding high anomaly detection performance in scenarios where task boundaries are not known to the model. The method is able to

yield high anomaly detection performance even with very low requirements in terms of memory footprint. Furthermore, the memory representation of the induced concepts closely recalls the natural task characterization presented in data. Finally, an ablation study confirmed that the synergic combination of all components provides a higher anomaly detection performance than any of the components in isolation.

As future work, we will investigate novel approaches for model-agnostic experience replay in lifelong anomaly detection. Another research direction worth addressing is that of challenging lifelong anomaly detection scenarios in settings with heterogeneous tasks, as shown in the 3IDS scenario in our experiments, where the robustness of the model is further challenged by overlapping and highly varying tasks. Another interesting research direction is that of detection of contextual anomalies in lifelong learning scenarios, where part of the normal data for one task is considered as an anomaly for another task. Finally, we will explore different learning settings including active learning and semi-supervised contrastive learning.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The work presented in this article was partially funded by DARPA through the project "Lifelong Streaming Anomaly Detection" (Grant N. A19-0131-003 and A21-0113-002). The authors also acknowledge the support of the Polish Ministry of Education and Science assigned to AGH University of Science and Technology in Kraków, Poland. The authors also acknowledge the support of NVIDIA through the donation of a Titan V GPU.

Appendix

In this section, we illustrate some concepts related to our method's components and their combined effect in a typical lifelong learning setting.

A.1. Lifelong Change Point Detection and consolidation

While *Lifelong Change Point Detection* allows to detect task changes, with a positive impact in terms of timely model updates, it also positively impacts memory induction and management strategies. A simple change point detector would not be able to discriminate between new tasks and previously observed tasks. Fig. A.15 shows the different memory hierarchies obtained with conventional non-lifelong change point detection, and lifelong change point detection, when the detection of a change is used as feedback to create a new concept in the memory. In the former case, it is possible to observe that every change triggers the creation of a new node in the memory hierarchy. In the latter case, we can observe a more accurate memory hierarchy that depicts the true characterization of the data presented to the model in the lifelong learning setting. This result is due to the ability of lifelong change point detection to characterize recurring tasks and newly observed tasks, and update the memory accordingly. Intuitively, a more accurate memory hierarchy should positively impact the performance of the *Memory Summarization* component during the summarization process, and the *Experience Replay* component, since a more precise characterization of tasks yields a more balanced representation of experiences which is used to update the model.

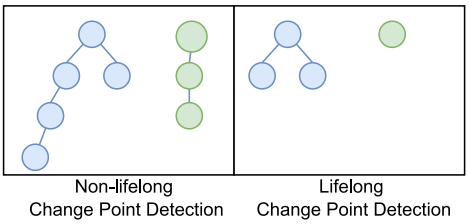


Fig. A.15. Conceptual representation of the impact of the *Lifelong Change Point Detection* component in memory management. Non-lifelong Change Point Detection (left) leads to the creation of new concept in the memory each time a change is detected. *Lifelong Change Point Detection* (right) instead, since it is able to characterize changes to recurrent or new tasks. Recurring changes do not require the creation of new nodes in the memory, leading to the incorporation of those experiences in existing nodes.

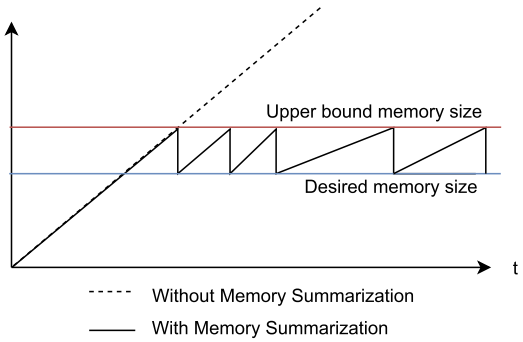


Fig. A.16. Conceptual representation of the impact of memory summarization. While a method with memory summarization keep memory occupation below a desired upper bound, a memory without memory summarization inevitably leads to the memory explosion issue.

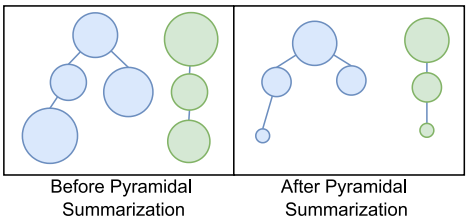


Fig. A.17. Conceptual representation of the impact of the Pyramidal Summarization.

A.2. Consolidation and Memory Summarization

Memory summarization allows to maintain memory size to a desired upper bound. While it is important to retain past knowledge, it is not feasible to keep all experiences in memory. Indeed, a viable lifelong learning system should ensure the compactness of the memory representation to be reliable and sustainable in long-term scenarios. Therefore, summarization represents an important process in a lifelong learning system, which prevents memory explosion issues. This process is graphically represented in Fig. A.16.

Another aspect worth noting is the impact of the *Consolidation* component in the pyramidal summarization process. In fact, Pyramidal summarization ensures a proper balance in the memory hierarchy extracted by the *Consolidation* component (see Fig. A.17). Assuming the absence of such component, implies that an equal number of samples would be stored for each observed concept, and it would not be possible to extract a memory hierarchy. On the other hand, the absence of pyramidal summarization would not allow to reflect the actual importance of root

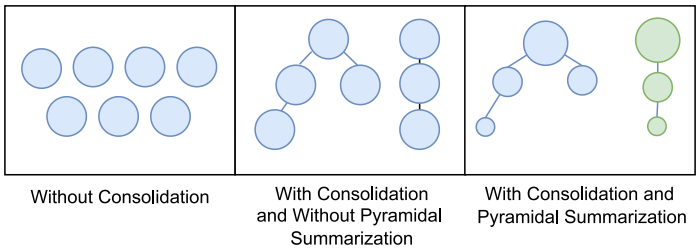


Fig. A.18. Conceptual representation of the combined impact of Consolidation and Pyramidal Summarization. The Consolidation component yields a memory hierarchy that can be effectively summarized by means of Pyramidal Summarization to reflect the actual importance of root concepts and sub-concepts.

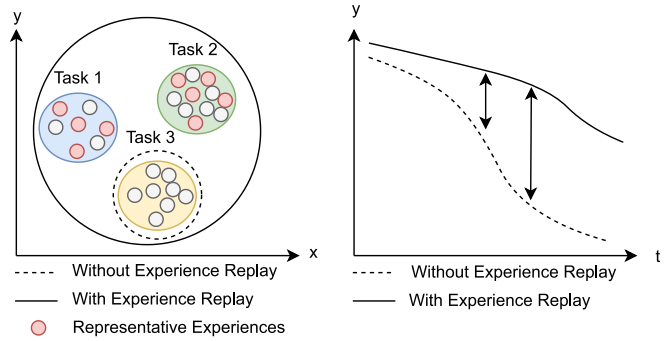


Fig. A.19. Conceptual representation of the impact of the *Experience Replay* component in a lifelong learning setting. In terms of **model fitting** (left), the model leveraging experience replay is fit with most representative experiences from Task 1 and 2, while accommodating new experiences from Task 3. On the other hand, the model without experience replay is exclusively updated using Task 3, leading to overfitting and forgetting. In terms of **expected model performance** (right), the model with experience replay retains a satisfactory performance over all previously learned tasks, whereas the model without experience replay quickly drops its performance as new tasks are learned.

concepts and sub-concepts in the hierarchical memory. Overall, their combined contribution yields a memory hierarchy that is properly balanced as shown in Fig. A.18. It is worth noting that the ability to effectively consolidate and summarize experiences strongly benefits from the synergic interaction between *Consolidation* and *Lifelong Change Point Detection*, which is responsible for identifying new concepts that are later organized in a hierarchy.

A.3. Memory Summarization and Experience Replay

While memory summarization allows to maintain representative experiences from the previously observed tasks, experience replay ensures that they are constantly incorporated in the model once it is updated. Their combined effect is twofold: *(i)* the model is not exclusively updated using the most recently observed task, but it also retains the most relevant knowledge from previous tasks, thus preventing overfitting and forgetting; *(ii)* the expected anomaly detection performance over all previously learned tasks is retained, differently than a model that does not provide experience replay capabilities. This concept is illustrated in Fig. A.19. Moreover, it is possible to note that a successful experience replay process relies on the high quality of available experiences in memory. This characteristic is fostered by all other components, i.e., Memory Summarization, Consolidation, and Lifelong Change Point Detection.

References

Abel, D., Arumugam, D., Lehnert, L., & Littman, M. (2018). State abstractions for lifelong reinforcement learning. In *International conference on machine learning* (pp. 10–19).

Abel, D., Jinnai, Y., Guo, S. Y., Konidaris, G., & Littman, M. (2018). Policy and value transfer in lifelong reinforcement learning. In *International conference on machine learning* (pp. 20–29).

Aljundi, R., Chakravarty, P., & Tuytelaars, T. (2017). Expert gate: Lifelong learning with a network of experts. In *2017 IEEE conference on computer vision and pattern recognition* CVPR, (pp. 7120–7129).

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, *2*(1), 1–18.

Berntson, G. G., Boysen, S. T., & Cacioppo, J. T. (1993). Neurobehavioral organization and the cardinal principle of evaluative bivalence. *Annals of the New York Academy of Sciences*, *702*, 75–102.

Bottou, L., & Bengio, Y. (1994). Convergence properties of the k-means algorithms. *Advances in Neural Information Processing Systems*, *7*.

Branco, P., Torgo, L., & Ribeiro, R. P. (2016). A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys*, *49*(2), 1–50.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on management of data* (pp. 93–104).

Buzzega, P., Boschini, M., Porrello, A., & Calderara, S. (2021). Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th international conference on pattern recognition* (pp. 2180–2187). IEEE.

Chaudhry, A., Ranzato, M., Rohrbach, M., & Elhoseiny, M. (2018). Efficient lifelong learning with A-GEM. In *International conference on learning representations*.

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explorations Newsletter*, *6*(1), 1–6.

Chen, Y., Diethe, T., & Lawrence, N. (2019). Facilitating bayesian continual learning by natural gradients and stein gradients. arXiv preprint arXiv: 1904.10644.

Corizzo, R., Baron, M., & Japkowicz, N. (2022). CPDGA: Change point driven growing auto-encoder for lifelong anomaly detection. *Knowledge-Based Systems*, Article 108756.

Corizzo, R., Ceci, M., Pio, G., Mignone, P., & Japkowicz, N. (2021). Spatially-aware autoencoders for detecting contextual anomalies in geo-distributed data. In *Discovery science: 24th international conference, DS 2021, Halifax, NS, Canada, October 11–13, 2021, Proceedings 24* (pp. 461–471). Springer.

Creech, G., & Hu, J. (2013). Generation of a new IDS test dataset: Time to retire the KDD collection. In *2013 IEEE wireless communications and networking conference* WCNC, (pp. 4487–4492).

de Masson D'Autume, C., Ruder, S., Kong, L., & Yogatama, D. (2019). Episodic memory in lifelong language learning. In *Advances in neural information processing systems, Vol. 32* (pp. 13132–13141).

Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., & Maltoni, D. (2018). Don't forget, there is more than forgetting: new metrics for continual learning. arXiv: 1810.13166.

Dittenbach, M., Merkl, D., & Rauber, A. (2000). The growing hierarchical self-organizing map. In *Proceedings of the IEEE-INNS-ENNS international joint conference on neural networks. IJCNN 2000. Neural computing: New challenges and perspectives for the new millennium, Vol. 6* (pp. 15–19). IEEE.

Doshi, K., & Yilmaz, Y. (2020). Continual learning for anomaly detection in surveillance videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 254–255).

Faber, K., Corizzo, R., Sniezynski, B., Baron, M., & Japkowicz, N. (2021). WATCH: Wasserstein change point detection for high-dimensional time series data. In *2021 IEEE international conference on big data (Big data)* (pp. 4450–4459). IEEE.

Faber, K., Corizzo, R., Sniezynski, B., Baron, M., & Japkowicz, N. (2022). LIFE-WATCH: Lifelong wasserstein change point detection. In *2022 international joint conference on neural networks* IJCNN, (pp. 1–8). IEEE.

Faber, K., Corizzo, R., Sniezynski, B., & Japkowicz, N. (2022). Active lifelong anomaly detection with experience replay. In *2022 IEEE 9th international conference on data science and advanced analytics* DSAA, (pp. 1–10). IEEE.

Faber, K., Pietron, M., & Zurek, D. (2021). Ensemble neuroevolution-based approach for multivariate time series anomaly detection. *Entropy*, *23*(11), 1466.

Fourure, D., Javaid, M. U., Posocco, N., & Tihon, S. (2021). Anomaly detection: How to artificially increase your F1-score with a biased evaluation protocol. In Y. Dong, N. Kourtellis, B. Hammer, & J. A. Lozano (Eds.), *Machine learning and knowledge discovery in databases. Applied data science track* (pp. 3–18). Cham: Springer International Publishing.

Frikha, A., Krompaß, D., & Tresp, V. (2021). ARCADe: A rapid continual anomaly detector. In *2020 25th international conference on pattern recognition* ICPR, (pp. 10449–10456). IEEE.

Goldstein, M., & Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLoS One*, *11*(4), Article e0152173.

Gopalakrishnan, S., Singh, P. R., Fayek, H., Ramasamy, S., & Ambikapathi, A. (2022). Knowledge capture and replay for continual learning. In *2022 IEEE/CVF winter conference on applications of computer vision* WACV, (pp. 337–345).

Graves, A., Bellemare, M. G., Menick, J., Munos, R., & Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. In *International conference on machine learning* (pp. 1311–1320). PMLR.

Grossberg, S. (1982). How does a brain build a cognitive code? In *Studies of mind and brain* (pp. 1–52). Springer.

Grossberg, S. (2013). Adaptive Resonance Theory: How a brain learns to consciously attend, learn, and recognize a changing world. *Neural Networks*, *37*, 1–47, Twenty-fifth Anniversay Commemorative Issue.

Haider, W., Hu, J., Slay, J., Turnbull, B., & Xie, Y. (2017). Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling. *Journal of Network and Computer Applications*, *87*, 185–192.

Hallin, M., Mordant, G., & Segers, J. (2021). Multivariate goodness-of-Fit tests based on Wasserstein distance. arXiv:2003.06684.

Higgins, I., Matthey, L., Glorot, X., Pal, A., Uria, B., Blundell, C., et al. (2016). Early visual concept learning with unsupervised deep learning. CoRR abs/1606.05579.

Isele, D., & Cosgun, A. (2018). Selective experience replay for lifelong learning. In *32nd AAAI conference on artificial intelligence, AAAI 2018* (pp. 3302–3309). Cited by: 55.

Joseph, K. J., & Balasubramanian, V. N. (2020). Meta-consolidation for continual learning. *Advances in Neural Information Processing Systems*, *33*, 14374–14386.

Junior, F. E. F., & Yen, G. G. (2019). Particle swarm optimization of deep neural networks architectures for image classification. *Swarm and Evolutionary Computation*, *49*, 62–74.

Khan, S., & Madden, M. G. (2014). One-class classification: taxonomy of study and review of techniques. *Knowledge Engineering Review*, *29*(3), 345–374.

Kingma, D. P., Welling, M., et al. (2019). An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, *12*(4), 307–392.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, *114*(13), 3521–3526.

Korycki, L., & Krawczyk, B. (2021a). Class-incremental experience replay for continual learning under concept drift. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3649–3658).

Korycki, Ł., & Krawczyk, B. (2021b). Streaming decision trees for lifelong learning. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 502–518). Springer.

Kumaran, D., Hassabis, D., & McClelland, J. L. (2016). What learning systems do intelligent agents need? Complementary learning systems theory updated. *Trends in Cognitive Sciences*, *20*(7), 512–534.

Kurle, R., Cseke, B., Klushyn, A., van der Smagt, P., & Günnemann, S. (2019). Continual learning with Bayesian neural networks for non-stationary data. In *International conference on learning representations*.

Lee, S., Ha, J., Zhang, D., & Kim, G. (2020). A neural Dirichlet process mixture model for task-free continual learning. In *International conference on learning representations*.

Li, Y.-F., Gao, Y., Ayoade, G., Tao, H., Khan, L., & Thuraisingham, B. (2019). Multistream classification for cyber threat data with heterogeneous feature space. In *The world wide web conference* WWW '19, (pp. 2992–2998). New York, NY, USA: Association for Computing Machinery.

Li, Z., & Hoiem, D. (2017). Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(12), 2935–2947.

Li, Z., Zhao, Y., Botta, N., Ionescu, C., & Hu, X. (2020). COPOD: copula-based outlier detection. In *2020 IEEE international conference on data mining* ICDM, (pp. 1118–1123). IEEE.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *2017 IEEE conference on computer vision and pattern recognition* CVPR, (pp. 936–944). IEEE.

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 eighth IEEE international conference on data mining* (pp. 413–422). IEEE.

Lomonaco, V. (2019). *Continual learning with deep architectures* (Ph.D. thesis), University of Bologna.

Lopez-Paz, D., & Ranzato, M. (2017). Gradient episodic memory for continual learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems, Vol. 30*. Curran Associates, Inc..

Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7765–7773).

Malondkar, A., Corizzo, R., Kiringa, I., Ceci, M., & Japkowicz, N. (2019). Spark-GHSOM: growing hierarchical self-organizing map for large scale mixed attribute datasets. *Information Sciences*, *496*, 572–591.

Maltoni, D., & Lomonaco, V. (2019). Continuous learning in single-incremental-task scenarios. *Neural Networks*, *116*, 56–73.

Mao, F., Weng, W., Pratama, M., & Yee, E. Y. K. (2021). Continual learning via inter-task synaptic mapping. *Knowledge-Based Systems*, *222*, Article 106947.

Mathieu, E., Rainforth, T., Siddharth, N., & Teh, Y. W. (2019). Disentangling disentanglement in variational autoencoders. In *International conference on machine learning* (pp. 4402–4412). PMLR.

Miikkulainen, R., Liang, J., Meyerson, E., Rawal, A., Fink, D., Francon, O., et al. (2019). Evolving deep neural networks. In *Artificial intelligence in the age of neural networks and brain computing* (pp. 293–312). Elsevier.

Moustafa, N., & Slay, J. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 Military communications and information systems conference (MilCIS)* (pp. 1–6).

New, A., Baker, M., Nguyen, E., & Vallabha, G. (2022). Lifelong learning metrics. arXiv:2201.08278.

Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2021). Deep learning for anomaly detection: A review. *ACM Computing Surveys*, *54*(2), 1–38.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, *113*, 54–71.

Parisi, G. I., Tani, J., Weber, C., & Wermter, S. (2017). Lifelong learning of human actions with deep neural network self-organization. *Neural Networks*, *96*, 137–149.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Provost, F., & Fawcett, T. (2001). Robust classification for imprecise environments. *Machine Learning*, *42*(3), 203–231.

Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., et al. (2016). Variational autoencoder for deep learning of images, labels and captions. *Advances in Neural Information Processing Systems*, *29*.

Raghavan, A., Hostetler, J., Sur, I., Rahman, A., & Divakaran, A. (2020). Lifelong learning using eigentasks: Task separation, skill acquisition, and selective transfer. arXiv preprint arXiv:2007.06918.

Rajasegaran, J., Hayat, M., Khan, S., Khan, F. S., & Shao, L. (2019). Random path selection for incremental learning. *Advances in Neural Information Processing Systems*.

Santhakumar, K., & Kasaei, H. (2022). Lifelong 3D object recognition and grasp synthesis using dual memory recurrent self-organization networks. *Neural Networks*, *150*, 167–180.

Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., & Platt, J. C. (2000). Support vector method for novelty detection. In *Advances in neural information processing systems* (pp. 582–588).

Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y. W., Pascanu, R., et al. (2018). Progress and Compress: A scalable framework for continual learning. In *Proceedings of machine learning research: vol. 80, Proceedings of the 35th international conference on machine learning* (pp. 4528–4537). Stockholmsmässan, Stockholm Sweden: PMLR.

Sengupta, S., Basak, S., Saikia, P., Paul, S., Tsalavoutis, V., Atiah, F., et al. (2020). A review of deep learning with special emphasis on architectures, applications and recent trends. *Knowledge-Based Systems*, *194*, Article 105596.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2014). Overfeat: Integrated recognition, localization and detection using convolutional networks. In *2nd international conference on learning representations, ICLR 2014 - Conference track proceedings*.

Serra, J., Suris, D., Miron, M., & Karatzoglou, A. (2018). Overcoming catastrophic forgetting with hard attention to the task. arXiv preprint arXiv:1801.01423.

Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 806–813).

Sherrington, C. (1952). *The integrative action of the nervous system*. CUP Archive.

Shin, H., Lee, J. K., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. In *Advances in neural information processing systems* (pp. 2990–2999).

Soltoggio, A., Stanley, K. O., & Risi, S. (2018). Born to learn: The inspiration, progress, and future of evolved plastic artificial neural networks. *Neural Networks*, *108*, 48–67.

Stanley, K. O., & Miikkulainen, R. (2002). Efficient reinforcement learning through evolving neural network topologies. In *Proceedings of the 4th annual conference on genetic and evolutionary computation* (pp. 569–577).

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (pp. 1–6).

Tinbergen, N. (1950). The hierarchical organization of nervous mechanisms underlying instinctive behaviour. In *Symposia of the society for experimental biology, Vol. 4* (pp. 305–312).

Titsias, M. K., Schwarz, J., Matthews, A. G. d. G., Pascanu, R., & Teh, Y. W. (2019). Functional regularisation for continual learning with Gaussian processes. In *International conference on learning representations*.

Ünal, H. T., & Başçiftçi, F. (2021). Evolutionary design of neural network architectures: a review of three decades of research. *Artificial Intelligence Review*, 1–80.

van de Ven, G. M., & Tolias, A. S. (2018). Generative replay with feedback connections as a general strategy for continual learning. arXiv preprint arXiv:1809.10635.

Van de Ven, G. M., & Tolias, A. S. (2019). Three scenarios for continual learning. arXiv preprint arXiv:1904.07734.

Wiewel, F., & Yang, B. (2019). Continual learning for anomaly detection with variational autoencoder. In *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing* ICASSP, (pp. 3837–3841). IEEE.

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., et al. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference* (pp. 187–196).

Yoon, J., Yang, E., Lee, J., & Hwang, S. J. (2018). Lifelong learning with dynamically expandable networks. In *International conference on learning representations*.

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. In *Proceedings of the 34th international conference on machine learning-Volume 70* (pp. 3987–3995).

Zhao, Y., Hu, X., Cheng, C., Wang, C., Wan, C., Wang, W., et al. (2021). SUOD: Accelerating large-scale unsupervised heterogeneous outlier detection. In *Proceedings of machine learning and systems, Vol. 3*.

Zhao, Y., Nasrullah, Z., & Li, Z. (2019). Pyod: A python toolbox for scalable outlier detection. arXiv preprint arXiv:1901.01588.

Zhao, T., Wang, Z., Masoomi, A., & Dy, J. (2022). Deep Bayesian unsupervised lifelong learning. *Neural Networks*, *149*, 95–106.

Chapter 15

Active Lifelong Anomaly Detection with Experience Replay

In this paper, we proposed a framework for leveraging active learning to reduce the contamination of replay buffers in unsupervised class-incremental scenarios. The framework supports any memory-based experience replay method, any model, and any active learning strategy. We devised two strategies to select a limited number of data samples as queries for the oracle. In addition, the strategies analyze labeled anomalies to examine the replay buffer further to identify and remove additional data points that are likely to be anomalies.

Our experiments on three network and host-based intrusion detection datasets demonstrated that active lifelong anomaly detection can be a valuable asset in increasing the robustness of models and that the framework is able to improve anomaly detection performance working under a very limited budget.

Contributions of Kamil Faber: In this work, Kamil Faber co-designed and developed the general active lifelong anomaly detection framework, as well as querying and cleaning strategies. Kamil Faber carried out the experiments and was co-author of all sections of the paper.

This paper was published at the 2022 IEEE International Conference on Data Science and Advanced Analytics (DSAA; CORE A-Rank; 140 points).

Note: In this paper, we use the term “task” instead of “concept” as it was published before the term “concept” was created.

2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA) | 978-1-6654-7330-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/DSAA54385.2022.10032405

Active Lifelong Anomaly Detection with Experience Replay

Kamil Faber
Institute of Computer Science
AGH University of Science and Technology
Krakow, Poland
kfaber@agh.edu.pl

Roberto Corizzo
Department of Computer Science
American University
Washington, DC, USA
rcorizzo@american.edu

Bartlomiej Sniezynski
Institute of Computer Science
AGH University of Science and Technology
Krakow, Poland
bartlomiej.sniezynski@agh.edu.pl

Nathalie Japkowicz
Department of Computer Science
American University
Washington, DC, USA
japkowic@american.edu

***Abstract*—Anomaly detection tools present the potential to enhance defense policies and protection against different types of threats, supporting public safety and national security. Lifelong anomaly detection showcases new and challenging scenarios in which models are challenged to automatically adapt to changing conditions without forgetting past knowledge. However, the presence of anomalies in incoming data may significantly impact the robustness of models in such scenarios. Although active learning strategies could be an asset to increase model longevity and robustness, they have never been explored in this context. In this paper, we propose an active lifelong anomaly detection framework for class-incremental scenarios that supports any memory-based experience replay method, any query strategy, and any anomaly detection model. While experience replay allows models to consolidate past knowledge and simultaneously adapt to new knowledge, an active learning module reduces the number of anomalies memorized in the replay buffer. We propose two strategies that automatically identify and remove additional data points that are likely to be anomalies based on the oracle's feedback. Our experiments on popular host-based and network-based intrusion detection datasets show that our framework can improve the anomaly detection performance of models under low labeling budget constraints.**

***Index Terms*—anomaly detection, intrusion detection, lifelong learning, active learning, experience replay**

I. Introduction

Anomaly detection models represent important decision support tools in real-world domains and are an essential component in many cybersecurity applications. Relevant examples include detecting intrusions and attacks in network traffic and in smart grids, as well as detecting cyber threats such as distributed denial-of-service (DDoS) attacks, spamming, and attacks directed to mobile and IoT devices [1]–[4]. In this context, anomaly detection models have the potential to mitigate threats and protect both private companies and national security, fostering public safety at the environmental and at the infrastructure level.

Despite the progress made in recent years [5], [6] one crucial drawback of conventional anomaly detection models is that, by focusing on adaptation to rapidly changing environments, i.e. high plasticity, they are subject to forgetting past knowledge, i.e. limited stability. This issue results in the inability to handle recurring concepts that are common in real-world applications. Lifelong learning is an emerging machine learning paradigm that attempts to find a proper stability-plasticity trade-off involving cross-disciplinary knowledge in the fields of biology, neuroscience, and computer science [7]. Besides the fact that most of the research works in lifelong learning are directed to image classification and reinforcement learning [7], [8], lifelong learning for anomaly detection problems has the potential to yield sophisticated models that are capable of detecting anomalies while adapting to changing environments and avoiding forgetting knowledge acquired in the past [9]–[11].

A different thread of research is that of active learning for anomaly detection, which addresses the challenge of labeled background data availability in real-world settings. Since anomalies are likely to be present in raw background data, memory-based lifelong learning approaches such as experience replay may be significantly affected, since they are likely to incorporate anomalies in replay buffers used for model updates. In turn, models may be less robust, yielding a sub-optimal anomaly detection performance. Therefore, active learning in the context of lifelong anomaly detection has the potential to overcome this issue. However, to the best of our knowledge, there are still no studies that focus on combining the peculiar features of lifelong anomaly detection and active learning.

In this paper, we propose an active lifelong anomaly detection framework for class-incremental scenarios which supports any memory-based experience replay method, and any query strategy. While experience replay is adopted to consolidate past knowledge and simultaneously allow for adaptation to new knowledge, an active learning module aims to ensure that the consolidated past knowledge presents none or a limited

number of anomalies.

We devise two strategies to select a limited number of data samples as queries for the oracle. In addition, the strategies exploit queries labeled as anomalies to further explore the replay buffer in order to identify and remove additional data points that are likely to be anomalies. By doing so, they aim to remove a larger number of anomalies than the budget (i.e. the number of available queries or data points to be labelled by the oracle), thus reducing contamination in the replay buffer, which in turn yields more robust models. Our framework is model-agnostic, since it can be applied to any off-the-shelf semi-supervised or unsupervised anomaly detection model. In this work, we put particular focus on intrusion detection, since it is an essential asset to enhance defense policies, supporting public safety and national security. We evaluate our approach on three common host-based and network-based intrusion detection datasets. Experimental results show that our approach is capable of improving the anomaly detection performance of frequently adopted anomaly detection models under low labeling budget constraints.

The paper is structured as follows. Section II summarizes related works and describes lifelong learning scenarios. Section III describes our framework and proposed strategies. Section IV describes the experimental settings, baseline strategies and scenarios used in our study. Section V presents and discusses the results. Finally, Section VI concludes the paper summarizing the key results achieved and outlining ideas for future work.

II. Background

In this section, we describe works in the literature pertaining to our study. We first describe commonly adopted lifelong learning scenarios and provide a categorization of lifelong learning methods. We proceed with an overview on active learning methods. Finally, we focus on anomaly detection methods and models that incorporate an active learning component.

A. Lifelong learning scenarios

A lifelong learning setting usually consists of a continuous process in which a sequence of tasks is presented to the learning model [7], [12]. The goal of the learner is not only to adapt to the changing environment, but also to keep the knowledge acquired from the previously learned tasks. In this setting, the model is systematically evaluated on all tasks, to assess how well it performs in terms of adaptation and knowledge preservation. One popular scenario in image classification is the Split-MNIST dataset [13], which splits 10 classes of handwritten digit images in 5 binary tasks containing 2 digits each.

Scenarios are often categorized based on the availability of two key characteristics: task labels and task boundaries[1]. The availability of task boundaries implies that the model is aware during the learning phase that a new task occurs, and that new data comes from a new task, allowing the model to incorporate data and learn the new task. In addition to knowledge on task boundaries, the model may also be informed about which task is currently being presented (task label) during both the learning and the inference phases. Task-incremental scenarios provide the model with both task labels and task boundaries, which means that the model is aware that a new task is presented and knows which task is currently under evaluation. On the other hand, class-incremental scenarios provide the model with only task boundaries, which means that the model is aware of a task transition, but has no knowledge about which task is currently presented [12]. Similarly, class-incremental scenarios can be observed in recent lifelong anomaly detection works [10]. The main difference between anomaly detection and image classification in this context is that, in anomaly detection, a task provides a new characteristic of the normal class, instead of new classes that need to be classified. The goal of a lifelong anomaly detector is to adapt to the new behavior of the normal class, while still yielding satisfactory results on the previous characteristics of the normal class.

[1]Although more distinctive properties of lifelong learning scenarios can be identified, our discussion focuses on these two properties since they are the most pertinent for our study.

B. Lifelong learning methods

Lifelong learning methods are focused on improving the performance of neural networks in specific lifelong learning scenarios. We start with a general characterization of methods into three categories [7] and then we focus on the most related one for this study, i.e. experience replay.

The first category includes regularization-based methods that work by introducing constraints on weight updates during the incremental training of neural networks. One popular strategy is preventing the model from updating weights learned on previously trained tasks [14]. The second category focuses on dynamic architectures, in which the architecture of the network changes during the learning process. A notable example is expanding the network with new neurons or layers [15]. The third category, which is the most relevant for the research conducted in our study, includes replay-based methods, which recall prior data from previously seen tasks during the model update, to make sure that their characteristics are included in the updated model. There are two main categories of replay-based methods [7]. The first one focuses on selecting and storing the original data samples corresponding to learned tasks in the memory buffer. The stored data samples are then provided to the model in a so-called replay buffer. The second category is called Generative Experience Replay [16], as it leverages generative models to generate data reflecting the previously learned characteristics each time the model update process is executed. In our framework, we adopt a state-of-the-art combination of balanced replay buffers and reservoir sampling described and analyzed in [17]. While balanced replay buffer ensures that classes are equally (or almost equally) represented, reservoir sampling [18] decides which samples from each class should be remembered.

Most of the recent lifelong learning research is oriented to image classification and reinforcement learning. However,

lifelong anomaly detection methods are gaining momentum. One example of recent works is ARCADe, a meta-learning method that aims to identify the most appropriate parameter values for every single task in one-class image classification [10]. Another notable example that shows the real-life application of lifelong anomaly detection is a transfer learning method designed for detecting anomalies in surveillance videos [11]. Finally, a recent lifelong anomaly detection work closely related to online anomaly detection is CDPGA, which leverages change point detection and a dynamic architecture, presenting improved performance on new tasks [9]. Emerging studies confirm the usefulness of non-parametric change point detection in real-world applications [19], also to identify emerging tasks and recurrent tasks in time series data [20].

C. Active learning

The ability to gather enough labeled data to support training machine learning models represents an obstacle in many real-world domains. Labeled data is expensive to acquire and frequently relies on human labor. On the other hand, unlabeled data is abundant in many fields. The general challenge addressed by active learning methods is that of improving predictive models in scenarios with limited availability of labels, acquiring labels from an oracle for a limited portion of data samples. Active learning methods are composed by two primary components: an *oracle* that provides responses to a query, and a *query system*, which poses queries to the oracle [21]. Selecting the most appropriate data samples to be provided to the oracle is the most important capability of active learning methods. While methods based on *membership query synthesis* actively synthesize samples from the entire data space, *sequential sampling* methods draw samples from the underlying data distribution. The vast majority of methods in the literature, however, adopt a *pool-based sampling* approach, where a set of examples are labeled at a given time during the active learning process.

Querying strategies can be grouped in four main categories. *Heterogeneity-based models* aim at sampling from heterogeneous and uncertain regions of the data space. Examples include uncertainty sampling [22], query-by-committee [23], and expected model change [24]. *Performance-based models* tend to focus on the most unknown, noisy, or unrepresentative regions of the data space. One prominent example is expected error reduction [25], which minimizes the expected label uncertainty of data samples that are not yet labeled. *Representativeness-based models* tend to avoid querying queries on unrepresentative data samples or outliers, trying to assign a higher weight to dense regions of the input space, which are subject to queries [26]. *Hybrid models* attempt to combine multiple criteria for query selection, aiming at selecting data samples that are informative, i.e. close to the decision boundary of the learning model, and representative, i.e. part of a group of unlabeled data samples and unlikely to be obvious outliers.

D. Active learning for anomaly detection

Although most active learning methods are frequently focused on image classification problems, recent literature showed an increasing interest towards anomaly detection. The authors in [27] propose an active learning strategy in the context of network intrusion detection that extends the Support Vector Domain Description method (SVDD) by querying low-confidence observations in the data. The model is retrained after querying the labels for such observations, using unlabeled data in combination with newly labeled data. A different strategy is proposed in [28], where the most likely anomalous data samples identified by unsupervised models are selected for relabelling and are used retrain a neural network-based model. A query-by-committee active learning approach proposed in [29] is adopted to let the oracle judge cases of agreement and disagreement between multiple models in a distributed and cooperative learning environment. A semi-supervised VAE-based approach is proposed in [30], where active learning is used to select a small number of samples leveraging model uncertainty via the evidence of (variational) lower bound (ELBO). An ensemble learning approach is used in [31], where multiple models are used to predict anomaly scores and the maximum anomaly score is considered as oracle's feedback. Active learning strategies in the context of credit card fraud detection are explored in [32], including standard active learning, exploratory active learning, semi-supervised learning, and their combination. Results show that simple approaches such as highest risk querying noticeably improve model's accuracy, in opposition to exploratory and unsupervised active learning techniques. Another combined modeling approach is proposed in [33], where the authors propose the interaction between a supervised and an unsupervised model. Two active strategies are proposed: the former is based on model uncertainty, while the latter select the samples for which the two models are in disagreement. An extensive experimental study with one-class learning models in active anomaly detection has been conducted in [34], highlighting the validity of both random and principled strategies.

III. METHOD

In this paper, we focus on lifelong anomaly detection in a class-incremental scenario. In such scenario, given N tasks, $\mathbf{T}$ is a sequence of training sets, one for each task, and $\mathbf{E}$ is a sequence of evaluation data sets corresponding to each task. Anomaly detection models are presented with one training task at a time ($T_i \in T$) and their performance is evaluated on all $E_i \in E$. Models are updated using a replay buffer $R = \{R_1 \cup R_2 \cup \ldots \cup R_N\}, R_i \subset T_i$ which contains a summarized view of representative data points in each task, and allows the model to retain knowledge about previously learned tasks while incorporating knowledge about new tasks. The number of samples stored in the memory and therefore in the replay buffer is limited. This approach is in line with replay-based lifelong learning methods [17].

A. General framework for mitigating anomalies from replay buffers

A crucial issue in the context of unsupervised lifelong anomaly detection is that training data sets T_i may be contaminated, containing a varying rate of anomalies. Since the replay buffer R is built based on the original data for each task, anomalies in the data may result in anomalies in the replay buffer, which is used to update models. As a result, this issue may have influence on the inference capabilities of anomaly detection models during the evaluation phase, i.e. models may incorporate anomalies in the learned normal data distribution. Consequently, they may tend to flag anomaly data points as normal, resulting in an increased number of false negatives. This problem is exacerbated when anomalies from multiple tasks are included in data used to update models, which represents a more complex scenario than non-lifelong counterparts typically analyzed in the literature.

Our framework takes the replay buffer R_i as input and, each time a new task T_i is presented, it returns a cleaned version of the replay buffer $R_i^{'}$ in which possible anomalies are removed leveraging an active learning approach. Specifically, few data points from the replay buffers are selected for oracle's labelling, according to a desired budget B. Subsequently, an automated strategy takes place, exploiting oracle's feedback to remove a larger number of possible anomalies without performing additional queries. In principle, using $R_i^{'}$ for model update instead of R_i should yield a more robust anomaly detection model that is less subject to contamination. A graphical representation of the proposed framework is shown in Figure 1.

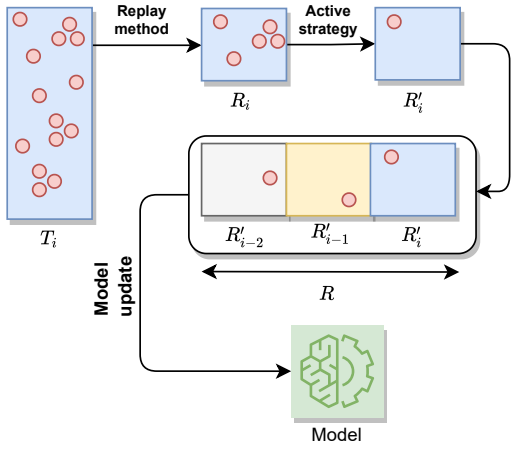


Fig. 1. Proposed framework for active lifelong anomaly detection. Data from each task T_i is provided to a *replay method* that yields a compact version of the task R_i. The *active learning strategy* reduces the number of potential anomalies (represented as red circles) residing in the replay buffer and yields R_i'. All tasks are included in R which is used for model update.

To accomplish this goal, we devise two distinct querying and cleaning strategies: Anomalous Cluster Removal strategy (ACR) and Similarity of Anomaly Scores strategy (SAS). While ACR analyzes the input feature space of data, SAS focuses on the analysis of anomaly scores returned by the model. Both strategies try to leverage the oracle's feedback to further clean the replay buffer, also including data points that are likely to be anomalies but were not included in oracle's queries.

B. Anomalous Cluster Removal strategy (ACR)

In the first step, the strategy leverages the input data space to cluster data points in the replay buffer. The k-Means algorithm is used to partition the available data points into k sets, using the original feature representation of data. The value for the k parameter depends on the available number of queries (budget B) and it determined according to the following equation:

$$k(B) = \min(B \cdot 10, \ |R_i|). \tag{1}$$

In the second step, the strategy randomly selects 10% of the number of clusters $k(B)$ generated according to the budget. The rationale for the choice of this percentage value is that we are interested in reducing the probability that heterogeneous data points (normal and anomalies) are included in the same cluster. That is why it is important to set the number of clusters to a much higher value than the budget B. This choice is important since, considering that clusters would be fully removed in the following step if labelled as anomaly, heterogeneous data points in the same cluster would imply that normal data points are removed from R. Therefore, the size of each cluster should be small enough to allow for a safe removal of potential anomalies. At the same, we aim at keeping the strategy parameter-less to facilitate its adoption. Each of the randomly selected clusters of data points is a candidate for elimination from the replay buffer. The cluster is eliminated if a point x_l closest to the center of the cluster has been labelled as anomaly by the oracle:

$$\begin{aligned} R_i' = R_i \setminus \{ \ x \ \mid \ & x \in R_i, \quad c(x) = c(x_l), \\ & O(q_l) = \text{anomaly} \\ & l \in \{0, 1, \dots B-1\}\}, \end{aligned} \tag{2}$$

where l describes the id of the query sent to the oracle, $O(x_l)$ the label assigned by the oracle and $c(x)$ is a function that returns the cluster id of a data point.

Intuitively, this strategy focuses on the exploitation of diversity and similarity in data leveraging the feature representation of input data. While diversity is ensured by the clustering step, which guarantees that different data points are assigned to different clusters, similarity is exploited in the cleaning step, which removes a full cluster of potential anomalies without human intervention and without defining strict heuristics or thresholding schemes. Moreover, the elimination step is supposed to exploit the existence of micro-clusters of anomalies in the data [35]. A graphical representation of the ACR strategy is shown in Figure 2.

C. Similarity of Anomaly Scores strategy (SAS)

Query strategies leveraging unsupervised models typically focus on selecting data points with the highest anomaly score

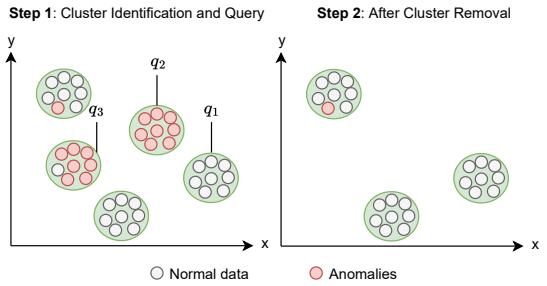


Fig. 2. ACR strategy: Step 1 is focused on clusters identification and selection of a small number of clusters (q_1, q_2, q_3), from each of which the data point closest to the cluster center is subject to oracle queries. Queries labelled as anomalies lead to the removal of the entire cluster they belong to.

as queries. However, this approach presents the drawback of not fully exploring the spectrum of the anomaly scores provided by the model, neglecting the fact that anomalies may also present low anomaly scores in model's predictions. In the first step, our proposed strategy queries data points based on their frequency within the full anomaly score range. We discretize the anomaly score range using an equal frequency approach [36], selecting a number of bins equal to the number of queries available. Subsequently, we select the data point to query, q_l, as the median data sample within each bin. More formally:

$$q_l = R_i^* \Big[(l + 0.5) \cdot \frac{|R_i|}{B \cdot 10} \Big], \tag{3}$$

where R_i^* is the list of samples in R_i sorted by anomaly scores by ascending order, $R_i^*[k]$ is k-th element of list R_i^* and $l = \{ 0, 1, \ldots, B-1 \}$ is an incremental index for the query.

Intuitively, this query strategy has the advantage to fully explore the anomaly score range, potentially identifying anomalies that were hidden in the lower range of anomaly scores, i.e. anomalies that the model was confidently predicting as normal. Moreover, this strategy allows us to implicitly exploit the confidence of unsupervised models without relying on an explicit measure of predictive confidence that is normally only available with supervised models such as Isolation Forest and Naive Bayes.

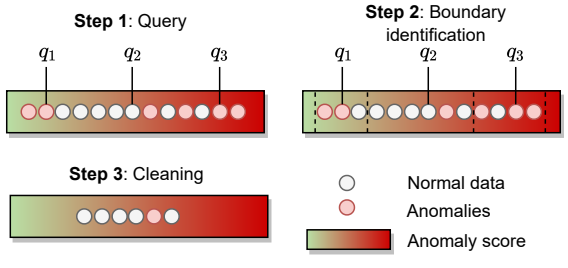


Fig. 3. SAS strategy: Step 1 queries data samples for the oracle with significantly different anomaly score values, taking into account the full range of anomaly scores returned by the model. Step 2 identifies boundaries for the neighborhood of each sample labelled as anomaly. Step 3 removes identified anomalies and their neighborhood from the replay buffer.

The second part of the strategy leverages data points labelled by the oracle and automatically labels additional data points that are close to those already labelled as anomalies. Our assumption is that, even if anomalies may present a diverse anomaly score from model's perspective, data points with a similar anomaly score of those labelled as anomalies are also likely to be anomalies, based on a locality criterion. Neighboring data points of labelled anomalies are selected to be removed, without additional oracle's intervention. More formally:

$$\begin{aligned} R_i' = R_i \setminus \{ x \mid \gamma(q_l) - \lambda \leq \gamma(x) \leq \gamma(q_l) + \lambda, \\ \lambda = \frac{b-a}{B \cdot 10}, \\ a = \min_{x \in R_i}(\gamma(x)), \quad b = \max_{x \in R_i}(\gamma(x)), \\ O(q_l) = \text{anomaly}, \\ l \in \{0, 1, \ldots, B-1\}, \}, \end{aligned} \tag{4}$$

where γ is the anomaly score returned by the model and $O(q_l)$ the label assigned by the oracle. A graphical representation of the SAS strategy is shown in Figure 3.

IV. Experimental Setup

Our experiments are designed to answer the following research questions:

- **RQ1:** To what extent does contamination in background data affect the anomaly detection performance of models?
- **RQ2:** Can our active lifelong anomaly detection framework boost the anomaly detection performance of models by properly removing anomalies from replay buffers?
- **RQ3:** In which scenarios do our active strategies improve model performance? Does a generalized optimal combination of active strategy and anomaly detection model exist?

To answer these questions, we perform a set of quantitative and qualitative experiments that assess the performance of models from different perspectives. In the following subsections, we describe the evaluation protocol followed in our study, including the metrics used for anomaly detection and lifelong learning, the datasets adopted and the learning scenarios addressed, as well as the competitor methods considered and their configuration. Results are discussed in Section V.

A. Evaluation protocol and metrics

We follow the evaluation protocol applied in most lifelong learning works [37] and described in Section II. One important aspect of the protocol is having access to training T and testing E sets for all tasks. The scenarios considered in this study are class-incremental, which means that the model is provided with information about the transition between tasks, but has no indication about which task is currently processed. In our experiments, all models are are presented with a sequence of training sets T and are updated after introducing each task T_i. After learning task T_i, the model is evaluated on testing sets from all tasks processed so far E_j, $j \leq i$. The scenario described so far is represented in Figure 4. This approach allows us to determine the performance of models

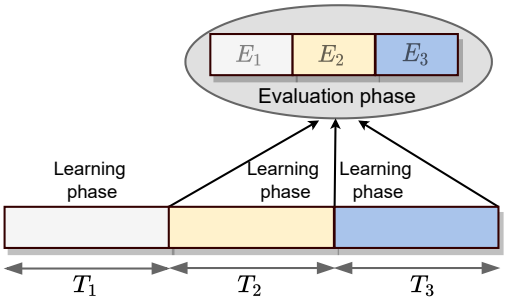


Fig. 4. Example of training sets representing a changing data distribution with tasks T_1, T_2, T_3 and corresponding testing sets E_1, E_2, E_3.

on previously learned tasks and therefore assess their ability to retain knowledge.

Our evaluation protocol produces a matrix $R_{N\times N}$, $N = |T|$ that gathers all obtained results. $R_{i,j}$ represents the ROC-AUC metric of the model evaluated on task j after learning task i. We showcase this process in Algorithm 1. The matrix $R_{N\times N}$ is then leveraged by us to compute the lifelong learning metrics.

Input: **T** – a sequence of N training sets
Input: **E** – a sequence of N testing sets
Input: L – learning model
Initialize results matrix $R_{N\times N}$
for $T_i \in$ **T do**
  Train L on T_i
  for $j \in \{0, 1, \ldots, i\}$ **do**
    Evaluate model L on data E_j and compute ROC-AUC
    Save results to $R_{i,j}$
  end
end
return R

Algorithm 1: Pseudo-code of the evaluation protocol

The most widely adopted metric in lifelong learning image classification is accuracy. However, this metric is not well-suited for anomaly detection due to the inherent differences in terms of problem definition and evaluation requirements among the two problems. One of the best suited metrics for assessing anomaly detection performance is ROC-AUC (Area Under Receiver Operating Characteristic Curve). A ROC plot shows the trade-off between the true negative rate and true positive rate. The essential advantage of ROC-AUC is being a threshold-free measure, which does not require a hard-set threshold and allows to assess the capabilities of the model in a much more complete way than threshold dependent metrics. In this work, we extend the definition of ROC-AUC to its lifelong version, so that it can properly measure the performance of the models on not only one task, but on all previously learned tasks after learning each new task. This approach conforms to the the definition of lifelong accuracy proposed in [37] and is defined as in the following:

$$\text{Lifelong ROC-AUC} = \frac{\sum_{i\geq j}^{N} R_{i,j}}{\frac{N(N-1)}{2}} \tag{5}$$

B. Datasets and scenarios

We perform experiments leveraging the following datasets:

- **NSL-KDD** [38] is a network-based intrusion dataset containing records of network traffic from the DARPA Intrusion Detection Systems program. It was built upon KDD-99 to solve the most important issues of its predecessor;
- **UNSW-NB15** [39] is a network-based intrusion dataset containing a mixture of modern network traffic from real modern setup and from synthesized malicious activities;
- **NGIDS-IDS** [40] is a host-based anomaly detection dataset created with modern security challenges in mind. It contains attacks from seven different families.

We create a few class-incremental lifelong learning scenarios based on the described datasets adopting the following procedure:

1) We cluster normal data into the desired number of clusters (reflecting the number of tasks N) leveraging a centroid-based clustering algorithm such as k-means.
2) We also cluster anomalies into the desired number of clusters.
3) We merge one cluster of normal class with its closest cluster of anomalies into a single task.
4) We separate each task into training and testing sets. The training set contains the specified number of anomalies according to a desired contamination rate C.

This procedure allows us to create class-incremental lifelong learning scenarios containing a specified number of tasks and contamination rate in the training set.

C. Methods

In our experiments, we use four unsupervised and semi-supervised anomaly detection models:

- **AE** [41]: Autoencoders are very popular neural network models that learn data distributions in a one-class manner by minimizing reconstruction error, which is defined as the difference between original and reconstructed data. The reconstruction error can be also used as an anomaly score, using the assumption that data from the originally learned distribution will present much lower reconstruction error than anomalies.
- **One Class Support Vector Machine (OC-SVM)** [42]: OC-SVM is a kernel-based semi-supervised method that aims to learn a separating hyperplane in a high-dimensional space. OC-SVM provides anomaly scores based on its comparison to the training distribution and its position inside the decision boundary.
- **Local Outlier Factor (LOF)** [43]: LOF is a method measuring the local density deviation of data with respect to its neighbors. The anomaly score is provided based on the ratio between the local density of the sample and the average local density of the nearest neighbors.

We also compare the two proposed ACR and SAS strategies with two commonly adopted baseline query strategies applied in non-lifelong active anomaly detection works:

- **Random (RND)** – this simple query strategy randomly selects B samples that are later labelled by an oracle and removed if they are truly anomalies. Despite being a very simple and completely random-based method, recent studies showed that it may yield promising results that outperform more complex strategies [34].
- **Highest Anomaly Score (HAS)** – this query strategy follows the assumption that the anomalous samples will have the highest anomaly score. In order to generate anomaly scores, the anomaly detection model (one of the four used in our study) is used in an unsupervised manner each time the strategy is applied. Once anomaly scores are predicted, the B samples with the highest value of anomaly score are labelled by the oracle and removed if they are true anomalies [32].

In our experiments, we adopt a state-of-the-art combination of balanced replay buffers and reservoir sampling [17]. The space in the replay memory is equally divided between each task T_i, while decision which samples from the specific task T_i should be kept is made leveraging reservoir sampling [18]. This approach ensures the proper balance between each task.

V. Results and Discussion

Results in Tables I, II, and III show the anomaly detection performance in terms of Lifelong ROC-AUC for all models (LOF, OC-SVM, AE) and strategies (Rand, HAS, ACR, SAS) with varying contamination rates (C) and budget levels (B). In the following subsections, we comments results in light of our research questions separately.

A. Impact of contamination on anomaly detection performance.

In this subsection our goal is to measure and analyze the impact of contamination on the anomaly detection performance of models in a class-incremental lifelong anomaly detection setting. Moreover, we showcase how the performance of different models can be improved by cleaning replay buffers from anomalies.

The results in Figure 5 show the performance of all models at varying contamination rates and assuming the ability to remove a different percentages of anomalies. Results are extracted adopting a budget-free variant of the HAS strategy, where the oracle receives queries until a desired amount of anomalies is removed $(20\%, 40\%, 60\%, 80\%, 100\%)$.

Results highlight some interesting patterns concerning the impact of data contamination on models' performance. Specifically, LOF has the potential to yield a high performance on all datasets, but its inferences are severely affected by contamination. As a result, its performance remains low when a moderate quantity of anomalies is removed $(20\% - 60\%)$, and it is strongly improved when the majority of the anomalies are removed $(80\% - 100\%)$.

Despite the fact that OC-SVM presents a poor performance on NGIDS independently from how many anomalies are present in the data, it is the most stable method with respect to contamination rates considering the other datasets. Indeed,

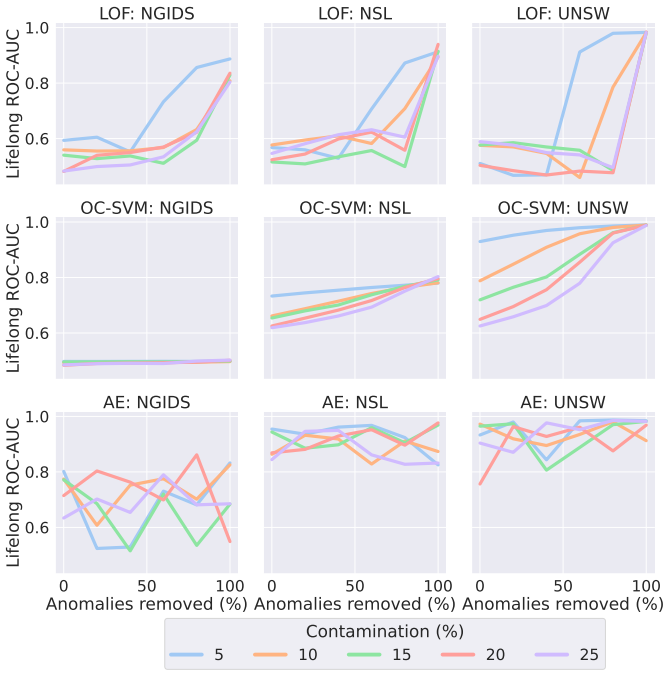


Fig. 5. Experimental study of the impact of contamination and replay buffer cleaning capabilities on the anomaly detection performance of all methods (LOF, OC-SVM, AE) and datasets (NGIDS, NSL, UNSW). Performance is measured in terms of Lifelong ROC-AUC with varying contamination rates (5%-25%) and simulating the removal of different percentages of anomalies from the replay buffer.

as more anomalies are removed, the performance of the model gradually improves, which denotes that the ability to remove any amount of anomalies is beneficial for the model.

AE generally exhibits a better performance when compared to LOF and OC-SVM, but its trend is less consistent than other methods with different percentages of removed anomalies. While on NSL and UNSW a non-monotonic increasing trend could still be observed, this does not occur with NGIDS, where AE's performance oscillates without a clear trend. This behavior is possibly due to the stochastic optimization of the model and the limited replay buffer size available for this dataset (500 samples per task).

Overall, our analysis show that contamination has a significant impact on the performance of anomaly detection models, and suggests that active learning strategies in the context of lifelong anomaly detection present a great potential to yield more robust models. Removing a sufficient amount of anomalies is positively correlated with model's performance to a varying degree based on model's characteristics. In some cases, removing a large number of anomalies is fundamental to make models successful, which suggests the importance of active learning strategies that adopt automated exploration and exploitation mechanisms in addition to oracle's queries (**RQ1**). However, they yield marginal improvements if the base models are unable to properly fit the characteristics of a specific dataset, as seen for OC-SVM with the NGIDS dataset.

TABLE I
EXPERIMENTAL RESULTS FOR THE NGIDS DATASET. LIFELONG ROC-AUC FOR ALL MODELS (LOF, OC-SVM, AE), CONTAMINATION RATES C, BUDGETS B, AND STRATEGIES (RND, HAS, ACR, SAS). THE BEST RESULTS FOR EACH COMBINATION OF METHOD, DATASET, BUDGET, AND CONTAMINATION RATE ARE SHOWN IN BOLD.

C	B	LOF				OC-SVM				AE			
		RND	HAS	ACR	SAS	RND	HAS	ACR	SAS	RND	HAS	ACR	SAS
0.05	0	0.593				0.497				0.615			
	5	0.530	0.593	0.591	**0.830**	**0.504**	0.502	0.492	0.501	0.595	0.612	**0.698**	0.566
	15	0.575	0.593	0.570	**0.831**	0.498	**0.502**	0.495	0.501	0.689	0.574	0.549	**0.723**
	25	0.560	0.593	0.546	**0.884**	0.503	**0.504**	0.497	0.503	0.552	0.701	**0.770**	0.620
	50	0.536	0.598	0.544	**0.887**	0.486	**0.505**	0.486	0.503	0.559	0.555	0.521	**0.782**
	100	0.547	0.609	0.546	**0.887**	0.501	**0.505**	0.502	0.503	0.469	0.555	**0.782**	0.688
0.10	0	0.559				0.493				0.707			
	5	0.524	0.559	0.503	**0.603**	**0.504**	0.492	0.489	0.492	0.513	0.540	0.366	**0.734**
	15	0.518	0.560	0.553	**0.785**	0.507	0.492	**0.508**	0.491	0.718	0.733	0.710	**0.785**
	25	0.558	0.561	0.557	**0.801**	**0.500**	0.498	0.498	0.491	**0.698**	0.546	0.389	0.451
	50	0.532	0.555	0.718	**0.806**	0.484	**0.501**	0.485	0.491	0.500	0.513	**0.683**	0.615
	100	0.521	0.576	**0.686**	0.805	0.478	**0.501**	0.481	0.491	0.470	0.556	0.755	**0.811**
0.15	0	0.540				0.496				0.530			
	5	0.516	0.540	0.561	**0.774**	**0.495**	0.492	0.482	0.492	0.712	**0.763**	0.535	0.555
	15	0.486	0.538	0.529	**0.819**	**0.495**	0.492	0.480	0.492	0.530	0.515	0.525	**0.639**
	25	0.507	0.540	0.494	**0.825**	0.490	**0.495**	0.489	0.491	**0.674**	0.559	0.532	0.618
	50	0.530	0.542	0.550	**0.825**	0.494	**0.498**	0.494	0.493	0.621	0.479	0.377	**0.698**
	100	0.521	0.513	0.553	**0.829**	0.487	**0.502**	0.487	0.499	0.529	0.553	0.561	**0.746**
0.20	0	0.482				0.484				0.475			
	5	0.471	0.481	0.485	**0.805**	0.495	**0.498**	0.492	**0.498**	**0.720**	0.632	0.532	0.613
	15	0.528	0.484	0.505	**0.834**	0.497	0.498	0.494	**0.499**	0.605	0.714	**0.785**	0.540
	25	0.517	0.486	0.657	**0.834**	0.496	**0.500**	0.488	0.499	0.443	0.411	0.531	**0.542**
	50	0.522	0.485	0.527	**0.834**	0.498	**0.504**	0.492	0.496	**0.604**	0.585	0.545	0.548
	100	0.543	0.507	0.539	**0.834**	0.486	**0.512**	0.485	0.498	0.496	0.519	0.352	**0.545**
0.25	0	0.483				0.485				0.447			
	5	0.504	0.483	0.536	**0.767**	0.485	0.484	**0.486**	0.482	0.446	0.512	**0.670**	0.550
	15	0.459	0.483	0.483	**0.796**	0.481	0.484	**0.497**	0.485	0.480	0.579	0.438	**0.587**
	25	0.457	0.488	0.482	**0.796**	0.483	**0.491**	0.484	0.487	**0.723**	0.510	0.645	0.647
	50	0.514	0.490	0.509	**0.796**	0.482	**0.495**	0.481	0.486	0.510	0.457	0.599	**0.715**
	100	0.484	0.485	0.677	**0.796**	0.482	**0.504**	0.480	0.499	0.479	**0.835**	0.399	0.576

TABLE II
EXPERIMENTAL RESULTS FOR THE NSL DATASET. LIFELONG ROC-AUC FOR ALL MODELS (LOF, OC-SVM, AE), CONTAMINATION RATES C, BUDGETS B AND STRATEGIES (RND, HAS, ACR, SAS). THE BEST RESULTS FOR EACH COMBINATION OF METHOD, DATASET, BUDGET, AND CONTAMINATION RATE ARE SHOWN IN BOLD.

C	B	LOF				OC-SVM				AE			
		RND	HAS	ACR	SAS	RND	HAS	ACR	SAS	RND	HAS	ACR	SAS
0.05	0	0.568				0.733				0.792			
	5	**0.604**	0.569	0.554	0.568	0.739	0.748	**0.749**	0.743	0.818	0.944	**0.947**	0.868
	15	0.547	0.570	0.569	**0.635**	0.735	**0.759**	0.740	0.743	0.818	0.831	0.835	**0.960**
	25	0.594	0.573	0.528	**0.666**	0.735	**0.766**	0.735	0.744	0.833	**0.928**	0.868	0.862
	50	0.587	0.582	0.588	**0.621**	0.740	**0.775**	0.739	0.747	0.857	0.825	**0.947**	0.824
	100	0.526	0.585	0.541	**0.644**	0.745	**0.781**	0.746	0.752	0.873	**0.931**	0.827	0.890
0.10	0	0.577				0.661				0.907			
	5	0.578	0.577	0.599	**0.684**	0.670	**0.684**	0.668	0.682	0.906	**0.942**	0.938	0.851
	15	0.551	0.578	**0.588**	0.587	0.673	**0.704**	0.672	0.687	0.852	0.831	**0.945**	0.828
	25	**0.586**	0.578	0.556	0.552	0.675	**0.717**	0.671	0.691	0.850	0.817	0.854	**0.886**
	50	0.621	0.588	**0.639**	0.575	0.682	**0.731**	0.683	0.693	0.862	0.925	0.921	**0.951**
	100	0.622	0.602	**0.626**	0.584	0.691	**0.760**	0.697	0.705	0.854	**0.956**	0.870	0.793
0.15	0	0.516				0.654				0.843			
	5	0.519	0.517	0.521	**0.642**	0.641	0.657	0.656	**0.666**	0.870	0.926	**0.928**	0.853
	15	0.500	0.518	0.551	**0.606**	0.659	**0.675**	0.659	0.668	**0.856**	0.821	0.796	0.855
	25	0.550	0.518	0.542	**0.688**	0.658	**0.683**	0.670	0.665	**0.934**	0.827	0.890	0.860
	50	0.449	0.520	0.483	**0.641**	0.653	**0.705**	0.656	0.669	0.924	0.857	0.934	**0.941**
	100	0.501	0.536	0.511	**0.622**	0.673	**0.744**	0.674	0.683	0.797	**0.964**	0.854	0.893
0.20	0	0.524				0.625				0.926			
	5	0.515	0.524	**0.527**	0.505	0.621	0.630	0.628	**0.636**	**0.913**	0.888	0.797	0.847
	15	0.545	0.523	**0.556**	0.524	0.628	0.634	0.639	**0.640**	0.844	0.901	0.857	**0.918**
	25	0.524	0.525	0.535	**0.541**	0.626	0.641	0.639	**0.643**	0.916	0.899	**0.940**	0.885
	50	0.532	0.527	0.514	**0.545**	0.634	**0.668**	0.639	0.653	0.918	**0.936**	0.867	0.917
	100	0.565	0.544	0.564	**0.606**	0.657	**0.695**	0.655	0.662	0.919	**0.937**	0.924	0.911
0.25	0	0.546				0.619				0.912			
	5	0.511	0.548	0.536	**0.693**	0.626	0.628	**0.644**	**0.644**	0.920	**0.944**	0.844	0.849
	15	0.528	0.551	0.551	**0.669**	0.627	0.632	0.634	**0.643**	0.830	0.858	0.850	**0.928**
	25	0.516	0.552	0.513	**0.720**	0.626	**0.634**	0.627	0.629	0.908	0.919	0.847	**0.925**
	50	0.531	0.553	0.538	**0.557**	0.633	0.654	0.629	**0.653**	**0.903**	0.836	0.897	0.819
	100	0.552	0.580	0.539	**0.656**	0.641	**0.684**	0.634	0.653	**0.921**	0.916	0.847	0.853

B. Impact of the framework on anomaly detection performance

Starting with the NGIDS dataset, results in Table I show that OC-SVM has overall a very poor performance at varying contamination rates, even though some improvements can be observed with any active learning strategy. For example, the ACR strategy improves the anomaly detection performance in terms of Lifelong ROC-AUC from 0.493 to 0.508 in the scenario with $C = 0.10$ and $B = 15$. AE presents a slightly better overall performance than OC-SVM, although it is still not satisfactory without any active learning strategy. However, a larger margin of improvement can be observed for all active strategies. For instance, for $C = 0.15$ and $B = 5$, the HAS strategy improves the performance from 0.530 to 0.763, which is a remarkable gain. However, the outcome of the different strategies is quite unstable since a clear trend with increasing levels of budget cannot be clearly identified. LOF presents the highest overall performance on this dataset, where active strategies show their best potential. For example, our SAS strategy with a very small budget of $B = 5$ allows to improve the performance from 0.484 to 0.805 in the scenario with a large contamination rate $C = 0.20$. Overall, one or more active strategies appear effective in improving the anomaly detection performance in 70 out of 75 combinations of models, contamination rates, and budgets with the NGIDS dataset.

On the NSL dataset, results in Table II show that our framework provides a large improvement across all methods and contamination rates. For example, with LOF, our SAS strategy with a budget $B = 25$ yields a strong improvement, increasing model's performance from 0.546 to 0.720 at a high contamination rate $C = 0.25$. Considering a representative example for OC-SVM, in the scenario with contamination rate $C = 0.15$, the HAS active strategy improves its performance from 0.654 to 0.744 using a moderate budget of $B = 100$. For AE, a remarkable case is that of contamination rate $C = 0.05$, where active strategies yield a considerable improvement under a minimal budget. For instance, ACS improves the performance from 0.792 to 0.947 with $B = 5$, whereas the SAS strategy improves it to 0.960 with $B = 15$. Overall, one or more active strategies appear effective in improving the anomaly detection performance in 71 out of 75 combinations of models, contamination rates, and budgets with the NSL dataset.

Shifting the focus on the UNSW dataset, results in Table III show that active strategies provide, in the average case, a moderately low improvement for LOF, and a more significant improvement for OC-SVM and AE. For example, the HAS strategy for OC-SVM allows to improve its performance from 0.788 to 0.943 and from 0.719 to 0.842 with a low budget $B = 25$ in the scenarios with contamination rate $C = 0.10$

TABLE III
EXPERIMENTAL RESULTS FOR THE UNSW DATASET. LIFELONG ROC-AUC FOR ALL MODELS (LOF, OC-SVM, AE), CONTAMINATION RATES C, BUDGETS B AND STRATEGIES (RND, HAS, ACR, SAS). THE BEST RESULTS FOR EACH COMBINATION OF METHOD, DATASET, BUDGET AND CONTAMINATION RATE ARE SHOWN IN BOLD.

C	B	LOF				OC-SVM				AE			
		RND	HAS	ACR	SAS	RND	HAS	ACR	SAS	RND	HAS	ACR	SAS
0.05	0	0.511				0.929				0.893			
	5	**0.531**	0.513	0.513	0.511	0.943	**0.971**	0.950	0.947	0.931	**0.984**	0.919	0.891
	15	**0.580**	0.515	0.516	0.546	0.944	**0.988**	0.944	0.947	0.899	**0.985**	0.921	0.895
	25	0.505	0.516	0.476	**0.592**	0.943	**0.989**	0.950	0.947	0.905	**0.986**	0.907	0.879
	50	0.520	**0.526**	0.502	0.500	0.945	**0.990**	0.947	0.953	**0.983**	0.859	0.822	0.826
	100	0.511	**0.535**	0.488	0.505	0.959	**0.990**	0.960	0.960	0.892	**0.927**	0.895	0.910
0.10	0	0.576				0.788				0.893			
	5	**0.577**	**0.577**	0.532	**0.577**	0.798	**0.834**	0.803	0.792	**0.978**	0.888	0.841	0.814
	15	**0.607**	0.581	0.558	0.575	0.812	**0.912**	0.812	0.799	0.879	0.890	**0.965**	0.788
	25	0.576	0.583	**0.608**	0.568	0.809	**0.943**	0.815	0.795	0.911	0.933	0.916	**0.978**
	50	0.557	**0.598**	0.560	0.548	0.821	**0.958**	0.827	0.805	**0.981**	0.910	0.895	0.890
	100	0.558	**0.613**	0.565	0.582	0.847	**0.971**	0.852	0.824	0.979	0.915	**0.980**	0.919
0.15	0	0.577				0.719				0.860			
	5	0.599	0.577	0.571	**0.584**	0.718	**0.756**	0.748	0.726	**0.971**	0.880	0.889	0.879
	15	0.561	0.577	0.601	**0.612**	0.743	**0.805**	0.753	0.730	0.829	**0.890**	0.870	0.768
	25	0.568	**0.582**	0.572	0.580	0.733	**0.842**	0.742	0.734	0.877	0.867	0.813	**0.885**
	50	**0.608**	0.590	0.601	0.564	0.741	**0.862**	0.738	0.737	0.881	**0.897**	0.887	0.747
	100	0.577	**0.628**	0.584	0.560	0.762	**0.906**	0.772	0.747	0.887	0.910	**0.976**	0.875
0.20	0	0.503				0.649				0.860			
	5	0.521	0.504	**0.547**	0.503	0.651	**0.676**	0.671	0.645	0.787	0.816	**0.864**	0.849
	15	0.470	0.506	**0.552**	0.514	0.651	**0.732**	0.658	0.646	0.815	**0.889**	0.733	0.736
	25	0.524	0.509	**0.542**	0.509	0.682	**0.762**	0.702	0.650	0.796	0.892	0.722	**0.966**
	50	0.512	**0.513**	0.509	0.489	0.653	**0.789**	0.670	0.663	0.846	0.783	**0.968**	0.846
	100	0.519	**0.538**	0.537	0.477	0.699	**0.821**	0.706	0.679	**0.971**	0.917	0.844	0.789
0.25	0	0.589				0.625				0.739			
	5	0.525	**0.589**	0.569	0.587	0.619	**0.655**	0.623	0.631	**0.958**	0.801	**0.958**	0.790
	15	0.522	**0.592**	0.545	0.575	0.620	**0.707**	0.660	0.626	0.846	0.833	**0.963**	0.806
	25	0.555	**0.594**	0.554	0.547	0.636	**0.727**	0.637	0.623	0.808	0.814	**0.959**	0.725
	50	0.557	**0.606**	0.547	0.531	0.650	**0.751**	0.637	0.631	0.961	0.858	0.867	**0.963**
	100	0.570	**0.629**	0.532	0.532	0.674	**0.777**	0.647	0.658	0.725	**0.909**	0.779	0.882

and $C = 0.15$, respectively. With the same budget, results with AE and the SAS strategy show an improvement from 0.893 to 0.978 and from 0.860 to 0.966 with contamination rate $C = 0.10$ and $C = 0.20$, respectively. The highest performance is achieved by the ACR strategy with a budget $B = 100$ and contamination rate $C = 0.10$, increasing model's performance from 0.893 to 0.980. Overall, one or more active strategies show to be effective in improving the anomaly detection performance in all 75 combinations of models, contamination rates, and budgets with the UNSW dataset.

In general, our experiments clearly show that our proposed active learning framework allows to improve the performance of models in the lifelong anomaly detection problem (**RQ2**). However, a moderate to large variability in the improvements can be observed across different datasets, as well as different contamination rates and budgets, and deserves further investigation.

C. Interaction between different strategies and methods

We analyze the performance of different strategies, and compare baseline strategies (RND and HAS) with our two proposed strategies (ACR and SAS).

Starting with the NGIDS dataset, results in Table I show that OC-SVM yields a poor performance, showcasing the inability to learn a proper separation boundary, which makes drawing considerations about the effectiveness of active strategies very challenging. On the other hand, the SAS strategy strongly improves the performance for LOF in all budget and contamination configurations, significantly outperforming other strategies. For AE, the situation is more uneven, and it is difficult to highlight a clear winning active strategy. However, our SAS strategy improves baseline strategies by a significant margin in the majority of the cases (15 out of 25), whereas ACR improves baselines in 10 out of 25 cases.

On the NSL dataset, results in Table II show that the HAS strategy is the best for OC-SVM (19 out of 25 cases), and the largest margin is achieved with higher budget values. However, all active strategies yield a similar performance gain. For LOF, it is difficult to highlight the best strategy. However SAS improves baselines in the majority of the cases (19 out of 25), whereas ACR improves baselines in 10 out of 25 cases, both with a varying margin of performance gain. For AE, baseline strategies are the best in 12 cases out of 25, whereas our proposed strategies yield the best performance in 13 out of 25 cases.

Shifting the focus on the UNSW dataset, the HAS strategy surprisingly yields the highest performance for OC-SVM in all 25 configurations, indicating that the anomaly scores provided by the model on data in the replay buffer represent a reliable way to isolate anomalies from normal data in this dataset. LOF appears to yield a very weak performance on this dataset, and all active strategies appear to be insufficient to improve its inferences. For AE, at low contamination rates, HAS presents a very good performance. On the other hand, for high contamination rates and low budgets, ACR is the preferred strategy. For example, with $C = 0.25$ and $B = 15$, ACR allows to boost the performance from 0.739 to 0.963.

Overall, taking into consideration the datasets where the base models can achieve reasonable results, OC-SVM seems to integrate better with simpler strategies such as HAS, whereas LOF yields a significantly stronger performance with the SAS strategy. AE presents a very good ability to successfully incorporate all active strategies to boost anomaly detection performance (**RQ3**).

VI. CONCLUSION

In this work, we explored active learning in the context of the lifelong anomaly detection problem. We proposed a framework for class-incremental scenarios that supports any memory-based experience replay method and any query strategy. Leveraging experience replay, the framework allows models to consolidate past knowledge while they adapt to new knowledge. Our active learning module further enhances this capability reducing the number of anomalies in the replay buffer. We performed experiments on three network and host-based intrusion detection datasets and involving four active learning strategies. Results showed that active lifelong anomaly detection can be a valuable asset to increase the longevity and robustness of models, and that our proposed framework is able to improve the anomaly detection performance of different models under low labeling budget con-

straints. As future work, we aim to investigate how active learning anomaly detection impacts model forgetting on previously learned tasks. Additional research directions include the adoption of curriculum learning and multi-dataset learning capabilities in lifelong anomaly detection models.

REFERENCES

[1] B. Daley, E. Serra, A. Cuzzocrea, Identifying malicious users in the offshore leaks networks via structural node representation learning, in: 2021 IEEE International Conference on Big Data (Big Data), IEEE, 2021, pp. 5095–5101.

[2] J. Carpenter, J. Layne, E. Serra, A. Cuzzocrea, Detecting botnet nodes via structural node representation learning, in: 2021 IEEE International Conference on Big Data (Big Data), IEEE, 2021, pp. 5357–5364.

[3] K. Faber, L. Faber, B. Sniezynski, Autoencoder-based ids for cloud and mobile devices, in: 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), 2021, pp. 728–736.

[4] R. Corizzo, E. Zdravevski, M. Russell, A. Vagliano, N. Japkowicz, Feature extraction based on word embedding models for intrusion detection in network traffic, Journal of Surveillance, Security and Safety 1 (2) (2020) 140–150.

[5] Z. Li, Y. Zhao, N. Botta, C. Ionescu, X. Hu, Copod: copula-based outlier detection, in: 2020 IEEE International Conference on Data Mining (ICDM), IEEE, 2020, pp. 1118–1123.

[6] Y. Zhao, X. Hu, C. Cheng, C. Wang, C. Wan, W. Wang, J. Yang, H. Bai, Z. Li, C. Xiao, et al., Suod: Accelerating large-scale unsupervised heterogeneous outlier detection, Proceedings of Machine Learning and Systems 3 (2021).

[7] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review, Neural Networks 113 (2019) 54 – 71.

[8] D. Abel, Y. Jinnai, S. Y. Guo, G. Konidaris, M. Littman, Policy and value transfer in lifelong reinforcement learning, in: International Conference on Machine Learning, 2018, pp. 20–29.

[9] R. Corizzo, M. Baron, N. Japkowicz, Cpdga: Change point driven growing auto-encoder for lifelong anomaly detection, Knowledge-Based Systems (2022) 108756.

[10] A. Frikha, D. Krompaß, V. Tresp, Arcade: A rapid continual anomaly detector, in: 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, 2021, pp. 10449–10456.

[11] K. Doshi, Y. Yilmaz, Continual learning for anomaly detection in surveillance videos, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, 2020, pp. 254–255.

[12] G. M. Van de Ven, A. S. Tolias, Three scenarios for continual learning, arXiv preprint arXiv:1904.07734 (2019).

[13] F. Zenke, B. Poole, S. Ganguli, Continual learning through synaptic intelligence, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, 2017, pp. 3987–3995.

[14] A. Sharif Razavian, H. Azizpour, J. Sullivan, S. Carlsson, Cnn features off-the-shelf: an astounding baseline for recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2014, pp. 806–813.

[15] T. Diethe, T. Borchert, E. Thereska, B. Balle, N. Lawrence, Continual learning in practice, NeurIPS Continual Learning Workshop (2018).

[16] H. Shin, J. K. Lee, J. Kim, J. Kim, Continual learning with deep generative replay, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., 2017.

[17] P. Buzzega, M. Boschini, A. Porrello, S. Calderara, Rethinking experience replay: a bag of tricks for continual learning, in: 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, 2021, pp. 2180–2187.

[18] J. S. Vitter, Random sampling with a reservoir, ACM Trans. Math. Softw. 11 (1) (1985) 37–57.

[19] K. Faber, R. Corizzo, B. Sniezynski, M. Baron, N. Japkowicz, Watch: Wasserstein change point detection for high-dimensional time series data, in: 2021 IEEE International Conference on Big Data (Big Data), IEEE, 2021, pp. 4450–4459.

[20] K. Faber, R. Corizzo, B. Sniezynski, M. Baron, N. Japkowicz, Lifewatch: Lifelong wasserstein change point detection, in: 2022 International Joint Conference on Neural Networks (IJCNN), IEEE, 2022.

[21] C. C. Aggarwal, X. Kong, Q. Gu, J. Han, S. Y. Philip, Active learning: A survey, in: Data Classification, Chapman and Hall/CRC, 2014, pp. 599–634.

[22] D. D. Lewis, J. Catlett, Heterogeneous uncertainty sampling for supervised learning, in: Proceedings of the 11th International Conference on Machine Learning (ICML), 1994.

[23] P. Melville, R. J. Mooney, Diverse ensembles for active learning, in: Proceedings of the 21st International Conference on Machine Learning (ICML), 2004, p. 74.

[24] B. Settles, M. Craven, S. Ray, Multiple-instance active learning, Advances in neural information processing systems 20 (2007).

[25] N. Roy, A. McCallum, Toward optimal active learning through sampling estimation of error reduction, in: Proceedings of the 18th International Conference on Machine Learning (ICML), 2001, pp. 441–448.

[26] B. Settles, M. Craven, An analysis of active learning strategies for sequence labeling tasks, in: proceedings of the 2008 conference on empirical methods in natural language processing, 2008, pp. 1070–1079.

[27] N. Görnitz, M. Kloft, K. Rieck, U. Brefeld, Active learning for network intrusion detection, in: Proceedings of the 2nd ACM workshop on Security and artificial intelligence, 2009, pp. 47–54.

[28] T. Pimentel, M. Monteiro, A. Veloso, N. Ziviani, Deep active learning for anomaly detection, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, 2020, pp. 1–8.

[29] M. Guarascio, N. Cassavia, F. S. Pisani, G. Manco, Boosting cyber-threat intelligence via collaborative intrusion detection, Future Generation Computer Systems (2022).

[30] T. Huang, P. Chen, R. Li, A semi-supervised vae based active anomaly detection framework in multivariate time series for online systems, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 1797–1806.

[31] X. Tang, Y. S. Astle, C. Freeman, Deep anomaly detection with ensemble-based active learning, in: 2020 IEEE International Conference on Big Data (Big Data), IEEE, 2020, pp. 1663–1670.

[32] F. Carcillo, Y.-A. Le Borgne, O. Caelen, G. Bontempi, Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization, International Journal of Data Science and Analytics 5 (4) (2018) 285–300.

[33] D. Labanca, L. Primerano, M. Markland-Montgomery, M. Polino, M. Carminati, S. Zanero, Amaretto: An active learning framework for money laundering detection, IEEE Access (2022).

[34] H. Trittenbach, A. Englhardt, K. Böhm, An overview and a benchmark of active learning for outlier detection with one-class classifiers, Expert Systems with Applications 168 (2021) 114372.

[35] M. Goldstein, S. Uchida, A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, PloS one 11 (4) (2016) e0152173.

[36] Y. Yang, G. I. Webb, X. Wu, Discretization methods, in: Data mining and knowledge discovery handbook, Springer, 2009, pp. 101–116.

[37] N. Díaz-Rodríguez, V. Lomonaco, D. Filliat, D. Maltoni, Don't forget, there is more than forgetting: new metrics for continual learning (2018). arXiv:1810.13166.

[38] M. Tavallaee, E. Bagheri, W. Lu, A. A. Ghorbani, A detailed analysis of the kdd cup 99 data set, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6.

[39] N. Moustafa, J. Slay, Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), in: 2015 Military Communications and Information Systems Conference (MilCIS), 2015, pp. 1–6.

[40] W. Haider, J. Hu, J. Slay, B. Turnbull, Y. Xie, Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling, Journal of Network and Computer Applications 87 (2017) 185–192.

[41] S. Sengupta, S. Basak, P. Saikia, S. Paul, V. Tsalavoutis, F. Atiah, V. Ravi, A. Peters, A review of deep learning with special emphasis on architectures, applications and recent trends, Knowledge-Based Systems 194 (2020) 105596.

[42] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, J. C. Platt, Support vector method for novelty detection, in: Advances in neural information processing systems, 2000, pp. 582–588.

[43] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, 2000, pp. 93–104.

Chapter 16

Distributed Continual Intrusion Detection: A Collaborative Replay Framework

In this work, we designed a novel collaborative continual intrusion detection framework that addresses the limitations of currently proposed lifelong methods. Moreover, we also proposed collaborative experience replay supporting the sharing of knowledge gathered across different nodes, leading to the improvement in the robustness of the anomaly detection model robustness. The framework is built upon independent services with clear responsibilities, supporting a modular, flexible, scalable architecture.

Our experimental evaluation showcased the efficacy of our collaborative framework when compared to non-collaborative and non-continual learning strategies. Our adaptive double-balanced replay technique proved its efficiency while working under strict resource limitations.

Contributions of Kamil Faber: In this work, Kamil Faber designed and developed the distributed continual intrusion detection framework, as well as collaborative replay. Kamil Faber carried out the experiments and was co-author of all sections of the paper.

This paper was published at the 2023 IEEE International Conference on Big Data (CORE B-Rank; 70 points).

Note: In this paper, we started using the term “continual” instead of “lifelong” due to its raising popularity.

Distributed Continual Intrusion Detection: A Collaborative Replay Framework

Kamil Faber*, Bartlomiej Sniezynski*, Roberto Corizzo*†
* AGH University of Krakow
Krakow, Poland
kfaber@agh.edu.pl, bartlomiej.sniezynski@agh.edu.pl
† American University
Washington DC, USA
rcorizzo@american.edu

2023 IEEE International Conference on Big Data (BigData) | 979-8-3503-2445-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/BigData59044.2023.10386211

***Abstract*—Intrusion Detection System is a strategic analytical tool for the security of organizations and institutions. Among existing approaches, distributed and collaborative intrusion detection approaches are particularly effective since they combine data analysis from multiple sources to provide increased model robustness. Although many state-of-the-art approaches have the ability to adapt to evolving environments and incoming data, they are subject to catastrophic forgetting of past knowledge. At the same time, recent works in lifelong continual anomaly detection showcase the merit of simultaneous adaptation and knowledge retention. However, lifelong methods are thus far limited to the analysis of a single data source and do not provide distributed and collaborative learning capabilities. In this paper, we fill this gap by proposing a novel distributed continual learning intrusion detection framework with collaborative experience replay. The system is built from independent Detection Nodes and a Continual Learning Center. While the nodes are in charge of data selection and intrusion detection, the Continual Learning Center implements a collaborative replay strategy, performs model updates, and broadcasts the most recent model to the nodes. The separation of responsibilities allows for the decomposition of the system into task-oriented services, leading to a modular, flexible, and scalable architecture. An extensive evaluation involving popular network intrusion detection datasets shows the potential of our framework and the improvement in detection performance that can be achieved with the collaborative replay strategy.**

***Index Terms*—Continual learning, Lifelong learning, Anomaly detection, Intrusion detection, Collaborative learning**

I. Introduction

Intrusion detection systems are crucial tools in ensuring the security of organizations and institutions [1]. Monitoring network activities for suspicious behavior enables the opportunity to raise security alerts and take appropriate countermeasures when intrusions are detected. Relevant examples of real-world settings include monitoring both essential infrastructure, such as data centers, smart grids, and IoT devices, as well as everyday appliances, such as smart homes or mobile devices [2]–[5]. In this context, intrusion detection systems have the potential to improve the safety of both private companies and national security. The research community explored a wide diversity of intrusion detection methods in the past decades [6]. The most popular ones include *signature-based approaches*, which are focused on detecting known malicious patterns, and *anomaly-based approaches*, which rely on the detection of behavior that significantly differs from the normally observed one [7].

Among *anomaly-based approaches*, machine learning methods represent the dominant class of ongoing research due to their high potential to yield intelligent inferences that go beyond pre-defined rules and patterns [7]. Moreover, in the intrusion detection context, it is often unreasonable to assume the availability of knowledge about attacks due to the quickly evolving nature of data [8]. As a result, significant interest has been recently devoted to semi-supervised and unsupervised machine learning, which contemplate the possibility of limited to no knowledge of the anomaly class. In this realm, one-class learning models have shown to be particularly robust and flexible since they are capable of detecting previously unknown attacks such as 0-days [6], which is a required condition in the constantly evolving landscape of threats.

Considering the rapidly evolving nature of network traffic, there is a need for models to be reactive and dynamic as conditions change over time [8]. Online learning approaches appear particularly suitable for this goal but are excessively geared towards adaptation to new information, which, on the other hand, leads to forgetting past knowledge [9]. Continual/lifelong[1] learning approaches address this limitation since their goal is to simultaneously deal with adaptation to new information and retention of knowledge acquired in the past [10]. By doing so, they present the ability to model a comprehensive view of the environment, resulting in more robust inference capabilities [11]. However, state-of-the-art continual learning approaches for intrusion detection lack the exploitation of multi-sourced information, commonly found in network infrastructures, and which exploitation was shown to be beneficial for model accuracy in non-lifelong contexts

The paper was supported by funds of the Polish Ministry of Science and Higher Education allocated to the AGH University, and by program "Excellence initiative - research university" for AGH University.

[1]The terms "continual" and "lifelong" are commonly used interchangeably. For the sake of consistency, from now on, we will use "continual".

[12], [13]. Being restricted to the analysis of a single source represents a limitation for existing approaches, which may result in reduced predictive performance. This aspect creates a gap in the current state-of-the-art, which could be addressed by empowering continual learning methods for intrusion detection with collaborative learning capabilities.

To this aim, in this paper, we propose a novel collaborative continual intrusion detection framework that leverages multi-sourced data. The key aspect of our framework is a collaborative adaptive double-balanced replay strategy, which provides a comprehensive view of traffic data collected at multiple nodes and allows the model to preserve past knowledge while effectively adapting to the environment. By leveraging combined information occurring at multiple nodes, our approach allows to build a more accurate model that entails inter-node dynamics and patterns, which can potentially facilitate the recognition of normal behavior previously observed at other nodes and a more robust detection of abnormal patterns. The proposed framework is designed with well-defined responsibilities (data selection, collaborative replay, model update, intrusion detection, and model broadcasting), allowing them to be devised as independent services, fostering a modular, flexible, and scalable architecture. An extensive evaluation involving popular network intrusion detection datasets shows the potential of our approach and the improvement in performance that can be achieved with the introduction of collaborative continual learning capabilities for intrusion detection.

The paper is structured as follows. Section II discusses related works in intrusion detection and continual learning. Section III describes our proposed approach. Section IV describes the experimental setup and discusses the extracted results. Section V wraps up the paper with a summary of the results and a description of possible directions for future work.

II. Background

In this section, we first describe distributed and collaborative intrusion detection. Then, we introduce continual learning and its application to intrusion detection.

A. Intrusion Detection in Service-Oriented Collaborative Setup

Intrusion detection is a crucial tool in ensuring organizations' security, and, therefore, it attracts a lot of interest, resulting in a significant number of published works [14], [15]. Despite significant efforts, the research in intrusion detection systems faces a lot of arising challenges, such as explainability [16] and adversarial attacks [17].

We can also observe an emerging trend to extend intrusion detection systems to distributed service-oriented architectures that bring the advantage of scalability and flexibility [18].

The authors of [19] propose a novel approach to virtualizing intrusion detection systems as microservices that can be customized, shared, and scaled independently. A microservice-oriented intrusion detection is also explored in [20], where the authors propose an intra/inter-class-structure-based variational few-shot learning model for edge devices in distributed IoT. Moving to distributed intrusion detection creates the opportunity to leverage multi-agent systems [21] and blockchain technology [22].

Another relevant research thread is collaborative intrusion detection, where knowledge from multiple sources is used to provide a broader perspective and perform more robust inferences [23]. The authors in [24] propose a deep learning-based intrusion detection algorithm based on the generative adversarial networks where edge network traffic is analyzed by several intrusion detection models. The work in [23] leverages generative adversarial networks to design a collaborative intrusion detection system for VANETs, where multiple SDN controllers jointly train a global model for the entire network. The study in [25] presents an intrusion detection approach where nodes are allowed to share information in the context of Blockchain technology.

B. Continual Lifelong Learning

Real-world machine learning systems experience a continuous stream of information and very often rapidly changing conditions. That is why they are required to continuously learn and adapt. The ability to learn new skills while retaining previously acquired knowledge is known as continual or lifelong learning [10]. They are inherently different from online learning, as continual learning goals are twofold: adapting and retaining knowledge, while online learning models focus only on adaptation and are prone to catastrophic forgetting [26]. State-of-the-art works in continual learning can be broadly categorized as regularization-based, dynamic architectures, and replay-based [10]. While regularization-based methods introduce constraints on weight updates in neural networks during training, dynamic architecture methods adapt the model architecture during the learning process as they encounter new tasks. Replay-based methods, the closest to the approach proposed in our paper, selectively store data samples from previously learned tasks in compact memory buffers that can be used for model updates.

The adoption of continual learning in intrusion detection settings is quickly attracting attention in recent research. However, it is still underrepresented compared to other settings, such as image classification or reinforcement learning. One of the first works, [27], proposes a generative anomaly detection approach with experience replay. Another work leveraging experience replay, [28], focuses on detecting transitions in a task-agnostic setting with lifelong change point detection and adapts a VAE anomaly detection model accordingly with a replay-based model update strategy that leverages a hierarchical memory. Authors in [29] showcased that active learning may improve the performance of replay-based lifelong anomaly detection.

One disadvantage of existing lifelong intrusion detection approaches is that they focus on intrusion detection with the assumption that there is only a single data source, without considering distributed scenarios which are much more likely to be found in real-world intrusion detection scenarios. In this work, we aim to overcome this limitation by bridging the

world of continual learning with that of collaborative intrusion detection.

III. Method

In this section, we explain the specifics of our framework. First, we provide a system-level overview, which entails our collaborative continual intrusion detection framework. Second, we describe our proposed replay strategy in detail, providing a formalization of the approach and highlighting its advantages.

A. System-level overview

The architecture of our proposed framework consists of multiple Intrusion Detection Nodes and a Continual Learning Center, each of which has specific responsibilities realized in the form of independent services. Conceptually, Detection Nodes monitor network traffic in a well-defined scope of network devices. It is noteworthy that network traffic monitoring capabilities are general and can handle a variety of sources, such as a conventional network segment, as well as IoT devices, and cloud infrastructures. Every Detection Node is responsible for detecting intrusions in the monitored network traffic with the use of a machine learning model which is shared across Detection Nodes. On the other hand, the Continual Learning Center is responsible for ensuring that the model is updated and promptly broadcasted to Detection Nodes, ensuring that they are able to carry out the intrusion detection task according to the most recently available information. To do that, the Continual Learning Center stores a collaborative experience replay buffer constructed leveraging selected data transferred from Detection Nodes. The architecture of the proposed framework is visually presented in Figure 1.

Going into details about the specific responsibilities of the framework, the *Data selection Service* selects data that should be transferred to the Continual Learning Center, allowing us to minimize the network transfer overhead of the architecture and to focus on the most salient information observed at single nodes. It is a feasible assumption since network traffic data is often characterized by recurring patterns, which only require to be represented once in order to be incorporated by the models. To this end, we leverage Reservoir Sampling [30] as an efficient way to select data points from network traffic monitored by Detection Nodes. Intuitively, Reservoir sampling assigns a probability p to each sample and selects the samples with the highest probability given a budget constraint. It is noteworthy that no sensitive data has to be transferred anywhere outside the local Detection Nodes, as the framework is able to leverage summarized network flows.

Selected data is transferred to the *Collaborative Replay Service*. Replay is fundamental in continual strategies to ensure that models are constantly updated with newly observed data without losing knowledge (i.e. forgetting) acquired from previously observed data [10]. To this aim, a replay buffer is a summarized representation of data used for model updates, and an effective replay strategy is fundamental to ensure that this goal is achieved. The *Collaborative Replay Service* leverages the combined data collected from multiple nodes and decides which data samples should be stored in the *Replay Data Storage* and recalled for future model updates. Further details about the replay strategy are provided in the subsection III-B. Leveraging a subset of data points allows for small storage requirements for the replay data. It is worth noting that the size of the replay buffer can be customized according to a desired granularity based on domain characteristics and storage availability.

Each time a new significant change in network traffic characterization is presented, the framework has to update the model accordingly to reflect the relevant changes and yield accurate inferences. This capability is realized by the *Model Update Service*, which takes the replay data buffer and the last available version of the model as input. It is important to note that our framework is model-agnostic since it supports any machine learning-based anomaly detection model. In our work, we adopt one-class learning models, which are commonly used in anomaly detection and do not require knowledge of anomalous patterns. The updated model is broadcasted to *Intrusion Detection Service* in all Detection Nodes, allowing them to leverage the newly acquired collaborative knowledge. Having one model broadcasted to all Detection Nodes avoids the computational requirements that would be required if every Detection Node was in charge of training and maintaining an independent node-specific model. At the same time, sharing knowledge across nodes provides Detection Nodes with a global view of network traffic, which, in principle, corresponds to a more robust intrusion detection based on the most recent available information. It is noteworthy that, in the beginning, the framework requires gathering data needed for the initial model training stage. After that, the framework iteratively processes new data and responds to changes following the above-described workflow.

All these capabilities are designed in the form of independent services, which allows for clear isolation of the framework's responsibilities. While some services are machine learning models-oriented (Model Update, and Intrusion Detection), others are data-intensive (Data Selection, Collaborative Replay strategy). Our framework allows for flexible deployment of the different services using specific hardware resources. For instance, machine learning models can be deployed on GPU, whereas data-intensive services can be deployed on CPU. From the analytical viewpoint, the microservices-oriented distributed intrusion detection architecture supports achieving a high degree of robustness since each service can be deployed independently on computing nodes and replicated, ensuring high availability even if certain nodes become unreliable or unresponsive.

B. Replay strategy

This subsection formally describes our novel adaptive double-balanced collaborative experience replay strategy. To this aim, we first describe the learning scenario in our framework. Let us define a network as a set of N Detection Nodes. Each node k observes a sequence of concepts $C_1^k, C_2^k, \ldots,$ where each concept corresponds to a self-consistent behavior

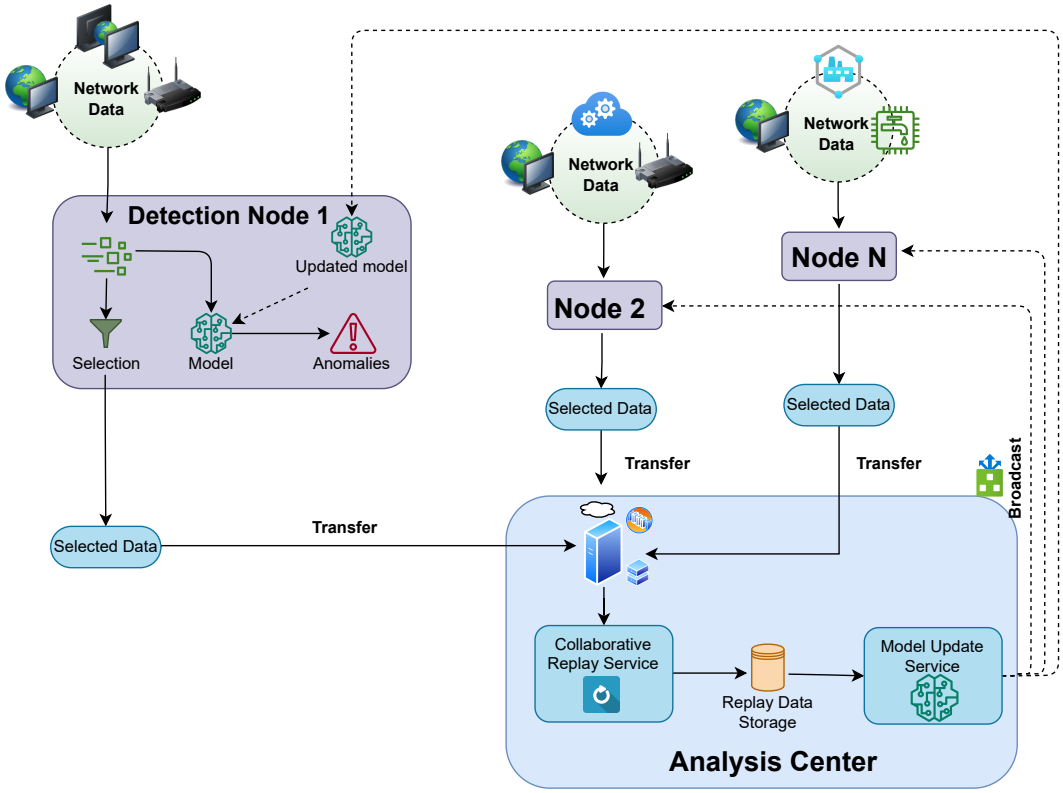


Fig. 1. Proposed architecture for collaborative continual intrusion detection consisting of four types of services: Data Selection, Intrusion Detection, Collaborative Replay, and Model Update.

or characteristic that can be identified in network traffic. A new concept may emerge when a significant change in the normal behavior of the network traffic occurs and requires the model to adapt in order to keep accurate detection. The challenge in this continual learning scenario is to incorporate new concepts in the model while preserving knowledge acquired from previously observed concepts. To this end, a continual learning strategy has to be adopted.

In this work, we devise an experience replay strategy that entails collaborative knowledge sharing across Detection Nodes. The rationale of our collaborative replay is that of learning a more robust model that entails inter-node dynamics and patterns. Such information has the potential to infuse a broader representation of network traffic patterns, which benefits from knowledge about concepts locally observed at specific Detection Nodes. Collaborative incorporation of such knowledge in the broadcasted model should, in principle, allow for more robust detection of intrusions than using a single source of information. The replay buffer allows storing a summarized version of all concepts observed so far and leverages this knowledge to prevent forgetting while updating the model.

Our proposed replay strategy is double-balanced, which means that the available space is equally divided between different Detection Nodes and then equally divided between the concepts from every Node. The space division and balancing are carried out adaptively without any prior knowledge about the number of Nodes or concepts. Figure 2 shows the outcome of our replay strategy with rebalancing realized by the *Collaborative Replay Service*, involving the addition of a new concept for an existing Node and a new Node.

More formally, our replay buffer consists of multiple parts based on the number of nodes and concepts observed so far. For each concept i from node k, we keep its summarized version R_i^k as part of the replay buffer. The complete experience replay buffer R can be defined as:

$$R = \{R_i^k \ ; \ \forall k \in \{0, 1, \ldots, N\}, \forall i \in \{0, 1, \ldots, c_k\}\}, \quad (1)$$

where c_k is the number of concepts observed so far at Detection Node k.

The entire replay buffer has a constant size, i.e., budget B, that depends on the available storage resources. The overall budget in continual learning replay is usually relatively small in comparison to observed data [28]. In our work, the budget for each observed concept is adaptively calibrated depending on the total budget B. Specifically, the budget for a single concept in Detection Node k is based on the number of nodes N currently attached to the network and on the number of concepts c_k observed for node k:

$$\beta(N, c_k) = \frac{B}{N * c_k} \quad (2)$$

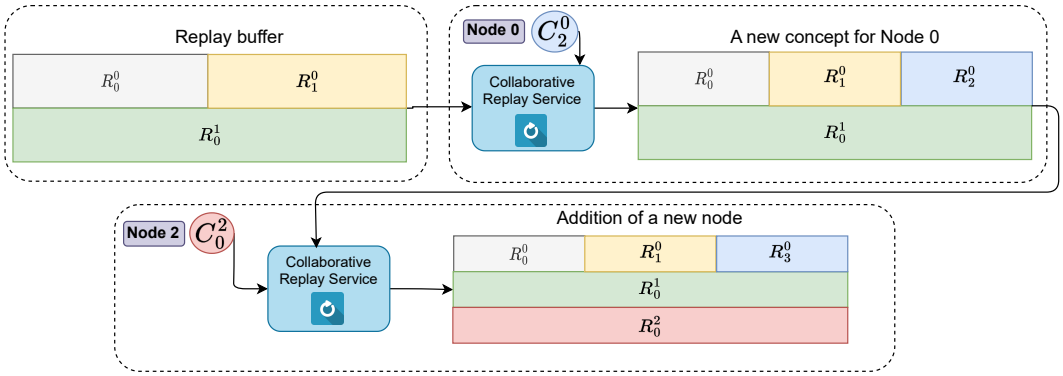


Fig. 2. Example of our adaptive double-balanced replay strategy. When a new concept for node 0 is presented, the strategy resizes the space of the other two concepts for the same node. When a new node is added, the replay service accommodates it by resizing all existing concepts from all nodes.

As a result, each time data from a new concept is presented, the replay buffer is adaptively rebalanced to accommodate a new concept. The algorithm performs rebalancing at two levels of abstraction: within-node and across-nodes.

Input: $\mathbf{X^k}$ – new data from node k
Input: N – the total number of nodes in the network so far
Input: $C = \{c_0, c_1, \ldots, c_N\}$ – a number of concepts observed so far for each node $0, 1, \ldots, N$
Input: R – experience replay built so far (see Eq. 1)
1 **if** $k > N$ **then** // New node (not yet present in replay buffer)
2 **for** $l \in \{0, 1, \ldots, N\}$ **do**
3 $B^l \leftarrow \beta(N+1, c_l)$ // Budget for each concept in node l; Eq: 2
4 $R^l \leftarrow Alg.2(R^l, B^l)$ // Resize concepts from node l
5 **end**
6 $B^k \leftarrow \beta(N+1, 1)$ // Budget for new node
7 $R^k_0 \leftarrow rs(\mathbf{X^k})$ // Add new concept with Reservoir sampling (rs)
8 $N \leftarrow N+1$
9 $c_k \leftarrow 1$
10 **else** // Node already present in replay buffer
11 $B^k \leftarrow \beta(K, c_k+1)$ // Budget for each concept in node k; Eq: 2
12 $R^k \leftarrow Alg.2(R^k, B^k)$ // Resize concepts from node k
13 $R^k_{(c_k+1)} \leftarrow rs(X^k, B^k)$ // Add new concept for node k
14 $c_k \leftarrow c_k + 1$
15 **end**
16 **return** R, N

Algorithm 1: Adaptive double-balanced experience replay algorithm.

Algorithm 1 describes how the replay behaves after receiving new data $\mathbf{X^k}$ originating from concept k. The algorithm operates differently depending on whether the source node k already has allocated space in the replay buffer or was never observed before. In the former case, intuitively, we need to accommodate a new node by reducing the space allocated for all other nodes (lines 2-5). Subsequently, the data for the newly added node is added to the replay buffer (lines 6-7). We leverage Reservoir sampling rs [30] to select data samples according to the budget. This leads to an increase in the number of nodes present in the network (line 8) and setting the number of concepts for node k to 1 (line 9). In the latter case, the algorithm re-balances the concepts observed within the existing node k by reducing the space allocated to existing concepts (lines 11-12) and adding a new concept (line 13). This leads to an increase in the number of concepts present in node k (line 14).

Algorithm 2 presents the process of within-node rebalancing for node k. Within-node rebalancing is required when a new concept is observed at an existing node in the network. Intuitively, it scales down buffer R^k_i for each summarized concept i for node k leveraging Reservoir sampling rs [30] to select data samples according to budget $B^k = \beta(N, c_k)$.

Input: $\mathbf{R^k} = \{R^k_0, R^k_1, \ldots\}$ – replay buffer for node k
Input: B^k – budget for every concept in node k
Output: $\mathbf{R'^k}$ – rebalanced and resized replay buffer
1 **for** $R^k_i \in R^k$ **do**
2 $R'^k_i \leftarrow rs(R^k_i, B^k)$ // Scale down using reservoir sampling
3 **end**
4 $R'^k \leftarrow \{R'^k_i, R'^k_1, \ldots\}$
5 **return** R'^k

Algorithm 2: Within-node rebalancing algorithm.

From the double-balanced feature viewpoint, the main advantage of our experience replay is to attribute the same importance to information collected from multiple nodes. Moreover, in the presence of a varying number of concepts assigned to multiple Detection Nodes, this approach allows us to avoid dominant or vanishing nodes (significantly more/less represented in the replay buffer due to an imbalanced number of concepts between nodes).

Moreover, the adaptive feature of our replay strategy presents two main advantages: *i)* It allows for full utilization of capacity as soon as the first concept is presented; *ii)* It does not require prior knowledge about the number of concepts presented in the learning scenario, contrarily to non-adaptive replay strategies.

IV. Experiments

Our experiments are directed at answering the following research questions:

- **RQ1**: Is the knowledge-sharing capability of collaborative continual anomaly detection effective in providing an increased performance of models in comparison to the non-collaborative approach?
- **RQ2**: Does our proposed adaptive double-balanced collaborative replay strategy provide effective results while operating under limited resource constraints for data transfer and storage?
- **RQ3**: Do continual learning strategies provide more robust detections for collaborative intrusion detection problems in comparison to updating the model with the most recent data?
- **RQ4**: What is the effectiveness of the different base learners (OC-SVM, COPOD, LOF, IF, VAE) in the collaborative continual intrusion detection setting?

A. Evaluation protocol and metrics

We follow the standard continual anomaly detection evaluation protocol devised in [28] and adapt it to distributed intrusion detection. The model is presented with a sequence of sets of concepts. Each set contains at most one concept from each node. The sequence can be formalized as:

$$\mathbf{C} = \{\{C_0^0, C_0^1, \ldots, C_0^N\}, \ldots \{C_k^0, C_k^1, \ldots, C_k^N\}\}, \quad (3)$$

where k represents the last observed concept observed. The model is first trained on a set of concepts $C_i = \{C_i^0, C_i^1, \ldots, C_i^N\}$ and then evaluated on all test sets of concepts observed so far $C_0, \ldots, C_i$. Contrarily to the standard online learning evaluation protocol, which only evaluates model performance on new tasks, the adopted protocol, which is the standard in continual learning, considers the performance not only on new, but also on previously learned concepts and, therefore, measures their ability to retain knowledge over time. The metric we use to assess model performance is Lifelong ROC-AUC, defined as:

$$\text{Lifelong ROC-AUC} = \frac{\sum_{i \geq j}^{N} R_{i,j}}{\frac{N(N+1)}{2}}, \quad (4)$$

where $R_{i,j}$ is the ROC-AUC performance on set of concepts j after learning set of concepts i, averaged across all nodes N. We adopt ROC-AUC as the base evaluation metric due to its threshold-free nature which allows for a more comprehensive assessment of the model considering all possible thresholds.

B. Models, datasets, and scenarios

In our experiments, we adopt five diverse popular anomaly detection models: *i)* One-Class Support Vector Machines (OC-SVM) [31] that compares new data with a hyperplane-based boundary learned from training data, *ii)* Copula-based Outlier Detection (COPOD) [32] that leverages tail probabilities of an empirical copula, *iii)* Local Outlier Factor (LOF) [33] that compares the ratio between the local density of new data and the average density of its nearest neighbors, *iv)* Isolation Forest (IF) [34] that leverages ensembles of trees and determines the anomaly score based on the length of the path from the root to the specific leaf, and *v)* Variational Auto-Encoder (VAE) [35] – a neural-network reconstruction-based model with generative capabilities.

We conduct experiments with three popular datasets for intrusion detection: i) CICIDS2017 [36] – modern dataset based on currently common attack families and modern network traffic characteristic; ii) UNSW [37] – dataset containing records describing a hybrid of the modern normal traffic and the contemporary synthesized attack activities; and iii) NSL-KDD [38] – an enhanced version of the commonly adopted KDD-99, proposed to solve the problems of its predecessor.

We extract continual learning scenarios from the raw datasets adopting the concept-incremental setting and three types of scenarios (CC, CR, R) devised in [39]. More in detail, the scenarios include concepts obtained using different clustering variants, such as clusters of anomalies assigned to the closest normal data cluster (CC), clusters of anomalies randomly assigned to normal data clusters (CR), and anomalies randomly assigned to normal data clusters (R). We adopt concept-incremental scenarios [39], where a model is not aware of what concept is exactly processed but knows that a new set of concepts is being presented and, therefore, has to learn them. It is noteworthy that this design creates additional complexity compared to the standard evaluation protocol in intrusion detection since the model is challenged with simultaneous adaptation and knowledge retention.

In addition to extracting results with our proposed replay strategy, we adopt two common continual learning baselines: *i)* **Naive**: Similarly to online learning, the Naive strategy updates the model with newly observed data without considering any explicit mechanism to preserve past knowledge; *ii)* **Cumulative**: It accumulates all data observed so far and updates the model accordingly. It is an unrealistic upper bound since it assumes unlimited resources, but it is useful as a comparison against resource-aware learning strategies. In our experiments, we present Naive and Cumulative strategies in both non-collaborative and collaborative settings.

C. Results discussion

In this section, we discuss the experimental results of our work. All results in Tables I, II, and III show two Lifelong ROC-AUC performance scores for each base model (OC-SVM, COPOD, LOF, IF, VAE) and scenario (CC, CR, R). The performance scores for each base model and scenario indicate

TABLE I
RESULTS IN TERMS OF LIFELONG ROC–AUC WITH THE **CUMULATIVE** STRATEGY FOR ALL DATASETS AND BASE MODELS IN NON-COLLABORATIVE (N-C) AND COLLABORATIVE (C) SETUP.

	OC-SVM		COPOD		LOF		IF		VAE	
	N-C	C	N-C	C	N-C	C	N-C	C	N-C	C
CICIDS (CC)	0.614	0.685	0.458	0.526	0.621	0.993	0.624	0.707	0.622	0.735
CICIDS (CR)	0.625	0.732	0.388	0.488	0.611	0.993	0.645	0.728	0.660	0.825
CICIDS (R)	0.530	0.620	0.386	0.476	0.572	0.995	0.707	0.797	0.604	0.714
NSL-KDD (CC)	0.761	0.793	0.646	0.814	0.560	0.856	0.795	0.907	0.762	0.852
NSL-KDD (CR)	0.845	0.874	0.761	0.899	0.525	0.889	0.879	0.945	0.843	0.906
NSL-KDD (R)	0.925	0.954	0.418	0.857	0.526	0.898	0.929	0.970	0.907	0.955
UNSW (CC)	0.682	0.725	0.474	0.516	0.641	0.890	0.539	0.605	0.594	0.709
UNSW (CR)	0.710	0.759	0.431	0.498	0.654	0.907	0.531	0.640	0.599	0.776
UNSW (R)	0.565	0.566	0.399	0.429	0.630	0.900	0.502	0.558	0.536	0.620

TABLE II
RESULTS IN TERMS OF LIFELONG ROC–AUC WITH THE **ADAPTIVE DOUBLE-BALANCED REPLAY** STRATEGY FOR ALL DATASETS AND BASE MODELS IN NON-COLLABORATIVE (N-C) AND COLLABORATIVE (C) SETUP.

	OC-SVM		COPOD		LOF		IF		VAE	
	N-C	C	N-C	C	N-C	C	N-C	C	N-C	C
CICIDS (CC)	0.612	0.711	0.458	0.394	0.621	0.982	0.617	0.679	0.592	0.716
CICIDS (CR)	0.623	0.781	0.388	0.302	0.611	0.984	0.643	0.754	0.632	0.722
CICIDS (R)	0.529	0.641	0.386	0.306	0.572	0.996	0.705	0.798	0.550	0.685
NSL-KDD (CC)	0.761	0.781	0.646	0.627	0.560	0.875	0.795	0.908	0.760	0.893
NSL-KDD (CR)	0.845	0.874	0.761	0.738	0.525	0.896	0.878	0.947	0.844	0.929
NSL-KDD (R)	0.925	0.950	0.418	0.359	0.526	0.894	0.929	0.971	0.907	0.964
UNSW (CC)	0.682	0.735	0.474	0.439	0.641	0.777	0.536	0.631	0.560	0.805
UNSW (CR)	0.710	0.779	0.431	0.386	0.654	0.831	0.527	0.653	0.584	0.828
UNSW (R)	0.565	0.573	0.399	0.374	0.630	0.864	0.501	0.563	0.529	0.669

TABLE III
RESULTS IN TERMS OF LIFELONG ROC–AUC WITH **NAIVE** STRATEGY FOR ALL DATASETS AND BASE MODELS IN NON-COLLABORATIVE (N-C) AND COLLABORATIVE (C) SETUP.

	OC-SVM		COPOD		LOF		IF		VAE	
	N-C	C	N-C	C	N-C	C	N-C	C	N-C	C
CICIDS (CC)	0.556	0.667	0.426	0.467	0.551	0.699	0.558	0.652	0.555	0.609
CICIDS (CR)	0.566	0.708	0.352	0.418	0.524	0.702	0.575	0.650	0.567	0.680
CICIDS (R)	0.481	0.581	0.348	0.434	0.472	0.767	0.593	0.699	0.484	0.551
NSL-KDD (CC)	0.730	0.774	0.584	0.675	0.620	0.633	0.720	0.862	0.743	0.830
NSL-KDD (CR)	0.811	0.865	0.658	0.806	0.641	0.584	0.794	0.925	0.824	0.895
NSL-KDD (R)	0.890	0.944	0.334	0.493	0.668	0.571	0.862	0.964	0.896	0.946
UNSW (CC)	0.562	0.668	0.457	0.491	0.590	0.658	0.500	0.594	0.554	0.620
UNSW (CR)	0.572	0.680	0.413	0.475	0.615	0.665	0.505	0.583	0.565	0.646
UNSW (R)	0.500	0.572	0.379	0.414	0.572	0.648	0.444	0.558	0.494	0.577

the performance without collaborative knowledge sharing (N-C) and with collaborative knowledge sharing (C). Results are averaged across five hyperparameter configurations for all methods except for COPOD due to its parameterless nature[2].

The results in Table I allow us to assess whether the knowledge-sharing capability of collaborative continual anomaly detection is effective in providing increased performance for continual intrusion detection models (**RQ1**). We can observe consistent behavior across all base models and scenarios, where model performance with collaborative knowledge sharing (right) is improved over model performance without knowledge sharing (left). The degree of improvement varies by a large amount across all experiments. On average, looking at aggregated performance across the three scenarios (CC, CR, R), we observe an improvement of 27.08% (from 0.58 to 0.73 for CICIDS), 20.64% (from 0.74 to 0.89 for NSL-KDD), and 18.98% (from 0.57 to 0.67 for UNSW). The smallest improvement can be identified with OC-SVM and the UNSW (R) scenario (from 0.565 to 0.566). However,

[2]For conciseness, we do not report standard deviation in the tables. The average standard deviation for Cumulative is: 0.015, for Adaptive Double-Balanced Replay: 0.016, and for Naive: 0.014.

it can be noticed that OC-SVM does not perform well in general with this scenario, while other methods, such as LOF, present a remarkably high improvement in the same scenario (from 0.63 to 0.90). Based on these results, we conclude that collaborative knowledge sharing is beneficial for continual intrusion detection models (**RQ1**).

Continuing our results analysis, we assess whether our proposed adaptive double-balanced collaborative replay strategy can provide effective results while operating under limited resource constraints for data transfer and storage **(RQ2)**. To this aim, we conducted experiments with our adaptive double-balanced collaborative replay strategy by significantly restricting computational resources: *i)* data transfer – 200 samples for each node and concept; and *ii)* experience replay buffer size – 200 samples per node (800 total for UNSW and CICIDS, 1200 total for NSL-KDD; less than 2% of total data). We present the results in Table II, where we observe a general pattern across 4 out of 5 base models and scenarios where model performance with the proposed double adaptive experience replay strategy with collaborative knowledge sharing (right) outperforms the variant without collaborative knowledge sharing (left). This behavior is not observed for COPOD as a base model, where the combination of collaborative knowledge sharing and the replay strategy does not yield satisfactory results. We attribute this behavior to the limited number of data samples allowed for each node in the replay buffer, which may not be sufficiently large for COPOD to yield an improved performance. In fact, this behavior was not observed with the cumulative strategy. On average, looking at aggregated performance across the three scenarios (CC, CR, R), we observe an improvement of 22.39% (from 0.57 to 0.70 for CICIDS), 13.77% (from 0.74 to 0.84 for NSL-KDD), and 17.61% (from 0.56 to 0.66 for UNSW). Comparing our extracted results with the cumulative strategy (Table I) allows us to measure the existing gap between the proposed learning strategy and an unrealistic upper bound without memory constraints. We observe that the gap is minimal: 5.38% for CICIDS, 6.05% for NSL-KDD, and 1.92% for UNSW. The results lead to the consideration that our collaborative adaptive double-balanced replay strategy is effective in yielding an increased predictive performance of a diverse set of intrusion detection models over its non-collaborative counterpart while operating under limited budget constraints (**RQ2**).

Another critical perspective is the analysis of whether continual learning strategies for collaborative intrusion detection provide more robust detection in comparison to constantly updating the model with the most recent data, as done in the Naive strategy (**RQ3**). Comparing the results for the Naive strategy in Table III to the results with the Cumulative strategy (Table I), we can observe that the performance for all base models on all scenarios is significantly higher with Cumulative strategy. This behavior is expected since the naive strategy has a natural inclination to forget, which leads to detrimental effects in continual settings. It is noteworthy that collaborative knowledge-sharing is still helpful and brings improvement over a non-collaborative approach, even in a non-continual Naive strategy. Another analysis pertains to the comparison between Naive and the results with our proposed replay strategy presented in Table II. Results show that the performance of our replay strategy is generally better than the Naive strategy. This result is remarkable, considering that the Naive strategy has no limitation in terms of the number of samples used in the data selection, while our replay strategy operates under strict constraints for data transfer and storage. This result highlights that continual learning strategies for collaborative intrusion detection are more effective than strategies that constantly update models with the most recent data due to their ability to retain past knowledge while fostering adaptation to new data (**RQ3**).

In addition to the previous analysis, we also investigate the effectiveness of the different base learners when presented with the collaborative continual intrusion detection setting (**RQ4**). Results in Table I show that LOF presents the strongest performance across all datasets and scenarios. We attribute this result to the inductive bias in LOF, which is based on nearest-neighbor search and matches the criteria used for scenario generation devised in [39]. VAE is the second-best method, with excellent robustness on NSL-KDSS and moderate to high results on CICIDS and UNSW. OC-SVM and IF follow in the ranking, with moderately high results on all datasets. COPOD presents good performance on NSL-KDD and weaker performance on the other two datasets. The trends observed in Table I with the cumulative strategy can also be identified with the double adaptive replay strategy, as shown in Table II. Overall, based on the performance observed in our results, base models appear suitable for collaborative continual intrusion detection replay strategy (**RQ4**). This result is particularly interesting, considering that such base models, popular in one-class learning and anomaly detection, were never assessed before in the context of collaborative continual intrusion detection. However, it is worth noting that performance could be improved in some scenarios, such as UNSW (R), and requires further investigation.

V. Conclusion

In this work, we proposed a novel collaborative continual intrusion detection framework along with a collaborative experience replay strategy and a clear separation of responsibilities of the framework into independent microservices. Our work fills the gap at the intersection of collaborative intrusion detection and continual learning, which was never explored before. Our novel collaborative replay strategy has the ability to share knowledge of multiple concepts observed at different nodes, enhancing the anomaly detection model's robustness by considering multi-node dynamics. Our extensive experiments show the effectiveness of our collaborative framework in comparison to non-collaborative and non-continual learning strategies. Moreover, our adaptive double-balanced replay strategy proved to yield a positive improvement while operating under strict resource limitations.

In future work, we will investigate new smart data selection strategies for collaborative continual intrusion detection, as

well as the effectiveness of different replay strategies. Moreover, we will devise new, more complex scenarios that will provide new challenges for base models.

REFERENCES

[1] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, Survey of intrusion detection systems: techniques, datasets and challenges, Cybersecurity 2 (1) (2019). doi:10.1186/s42400-019-0038-7.

[2] R. Corizzo, E. Zdravevski, M. Russell, A. Vagliano, N. Japkowicz, Feature extraction based on word embedding models for intrusion detection in network traffic, Journal of Surveillance, Security and Safety 1 (2) (2020) 140–150.

[3] J.-M. Storm, J. Hagen, O. A. Arntzen Toftegaard, A survey of using process data and features of industrial control systems in intrusion detection, in: Proceedings - 2021 IEEE International Conference on Big Data, Big Data 2021, 2021, p. 2170 – 2177. doi:10.1109/BigData52589.2021.9671325.

[4] S. Shaikh, C. Rupa, G. Srivastava, T. Reddy Gadekallu, Botnet attack intrusion detection in iot enabled automated guided vehicles, in: 2022 IEEE International Conference on Big Data (Big Data), 2022, pp. 6332–6336. doi:10.1109/BigData55660.2022.10020355.

[5] S. Ramapatruni, S. N. Narayanan, S. Mittal, A. Joshi, K. Joshi, Anomaly detection models for smart home security, in: 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), IEEE, 2019, pp. 19–24.

[6] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, Survey of intrusion detection systems: techniques, datasets and challenges, Cybersecurity 2 (1) (2019) 1–22.

[7] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, F. Ahmad, Network intrusion detection system: A systematic study of machine learning and deep learning approaches, Transactions on Emerging Telecommunications Technologies 32 (1) (2021) e4150.

[8] A. Alshamrani, S. Myneni, A. Chowdhary, D. Huang, A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities, IEEE Communications Surveys and Tutorials 21 (2) (2019) 1851 – 1877. doi:10.1109/COMST.2019.2891891.

[9] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al., Overcoming catastrophic forgetting in neural networks, Proceedings of the national academy of sciences 114 (13) (2017) 3521–3526.

[10] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review, Neural Networks 113 (2019) 54 – 71.

[11] A. Cossu, A. Carta, V. Lomonaco, D. Bacciu, Continual learning for recurrent neural networks: an empirical evaluation, Neural Networks 143 (2021) 607–627.

[12] G. Folino, P. Sabatino, Ensemble based collaborative and distributed intrusion detection systems: A survey, Journal of Network and Computer Applications 66 (2016) 1–16.

[13] K. Faber, L. Faber, B. Sniezynski, Autoencoder-based ids for cloud and mobile devices, in: 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), 2021, pp. 728–736.

[14] A. L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Communications Surveys & Tutorials 18 (2) (2016) 1153–1176. doi:10.1109/COMST.2015.2494502.

[15] Z. Yang, X. Liu, T. Li, D. Wu, J. Wang, Y. Zhao, H. Han, A systematic literature review of methods and datasets for anomaly-based network intrusion detection, Computers & Security 116 (2022) 102675. doi:https://doi.org/10.1016/j.cose.2022.102675.

[16] S. Neupane, J. Ables, W. Anderson, S. Mittal, S. Rahimi, I. Banicescu, M. Seale, Explainable intrusion detection systems (x-ids): A survey of current methods, challenges, and opportunities, IEEE Access 10 (2022) 112392–112415.

[17] H. Ali Alatwi, C. Morisset, Threat modeling for machine learning-based network intrusion detection systems, in: 2022 IEEE International Conference on Big Data (Big Data), 2022, pp. 4226–4235. doi:10.1109/BigData55660.2022.10020368.

[18] D. Malani, J. Modi, S. Lilani, Y. Desai, R. Dhanare, Implementation of intrusion detection systems in distributed architecture, in: International Conference on Cyber Warfare, Security and Space Research, Springer, 2021, pp. 144–155.

[19] N. Zhang, H. Li, H. Hu, Y. Park, Towards effective virtualization of intrusion detection systems, in: Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, 2017, pp. 47–50.

[20] W. Liang, Y. Hu, X. Zhou, Y. Pan, I. Kevin, K. Wang, Variational few-shot learning for microservice-oriented intrusion detection in distributed industrial iot, IEEE Transactions on Industrial Informatics 18 (8) (2021) 5087–5095.

[21] O. Achbarou, M. A. El Kiram, O. Bourkoukou, S. Elbouanani, A multi-agent system-based distributed intrusion detection system for a cloud computing, in: New Trends in Model and Data Engineering: 2018 International Workshops, October 24–26, Proceedings 8, Springer, 2018, pp. 98–107.

[22] R. Kumar, P. Kumar, R. Tripathi, G. P. Gupta, S. Garg, M. M. Hassan, A distributed intrusion detection system to detect ddos attacks in blockchain-enabled iot network, Journal of Parallel and Distributed Computing 164 (2022) 55 – 68, cited by: 24. doi:10.1016/j.jpdc.2022.01.030.

[23] L. Nie, Y. Wu, X. Wang, L. Guo, G. Wang, X. Gao, S. Li, Intrusion detection for secure social internet of things based on collaborative edge computing: a generative adversarial network-based approach, IEEE Transactions on Computational Social Systems 9 (1) (2021) 134–145.

[24] J. Shu, L. Zhou, W. Zhang, X. Du, M. Guizani, Collaborative intrusion detection for vanets: A deep learning-based distributed sdn approach, IEEE Transactions on Intelligent Transportation Systems 22 (7) (2020) 4519–4530.

[25] G. Gurung, G. Bendiab, M. Shiaele, S. Shiaeles, Cids: collaborative intrusion detection system using blockchain technology, in: 2022 IEEE International Conference on Cyber Security and Resilience (CSR), IEEE, 2022, pp. 125–130.

[26] R. Kemker, M. McClure, A. Abitino, T. Hayes, C. Kanan, Measuring catastrophic forgetting in neural networks, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 32, 2018.

[27] F. Wiewel, B. Yang, Continual learning for anomaly detection with variational autoencoder, in: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019, pp. 3837–3841.

[28] K. Faber, R. Corizzo, B. Sniezynski, N. Japkowicz, Vlad: Task-agnostic vae-based lifelong anomaly detection, Neural Networks 165 (2023) 248–273. doi:https://doi.org/10.1016/j.neunet.2023.05.032.

[29] K. Faber, R. Corizzo, B. Sniezynski, N. Japkowicz, Active lifelong anomaly detection with experience replay, in: 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2022, pp. 1–10.

[30] J. S. Vitter, Random sampling with a reservoir, ACM Trans. Math. Softw. 11 (1) (1985) 37–57.

[31] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, J. C. Platt, Support vector method for novelty detection, in: Advances in neural information processing systems, 2000, pp. 582–588.

[32] Z. Li, Y. Zhao, N. Botta, C. Ionescu, X. Hu, Copod: copula-based outlier detection, in: 2020 IEEE International Conference on Data Mining (ICDM), IEEE, 2020, pp. 1118–1123.

[33] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, 2000, pp. 93–104.

[34] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 Eighth IEEE International Conference on Data Mining, IEEE, 2008, pp. 413–422.

[35] J. An, S. Cho, Variational autoencoder based anomaly detection using reconstruction probability, Special Lecture on IE 2 (1) (2015) 1–18.

[36] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization. (2018).

[37] N. Moustafa, J. Slay, Unsw-nb15: a comprehensive data set for network intrusion detection systems, in: 2015 Military Communications and Information Systems Conference (MilCIS), 2015, pp. 1–6.

[38] M. Tavallaee, E. Bagheri, W. Lu, A. A. Ghorbani, A detailed analysis of the kdd cup 99 data set, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6.

[39] K. Faber, R. Corizzo, B. Sniezynski, N. Japkowicz, Lifelong learning for anomaly detection: New challenges, perspectives, and insights, arXiv preprint arXiv:2303.07557 (2023).

Chapter 17

Autoencoder-based IDS for cloud and mobile devices

In this work, we proposed an autoencoder-based intrusion detection system that has the ability to work in cloud and mobile environments. The framework can monitor networks such as virtual cloud networks as well as data from mobile environments. Intrusion detection nodes process the gathered traffic data, trying to identify any anomalies that could be symptoms of intrusions. Moreover, we created time-window-based features that are collaboratively shared between multiple intrusion detection nodes.

Our experimental evaluation showcased that the proposed approach achieves high detection results on a popular modern dataset. We also proved that the collaboratively shared data improved the detection capabilities of the models.

Contributions of Kamil Faber: In this work, Kamil Faber co-designed and developed the framework. Kamil Faber carried out the experiments and was co-author of all sections of the paper.

This paper was published at the 2021 IEEE/ACM International Symposium on Cluster, Cloud, and Internet Computing (CCGRid, A-Rank, 140 points).

2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)

2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid) | 978-1-7281-9586-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CCGrid51090.2021.00088

Autoencoder-based IDS for cloud and mobile devices

Kamil Faber*, Lukasz Faber† and Bartlomiej Sniezynski‡
Institute of Computer Science
AGH University of Science and Technology
Cracow, Poland
Email: *kfaber@agh.edu.pl, †faber@agh.edu.pl, ‡bartlomiej.sniezynski@agh.edu.pl

***Abstract*—Along with the popularization of cloud computing and the increase in responsibilities of mobile devices, there is a need for intrusion detection systems available for working in these two new areas. At the same time, the increase in computational power of mobile devices gives us the possibility to use them to do a part of data preprocessing. Similarly, more complex operations can be executed in the cloud – this concept is known as *mobile cloud computing*. In this paper, we propose an autoencoder-based intrusion detection system applicable to cloud and mobile environments. The system provides multiple data gathering points, allowing to monitor either fully controlled networks, like virtual networks in the cloud, or mobile devices scattered in different networks. The monitoring process uses both mobile devices and cloud computational power. Gathered network traffic records are sent to a proper intrusion detection node, which executes the detection process. In case of suspicious behavior, an alert of a possible intrusion can be sent to the device owner. The detection process is based on an autoencoder neural network, which brings significant advantages: an anomaly-based approach, training only on benign samples, and a good performance. To improve detection results, we created time-window-based features, and there is also a possibility to share computed statistics between intrusion detection nodes. In the experiments, we construct three models using pure network flows data and time-window-based features. The results show that the autoencoder-based approach can detect with a high performance attacks not known during the training process. We also prove that created derived features have a significant impact on detection results.**

***Index Terms*—intrusion detection system, autoencoder, machine learning, security, mobile cloud computing**

I. Introduction

As the years go by, the number of devices connected to the global network grows steadily. According to the IoT Analytics report by [1], there are almost 22 billion active connected devices worldwide. Simultaneously, the number of hacker attacks increases – according to [2], in 2020 Q2, the number of unique incidents grew by 59% compared to the previous year. A significant part of providing security is analyzing network traffic to detect abnormal behaviors, which can signal that the system is under attack. This task is usually carried by Intrusion Detection System (IDS), which is responsible for detecting an intrusion and then alerting an administrator. With the increasing use of the cloud architecture and mobile devices, there are new challenges that need to be faced, including the possibility to monitor mobile devices and hosts located in different networks. More issues and challenges were indicated in a study of intrusion detection techniques in mobile cloud computing environments in [3].

In this work, we propose a novel idea of an autoencoder-based intrusion detection system, which have the following features:

- It works on network flows data, without analyzing packet's payload, which allows to work with encrypted traffic and protects users privacy.
- It allows us to monitor mobile devices and detect anomalies in their behavior, therefore to notify their user about possible intrusion.
- It allows us to monitor networks in which we have access to all packets passing through (using packet sniffing) – like virtual networks in cloud environments.
- It can use mobile devices computing power to do some preprocessing, but most of the computation is done outside those devices.
- It shares data between different networks.
- It uses a neural network – autoencoder – to execute intrusion detection with an anomaly-based approach.

In the next section, we shortly overview related works, including machine learning methods used in intrusion detection, cloud-based IDS, and an autoencoder architecture. Then, we describe the proposed intrusion detection system indicating its most essential features. After that, we run experiments to show that the proposed autoencoder-based approach is suitable for intrusion detection on network flows. There are significant improvements if we use derived time-window-based features. At the end, we summarize our conclusions and indicate possible future works.

II. Related works

A. Intrusion Detection Systems

There are two main classifications for intrusion detection systems, discussed by [4]. One is based on a detection method, and the second one on a place where the detection process is applied. Moreover, it is possible to differentiate two main categories of detection methods. The first one is signature-based intrusion detection, which bases on previously accumulated knowledge about attacks and vulnerabilities. Advantages and disadvantages of such an approach were pointed out in many papers like [4] or [5]. The most crucial advantage is a low

978-1-7281-9586-5/21/$31.00 ©2021 IEEE
DOI 10.1109/CCGrid51090.2021.00088

false positive ratio and a high detection level of known attacks. Unfortunately, this approach can not handle the detection of previously unknown attacks. Another category is anomaly-based intrusion detection, which assumes that it is possible to detect an intrusion by observing deviations from the users' normal behavior or the system. It recognizes any abnormal activity as an intrusion. The main advantage of anomaly-based systems is that they can detect new, never-seen attacks, and the main disadvantage is usually a high false positive ratio. Other advantages and disadvantages were presented in [5], [6] and [7].

Taking into account a source of data, intrusion detection systems usually were divided into host-based (HIDS) and network-based intrusion detection systems (NIDS). In HIDS, data are gathered from hosts, while in NIDS, the detection can base on pure information about network flows or, with a technique called Deep Packet Inspection, on extracted payload of network packets. Recently, there were also multiple mentions (for example, in [8]) about a distributed intrusion detection system (DIDS), which contains a lot of smaller detection nodes that can communicate with each other or with a centralized server.

B. Machine learning methods for intrusion detection

Machine learning techniques were used in the intrusion detection problem for many years. However, recently, the deep neural network architecture is becoming more and more popular and this trend also impacts intrusion detection. Many of the new research papers focus on applying deep learning techniques in this domain.

In [9] the authors provided a complex overview of deep learning approaches applied in intrusion detection task. They also listed the most important data sets used during the last 20 years. The authors applied seven deep learning models to two modern data sets – CSE-CIC-IDS2018 and the Bot-IoT. Their work provides valuable insights and a summary of many machine learning approaches to the intrusion detection task.

In [6], the authors did an overview of flow-based anomaly detection techniques, including the older ones and the most modern ones. They indicated that the anomaly-based approach working on network flows data has great potential and may answer the needs of the intrusion detection systems. However, they also note that there are still many difficulties that need to be challenged, like minimizing the false positive ratio and improving detection results.

In [10] the authors applied deep neural networks with three different architectures, each with three layers and with different sizes of the hidden layer. They executed experiments on the original data distribution and more balanced data obtained using down-sampling, up-sampling, class balances, and spread sub-sampling. In the original distribution, they were able to achieve an accuracy 99.65%. In other cases, their results were slightly worse. In the up-sampling and spread sub-sample cases, the authors did not manage to achieve over 97%.

In [11] the authors attempted to detect port scan attacks with the Deep Neural Network (DNN) architecture. They used the same data set that we use – CICIDS2017. They filtered only attacks labeled as port scans, and applied the down-sampling method to get a subset with balanced classes. The results achieved for port scan were very high – both the precision and recall had value 99%. In the same paper, they applied SVM architecture to the same task; however, it had low results – the precision at level 80% and the recall – 70%.

The long short-term memory (LSTM) architecture (first proposed by [12]) is well-suited for tasks based on time and sequence series data. Since network traffic is an example of a time series, the LSTM was also applied in the intrusion detection problem. In [13] the authors implemented a classifier based on LSTM-RNN and compared it to other methods (like KNN, SVM, PNN). LSTM was also tested on CIDDS-001 data set by [14]. They tried a few different LSTM architectures and compared them to other methods (C4.5, Random Forest, Hoeffding Tree, Naive Bayes, SVM). The experiment results shown that LSTM achieves similar or better results like other methods, which indicates that considering a perspective more comprehensive than just a single network flow may improve results.

Over the last few years, there have been a few approaches to apply autoencoders – a particular type of neural network which we also use in our approach. In [15] the authors proposed a complex framework based on an ensemble of autoencoders with prior feature extraction, including window-based features. One of their main focus was creating a detection method capable of running online on devices with low resources. The authors tested the proposed solution in a real environment built from a few IoT devices and achieved promising results. In [16] the authors applied a deep autoencoder trained in a greedy layer-wise way and tested it on KDD-99 data set. A similar approach, but with additional random-forest feature selection, was applied in [17]. In [18] the authors proposed optimization of computational cost by using just one detector for most of the network flows and the second one only for the most challenging cases. There were also approaches to use specialized types of autoencoders – for example, in [19] a denoising autoencoder was applied and verified on NSL-KDD data set, achieving high results.

C. Cloud-based IDS

A need to create intrusion detection systems explicitly adapted for the cloud environment was mentioned and emphasized in [20] and in [8], in which the authors provided a study of challenges and solutions for cloud intrusion detection. There is also a comprehensive study of computational intelligence intrusion detection techniques in mobile cloud computing environments [3]. The authors compared data sets and methods applied since the beginning of the cloud. They also indicated many open research issues, including the location of the IDS.

In [21] the authors proposed an intrusion detection method using deep neural network architecture build with Improved Genetic. Their solution achieved good results in detecting

attacks basing on network flows from the data set used also in our experiments, CICIDS2017. In [22] the authors created a multi-layer neural network optimized with hybrid glow swarm optimization–tabu search for intrusion detection system in cloud environment. This optimization helped them improve detection performance. The innovative, hypervisor-level distributed network security framework was proposed in [23] to increase the cloud's security. The authors also applied an anomaly-based detection method and tested it on three data sets, including CICIDS2017. However, their work was more focused on protecting services based in the cloud than on mobile devices.

In [24] the authors proposed a hybrid intrusion detection system, which comprises two parts – the first one for detection on the virtual machines level and the second one for network threats detection, which is also our concern. Another innovative intrusion detection scheme, based on clustering algorithm and distance-based traffic filtration, was proposed in [25]. The authors of [3] provided in-depth analysis and overview of existing intrusion detection systems for mobile cloud computing. They also described open issues in this research area, including the IDS location, heterogeneity of environment, and lack of the up-to-date signatures.

D. Autoencoder

An autoencoder is a special type of neural network architecture. We may consider it as built of three parts:

1) Code: A code layer is a hidden layer placed between the encoder and the decoder. It represents a code h used to encode the input.
2) Encoder: The goal of the encoder is to transform the input into the code. It can be formally written as:
$$h = f(x) \tag{1}$$
where x the input and f is the encoding function provided by the encoder.
3) Decoder: The goal of the decoder is transforming the code into the output, which reflects the previously encoded input. It can be formally written as:
$$y = g(h) = g(f(x)) \tag{2}$$
where y is the output, h the code and g is the decoding function provided by the decoder.

As indicated by [26], in a training phase, we do not want the autoencoder to learn to simply copy an input, as it would not be advantageous. The goal is to restrict the model so that it must do an approximate copy. As a result, we force the model to choose which aspects of the input it should copy. Thanks to that, the autoencoder is often able to learn valuable properties of the data.

We may consider the autoencoder as a particular case of a feedforward network and train it with all the same techniques. The goal of the learning process is minimizing the given loss function, which is calculated by comparing the input and the output of the model.

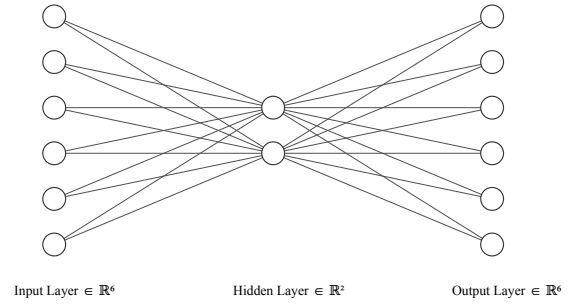


Fig. 1. Structure of the shallow undercomplete autoencoder.

According to [26], the autoencoders were initially usually used for features learning and dimensionality reduction, as they achieve better results than PCA. However, in the last years, the autoencoders have much wider application – they are used, or example, in generative modeling and anomaly detection. That last application is the main point of interest in that work.

Undercomplete autoencoder: One way to get useful properties in the hidden layer is to constrain the size of the code to have a smaller dimension than the input. This approach should force the autoencoder to capture the structure of the training data. The autoencoder with the code dimension smaller than the input size is named undercomplete. Its sample structure is presented in Figure 1.

III. Proposed solution

We propose a distributed IDS architecture that can monitor mobile devices and, at the same time, does most computation outside of those devices. We present the structure of the system in figure 2. We can differentiate three main elements of the proposed IDS:

- M – a set of Monitors;
- N – a set of Intrusion Detection Nodes;
- C – a set of connections between Monitors and Intrusion Detection Nodes.

Monitor is an element responsible for gathering data necessary for the intrusion detection process, preprocessing them, and sending them to Intrusion Detection Node (IDN). Every IDN receives data from a few Monitors and creates new derived features based on the time-window. In the next step, it shares computed statistics with other IDNs to improve detection performance. The core work of the IDN is to execute an anomaly-based detection process. We provide the details of every part of the system in the further paragraphs.

A. Gathering data in IDS

The data used by the proposed IDS are records of bidirectional network flows. There is no need to analyze the contents of the paylods of the packets. This approach results in two critical advantages: the ability to handle encrypted traffic and the protection of users' privacy.

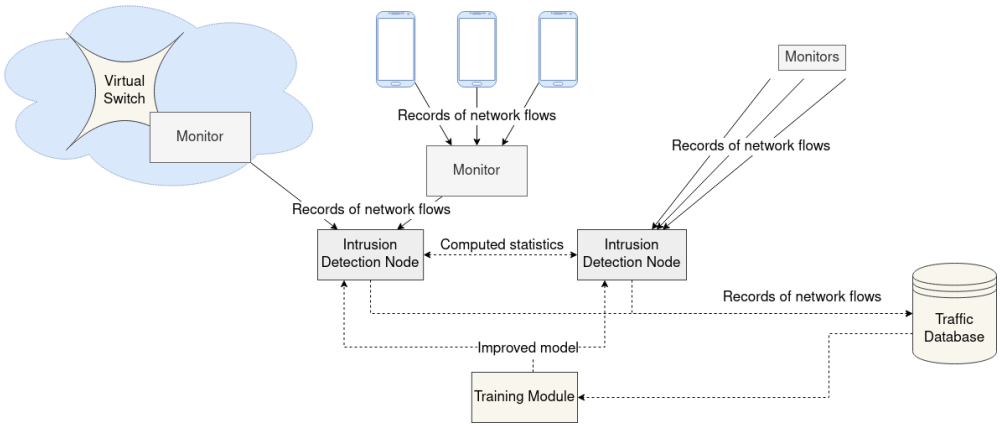


Fig. 2. A structure of the proposed IDS.

We differentiate two cases of data gathering and, therefore, two types of Monitors. As the first case, we consider a situation in which there is a possibility to monitor the whole traffic in the network. It concerns cases like controlled local networks, cellular base stations, or virtual switches in some cloud environments. As the second case, we consider monitoring mobile devices scattered in various networks, in which we do not have access to the provider's infrastructure and can not monitor them as a whole. In such a situation, we use those mobile devices as monitoring points. All of them gather data about their network traffic, preprocess them, and then send it to the monitor, which forwards it to the intrusion detection node.

We can consider a case in which mobile devices are infected and malware blocks or alters data before sending it to the monitor. In such a situation, there are two possible scenarios. If no data are sent, the intrusion detection node notices it and notify the owner about the issue. If data received from a mobile device are altered to hide malware actions and pretend to behave appropriately, the intrusion detection process may not detect infection. However, it does not affect the ability of the system to detect other intrusions.

B. Transforming data into features

There are two steps of processing gathered data to features. The first one is done at Monitors or the monitored mobile devices. Raw network traffic records are transformed to basic statistics describing every bidirectional network. Examples of such features are the duration time of the network flow and the number of bytes sent in the flow. This process can be done using tools like CICFlowMeter provided by [27].

The second step is executed at Intrusion Detection Node, and it includes computing statistics based on a broader perspective than the one available for a single monitor. The purpose of those derived features is to improve detection performance. They are computed based on data gathered from all Monitors assigned to the given Intrusion Detection Node and grouped into so-called time-window. Time-window data are network flows from some period. They can be limited either by time measurement (for example, 10 minutes) or a number of last connections that should be stored (for example, last 100 connections). We create the following features based on a time-window:

1) Number of connections from the source IP address;
2) Number of different destination ports referenced by the source IP address;
3) Number of different pairs (destination IP, destination port) referenced by the source IP address;
4) Number of connections to the same destination IP address;
5) Number of connections to the same destination IP address and destination port number;
6) Total flows duration between source and destination IP addresses.

C. Data shared between Intrusion Detection Nodes

To improve computing derived features, we also consider a possibility to share data between Intrusion Detection Nodes. IDN can periodically send minimal statistics required to compute features 1-3 from the previous paragraph. The decision which nodes share data must find a balance between pros and cons. Sharing data between too many nodes may increase data transfer and slow down the detection process. On the other hand, narrowing data exchange too much may lower the detection ratio.

D. Detection process

Detection process is executed by autoencoders – the neural network architecture shortly introduced in section II-D. There are several reasons why we decided to use autoencoders:

- Anomaly detection based approach;
- Good performance and detection results;
- A training phase requires only a benign class;
- Possibility to improve a model with the use of incremental learning.

Below, we shortly explain how we apply the autoencoder in the intrusion detection problem.

1) Training phase: The first step is training the model so that it learns the characteristics of the benign network traffic. A training process uses only normal data. The goal of the model is to minimize the value of reconstruction error. We use Mean Squared Error as a loss function – mean of the squares of the differences between the output and the input. During the training phase, the autoencoder learns how to reconstruct samples from typical characteristics. It means that it should provide a low reconstruction error for a benign sample.

2) Detection phase: In a detection process, the model is used to reconstruct an input sample. After that, we calculate a reconstruction error comparing predicted values to the original input. If the reconstruction error is high, it indicates that the sample is from different characteristics than the training data and therefore can be classified as an anomaly. It can be formally written as:

$$Label = \begin{cases} Normal, & \text{if } RE \leq Threshold \\ Anomaly, & \text{if } RE > Threshold \end{cases} \tag{3}$$

where

$$RE = p(g(f(x)) - x) \tag{4}$$

where *RE* is a Reconstruction Error and p is a metric function (in our case the mean squared error) and the threshold for a cut-off point is chosen arbitrarily. The choice of the threshold should be made concerning the potential cost of false positives and false negatives. All anomalies are classified as malicious traffic. In our experiments we do not choose a single value of the threshold, we show true positives ratio results for a few false positives values.

E. Notification for mobile device owner

If an intrusion detection node detects some suspicious behavior originating from an end-type device monitored by Monitors, notification about it may be sent to this device. Therefore, it is possible to inform an owner of the device (for example, a mobile phone) about suspicious behavior and a possible threat (malware, access without permission). It can be achieved, for example, by the use of push notifications from a dedicated mobile app or email messages to a user protected by the system. This approach can help in the detection of compromised mobile devices, while at the same time, it does not require running the whole detection process on mobile devices. Thanks to that, it is possible to optimize how much computation is done on the mobile phone while moving the rest to the Monitors or Intrusion Detection Nodes.

F. Model retraining

In real-life situations, the characteristic of network traffic is not constant – it evolves due to changes in infrastructure or users' behavior. Such changes are known as concept drift. To keep the models up to date, the system should be extended with the possibility to improve the model with time. It requires periodic gathering records of network traffic in the Traffic Database and using them to improve the model in the Training Module. The model can be fully retrained or improved with the use of incremental learning. After that, the new model can be uploaded to the intrusion detection nodes. That approach requires administrators activity, as they must decide when the model should be retrained and filter training samples so that they contain only benign traffic.

G. Potential of scalability

The distributed nature of the proposed intrusion detection system also makes it easier to scale. The first issue to consider is creating smaller subsets of intrusion detection nodes that share data between them but not between all the detection nodes. It may have a slight negative impact on detection results, but at the same time, it minimizes the volume of shared data.

We also should consider whether the central traffic database will become a bottleneck. As the database's primary goal is to provide data for the training module, we do not have to send and store all records of network flows there. As we mentioned in section III-F, this process requires only periodic gathering records of network traffic. We can also have a separate traffic database and training module for every separated subset of intrusion detection nodes. This approach can lead to the creation of models specialized for a particular part of the network. However, there is still a need to examine in which situations it will improve detection results and in which it can have a negative impact.

IV. Experiments

Experiments concentrate on testing the proposed approach to detect intrusions at Intrusion Detection Node. We set the following goals for them:

- We would like to show that the proposed autoencoder-based architecture is suitable for the intrusion detection task using network flows as input data.
- We would like to show improvement in detection performance due to derived features based on a time-window.

In the experiments, we compare the effectiveness of three types of models:

1) Undercomplete autoencoder for single flow features;
2) Undercomplete autoencoder for single flow features and prepared by us window-based features, where the window contains last 100 samples;
3) Undercomplete autoencoder for single flow features and window-based features, where the window contains samples from the last 10 minutes.

The hyperparameters in the used models were chosen manually based on many executions of experiments and are presented in tables I and II.

To evaluate proposed models, we use CICIDS2017 [28] – a modern data set generated by the Canadian Institute for Cybersecurity. In [29], a comprehensive survey of available data sets for network-based intrusion detection, the authors reviewed 35 different data sets, analyzing them from the

TABLE I
ARCHITECTURE OF THE UNDERCOMPLETE AUTOENCODER (MODEL 1).

L. no.	Type	Size	Activation function
1	Input	79	–
2	Dense	36	relu
3	Output	79	sigmoid

TABLE II
ARCHITECTURE OF THE UNDERCOMPLETE AUTOENCODER WITH TIME-WINDOW-BASED FEATURES (MODELS 2 AND 3).

L. no.	Type	Size	Activation function
1	Input	85	–
2	Dense	36	relu
3	Output	85	sigmoid

perspective of 15 various properties. They have shown that CICIDS2017 is a significant improvement compared to the previous ones. It contains 3.1M network flows gathered in an emulated environment over five days. Normal user behavior was generated by Benign Profile Agent simulating 25 users, while attacks were executed with the use of common tools. The data set contains the following attacks: DoS GoldenEye, Heartbleed, DoS Hulk, DoS Slowhttp, DoS slowloris, SSH-Patator, FTP-Patator, Web Attack, Infiltration, Bot, PortScan, and DDoS. Gathered network flows were transformed into 80 features by CICFlowMeter [27]. In our experiments, we use all of them except for *timestamp*. Details of the creation process and data set description are provided in [28]. The authors followed the criteria of a benchmark data set defined in [30]. The choice of executed attacks bases on a list of currently common attack families, and some have built-in obfuscation. It is suitable for our experiments as it contains records of network flows and can simulate network traffic present in the cloud. Before experiments, we cleaned data removing records with missing values. We also applied MinMaxScaler from scikit-learn python library. Detailed analysis of the data set was provided in [31].

V. ANALYSIS OF THE RESULTS

To evaluate the results of experiments, we decided to use the following metrics:

- Receiver Operating Characteristic (ROC) curve plot shows a true positive rate (TPR) against a false positive rate (FPR) [32]. TPR is also known as a detection rate and FPR as a false alarm rate.
- ROC-AUC is the Area Under ROC Curve. We decided to use ROC-AUC because those two metrics are very intuitive for the intrusion detection problem, as they provide dependence between TPR and FPR. It is beneficial, as detection rate and false alarm rates are very natural and easy to interpret in the considered context.
- A table with typical false positive rates (0.001, 0.01, 0.05) and corresponding true positive rates. It allows to read precise values easier than from a ROC plot and provides an easy way to compare the model's ability to achieve a high detection rate while keeping a low false detection rate.

TABLE III
ROC-AUC VALUES FOR ALL USED MODELS.

	Model	ROC-AUC
1	Single flow features	0.884
2	100 samples window	0.926
3	10 minutes window	0.988

TABLE IV
FPR AND TPR PAIRS FOR THE EXPERIMENTS.

	FPR	0	0.001	0.01	0.05
TPR	**Model 1**	0	0.36	0.50	0.60
	Model 2	0.0	0.28	0.42	0.49
	Model 3	0.34	0.79	0.89	0.94

We provide the ROC-AUC values for every model in the table III, while FPR and TPR pairs are in the table IV. We also present the ROC-AUC values for model 3 for every attack in the table V and confusion matrix for model 3 in the table VI. Below we provide a thorough analysis of those results, and we also show detailed results for different attacks.

Impact of working with more than a single network flow

Looking at the results in the table IV we can see that adding time-window-based features significantly improves results achieved by models. The second and the third models are significantly better in detecting malicious traffic than the

TABLE V
ROC-AUC VALUES FOR EVERY ATTACK IN MODEL 3

Attack type	ROC-AUC	Attack type	ROC-AUC
Bot	0.850	**Heartbleed**	0.999
DDoS	0.998	**Infiltration**	0.980
DoS GoldenEye	0.991	**PortScan**	0.990
DoS Hulk	0.999	**SSH Patator**	0.817
DoS Slowhttptest	0.980	**Web Brute Force**	0.635
DoS slowloris	0.907	**SQL Injection**	0.898
FTP-Patator	0.792	**Web XSS**	0.600

TABLE VI
CONFUSION MATRIX FOR FPR=0.01 (MODEL 3)

		Actual	
		P	N
Predict	P	499211	4543
	N	57345	449720

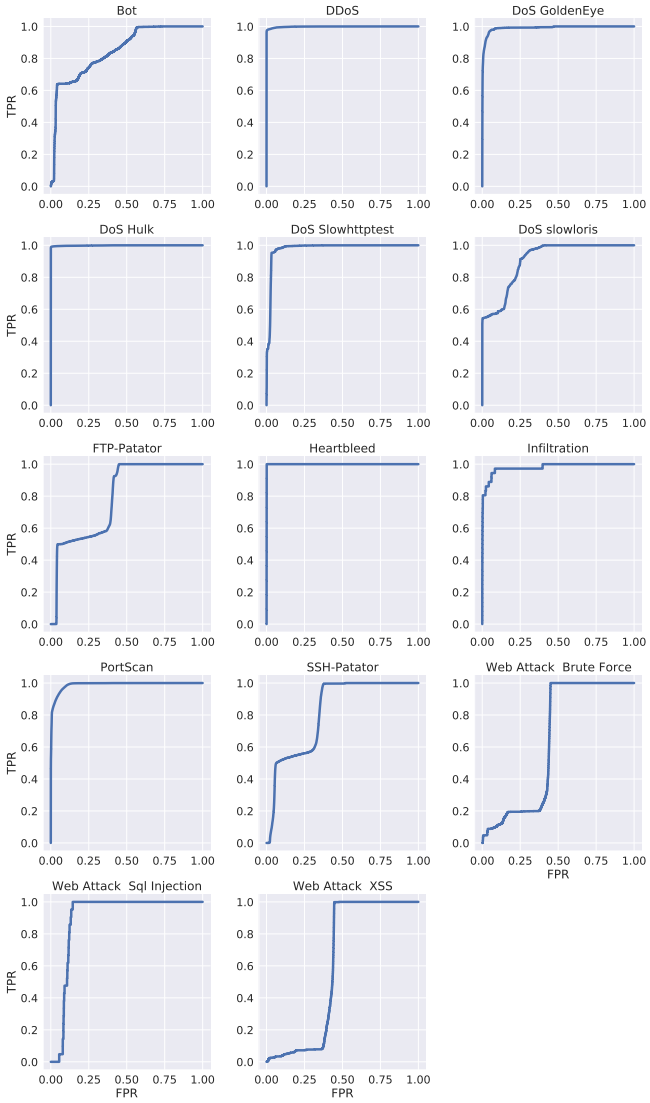


Fig. 3. ROC curves for every attack type in the experiment 3.

undercomplete model with only single network flow features as input.

In Figure 3 we present our analysis of separate ROC curves for every label in experiment 3 (the model trained with features computed for 10 minutes window).

We can notice that the following attacks are very well detected for the enhanced features set: DDoS, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, Heartbleed, Infiltration. Most of these attacks are based on sending many requests continuously, which may be why time-window features improve their detection. We can also notice that the mobile devices can be used in some of those attacks, like DDoS. Therefore, applying that approach may improve the detection of compromised devices and allow to notify the owners about possible threats.

By comparing results for model 1 (single flows) and model 3 (10 minutes window), we observed the following results of adding time-window-based features:

1) The ability to detect DDoS attacks, DoS Hulk, PortScan was significantly improved.
2) The ability to detect SSH-Patator, FTP-Patator, and Bot attacks was slightly improved.
3) There was no significant improvement in the detection of Web Attacks.

The first conclusion seems to be pretty obvious – all these attacks are based on sending many requests. The fascinating thing is that the detection of FTP-Patator is much better, while there is just a slight improvement in SSH-Patator detection. One possible explanation is that SSH protocol is encrypted

– however, solving this issue would require further studies. Lack of improvement in the detection of Web Attacks can be explained by the fact that those attacks depend more on a packet payload than some time-window-based features (see below).

Possibility of detecting all types of attacks

As we have already described, the autoencoders can detect some attacks using just features containing information about single data flows without a deep packet inspection. However, even with the usage of time-window features, some attacks are still undetectable, which means the proposed model cannot differentiate them from benign characteristics. It raises the question about the possibility of detecting such attacks without analyzing packets payload. The undetectable attacks are Web Attack Brute Force and Web Attack XSS.

It is not surprising that the model cannot detect web attacks, as they are usually distinguished by packet payload or analysis of logs from the victim device. From the perspective of network flows, web attacks may look like benign traffic. For example, XSS may have the same packet's characteristics. The only difference may be in the packet's payload, and its analysis requires deep packet inspection. It is worth noticing that web attacks usually are neither carried out by mobile devices nor targeted at mobile devices. Therefore, they are not the main problem from the perspective of protecting mobile devices. There are also other methods allowing applicable on application-level and capable of dealing with such attacks.

Comparison to the results of other studies

It is hard to compare our results to the ones achieved by the authors of the data set [30]. We apply a different approach (anomaly detection instead of standard classifiers). We also do not know how many samples of each class their test data set contained. Our test data set contains 45% of benign traffic and 55% of malicious traffic. The best results achieved by [30] were for algorithm ID3 (Iterative Dichotomiser 3): $\mathit{Precision} = 0.98$ and $\mathit{Recall} = 0.98$. The results achieved by us for the third model (for 0.05 FPR value) are: $\mathit{Precision} = 0.94$ and $\mathit{Recall} = 0.95$, which means that our results are similar (but slightly worse) comparing to theirs. However, we should note that the authors of [30] trained an instance of a model for every single attack type. They also applied feature selection on the full data set and chose 3 or 4 best features for every attack type. We use one model and train it with only benign traffic to check the model's ability to detect previously unseen attacks. Therefore, an advantage of our approach is the lack of need to know all attacks a priori.

In the following points, we shortly summarize the results of the experiments:

- The neural network autoencoder architecture can achieve high results in the intrusion detection problem and, therefore, suitable for use in intrusion detection nodes.
- New time-window-based derived features improve detection results.
- Not all types of attacks can be detected with this approach, probably because they are invisible from the network flows perspective (like web attacks).

VI. Conclusions

In this paper, we have proposed the autoencoder-based intrusion detection system specifically designed to work in cloud and mobile environments. It can monitor both: networks, in which we have access to all packets passing through, like in some virtual cloud environments, and mobile devices scattered into various networks around the world. It is also possible to notify the owners of the mobile devices that their devices may be compromised. Our experiments show that the autoencoder architecture, in connection with network flows data and derived features, can achieve high detection results. Therefore, it is suitable for use in the intrusion detection system operating in the cloud. However, it is worth noticing that not all types of attacks can be detected with this approach due to the use of network flows data only.

In future works, we would like to focus on defining a more generic framework for intrusion detection in cloud environments and preparing tools to monitor mobile devices. Another attractive research area is automatizing the decision of whether preprocessing should be executed on a mobile device or in the cloud, taking into account the computing power and resources available on a given mobile device. It would also be worthwhile to gather data other than network flows from mobile devices and create new features to improve the detection of compromised devices.

Acknowledgment

The research presented in this paper was supported by the funds assigned to AGH University of Science and Technology by the Polish Ministry of Science and Higher Education.

References

[1] Knud Lasse Lueth, "State of the IoT 2020," tech. rep., 2020. Available online: https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/.

[2] P. Technologies, "Cybersecurity threatscape: Q2 2020," tech. rep., 2019. Available online: https://www.ptsecurity.com/ww-en/analytics/cybersecurity-threatscape-2020-q2/.

[3] S. Shamshirband, M. Fathi, A. Chronopoulos, A. Montieri, F. Palumbo, and A. Pescapè, "Computational intelligence intrusion detection techniques in mobile cloud computing environments: Review, taxonomy, and open research issues," *Journal of Information Security and Applications*, vol. 55, 2020.

[4] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," *Computer Networks*, vol. 31, pp. 805–822, apr 1999.

[5] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, p. 20, dec 2019.

[6] R. Sharma, A. Guleria, and R. Singla, "An overview of flow-based anomaly detection," *International Journal of Communication Networks and Distributed Systems*, vol. 21, no. 2, p. 220, 2018.

[7] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, vol. 51, pp. 3448–3470, aug 2007.

[8] Y. Mehmood, A. Shibli, U. Habiba, and R. Masood, "Intrusion detection system in cloud computing: Challenges and opportunities," pp. 59–66, 12 2013.

[9] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, feb 2020.

[10] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, "Deep and Machine Learning Approaches for Anomaly-Based Intrusion Detection of Imbalanced Network Traffic," *IEEE Sensors Letters*, vol. 3, pp. 1–4, jan 2019.

[11] D. Aksu and M. Ali Aydin, "Detecting Port Scan Attempts with Comparative Analysis of Deep Learning and Support Vector Machine Algorithms," in *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), ANKARA, Turkey*, pp. 77–80, IEEE, dec 2018.

[12] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, nov 1997.

[13] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection," in *2016 International Conference on Platform Technology and Service (PlatCon), Jeju, South Korea*, pp. 1–5, IEEE, feb 2016.

[14] L. Nicholas, S. Y. Ooi, Y. H. Pang, S. O. Hwang, and S.-Y. Tan, "Study of long short-term memory in flow-based network intrusion detection system," *Journal of Intelligent & Fuzzy Systems*, vol. 35, pp. 5947–5957, dec 2018.

[15] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," feb 2018.

[16] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pp. 178–183, 2018.

[17] X. Li, W. Chen, Q. Zhang, and L. Wu, "Building auto-encoder intrusion detection system based on random forest feature selection," *Computers & Security*, vol. 95, p. 101851, 2020.

[18] M. Gharib, B. Mohammadi, S. H. Dastgerdi, and M. Sabokrou, "Autoids: Auto-encoder based method for intrusion detection system," *ArXiv*, vol. abs/1911.03306, 2019.

[19] R. C. Aygun and A. G. Yavuz, "Network Anomaly Detection with Stochastically Improved Autoencoder Based Models," in *Proceedings - 4th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud, SSC 2017*, pp. 193–198, Institute of Electrical and Electronics Engineers Inc., jul 2017.

[20] A. Patel, M. Taghavi, K. Bakhtiyari, and J. Celestino Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 25 – 41, 2013.

[21] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri, and M. Rida, "Smart approach to build a deep neural network based ids for cloud environment using an optimized genetic algorithm," in *Proceedings of the 2nd International Conference on Networking, Information Systems & Security*, NISS19, (New York, NY, USA), Association for Computing Machinery, 2019.

[22] M. Manickam and S. Rajagopalan, "A hybrid multi-layer intrusion detection system in cloud," *Research Journal of Biotechnology*, vol. 2017, pp. 167–174, 03 2019.

[23] R. Patil, H. Dudeja, and C. Modi, "Designing an efficient security framework for detecting intrusions in virtual network of cloud computing," *Computers and Security*, vol. 85, pp. 402–422, 2019.

[24] H. Toumi, A. Eddaoui, and M. Talea, "Cooperative intrusion detection system framework using mobile agents for cloud computing," *Journal of Theoretical and Applied Information Technology*, vol. 70, no. 1, pp. 76–84, 2014.

[25] S. Dey, Q. Ye, and S. Sampalli, "A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks," *Information Fusion*, vol. 49, pp. 205–215, 2019.

[26] A. Goodfellow, Ian; Bengio, Yoshua; Courville, *Deep learning, MIT Press*, vol. 26. 2016.

[27] Canadian Institute for Cybersecurity, "CICFlowmeter - Network Traffic Flow analyzer," 2021. Available online: https://github.com/ahlashkari/CICFlowMeter.

[28] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," pp. 108–116, aug 2019.

[29] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, sep 2019.

[30] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a Reliable Intrusion Detection Benchmark Dataset," *Software Networking*, vol. 2017, pp. 177–200, jan 2017.

[31] R. Panigrahi and S. Borah, "A detailed analysis of cicids2017 dataset for designing intrusion detection systems," vol. 7, pp. 479–482, 01 2018.

[32] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, jun 2006.

List of Figures

2.1 An example scenario built on top of MNIST dataset. 10

3.1 Comparison between conventional anomaly detection with model updates and lifelong anomaly detection 16
3.2 Comparison of training/update and inference for non-lifelong and lifelong anomaly detection in the scenario with four tasks 17
3.3 Lifelong scenarios variants 22
3.4 Summary of experimental results for all datasets (Energy, NSL-KDD, UNSW, Wind) in three scenario types (CC, CR, R) 23

4.1 The Proposed M2I and I2M lifelong learning benchmarks. 26

5.1 Difference between Wasserstein distance and Kullbach-Leibler (KL) divergence. 30
5.2 Lifelong consolidation: Number of clusters extracted (average and standard deviation) for all methods and datasets considered. 35

6.1 VLAD: Learning stage. 38
6.2 Comparison of conventional change point detection and lifelong anomaly detection in terms of memory storage. 39
6.3 VLAD: The Consolidation component. 39
6.4 VLAD: The Memory Summarization component. 40
6.5 VLAD: The Experience Replay component. 41
6.6 Rank plot for all datasets. 42
6.7 Comparison of ROC-AUC values obtained for VAE and VLAD on each concept after learning different concepts. 44

7.1 Proposed framework for active lifelong anomaly detection. 46
7.2 ACR active lifelong anomaly detection strategy. 47
7.3 SAS active lifelong anomaly detection strategy. 47
7.4 Experimental study of the impact of contamination and replay buffer cleaning capabilities on the anomaly detection performance. 48

8.1 A structure of the proposed Intrusion Detection System. 52
8.2 Proposed architecture for collaborative continual intrusion detection consisting of four types of services: Data Selection, Intrusion Detection, Collaborative Replay, and Model Update. . 53
8.3 Example of our adaptive double-balanced replay strategy. 55

List of Tables

4.1 Experimental results (EfficientNet – M2I) in terms of average performance (and rank) for all lifelong learning strategies. 27

5.1 Change point detection: results in terms of Cover and F1 metrics averaged across multiple parameter configurations. 34

6.1 Anomaly detection results in terms of Lifelong ROC-AUC, Backward Transfer (BWT), and Forward Transfer (FWT). 43

7.1 Experimental results for the UNSW dataset. 49

8.1 Results in terms of Lifelong ROC–AUC with the Cumulative strategy. 57
8.2 Results in terms of Lifelong ROC–AUC with the Adaptive Double-balanced Replay. 57

Bibliography

[1] David Abel, Yuu Jinnai, Sophie Yue Guo, George Konidaris, and Michael Littman. Policy and value transfer in lifelong reinforcement learning. In *International Conference on Machine Learning*, pages 20–29, 2018.

[2] Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[3] Charu C Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer, 2017.

[4] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, and Farhan Ahmad. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.

[5] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, pages 622–637. Springer, 2019.

[6] Antonio L. Alfeo, Mario G.C.A. Cimino, Giuseppe Manco, Ettore Ritacco, and Gigliola Vaglini. Using an autoencoder in the design of an anomaly detector for smart manufacturing. *Pattern Recognition Letters*, 136:272–278, 2020.

[7] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[8] Suresh Kumar Amalapuram, Akash Tadwai, Reethu Vinta, Sumohana S. Channappayya, and Bheemarjuna Reddy Tamma. Continual learning for anomaly based network intrusion detection. In *2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 497–505, 2022.

[9] Philip H. S. Torr Ameya Prabhu and Puneet K. Dokania. Gdumb: A simple approach that questions our progress in continual learning. *Lecture Notes in Computer Science (LNIP)*, 12347, 2020.

[10] D. Anguita, Alessandro Ghio, L. Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *ESANN*, 2013.

[11] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. *ACM Sigmod record*, 28(2):49–60, 1999.

[12] Maroua Bahri, Albert Bifet, João Gama, Heitor Murilo Gomes, and Silviu Maniu. Data stream analysis: Foundations, major tasks and tools. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(3):e1405, 2021.

[13] Megan M Baker, Alexander New, Mario Aguilar-Simon, Ziad Al-Halah, Sébastien MR Arnold, Ese Ben-Iwhiwhu, Andrew P Brna, Ethan Brooks, Ryan C Brown, Zachary Daniels, et al. A domain-agnostic approach for characterization of lifelong learning systems. *Neural Networks*, 160:274–296, 2023.

[14] Gary G Berntson, Sarah T Boysen, and John T Cacioppo. Neurobehavioral organization and the cardinal principle of evaluative bivalence. *Annals of the New York Academy of Sciences*, 702:75–102, 1993.

[15] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys (CSUR)*, 54(3):1–33, 2021.

[16] Andrew P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[17] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[18] Seok-Jun Bu and Sung-Bae Cho. Deep character-level anomaly detection based on a convolutional autoencoder for zero-day phishing url detection. *Electronics*, 10(12):1492, 2021.

[19] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and Simone Calderara. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2180–2187. IEEE, 2021.

[20] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[21] Feng Cao, Martin Ester, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *SDM*, 2006.

[22] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[23] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2019.

[24] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[25] Ameni Chetouane and Kamel Karoui. Ddos detection approach based on continual learning in the sdn environment. In Ajith Abraham, Tzung-Pei Hong, Ketan Kotecha, Kun Ma, Pooja Manghirmalani Mishra, and Niketa Gandhi, editors, *Hybrid Intelligent Systems*, pages 1199–1208, Cham, 2023. Springer Nature Switzerland.

[26] Roberto Corizzo, Michael Baron, and Nathalie Japkowicz. Cpdga: Change point driven growing auto-encoder for lifelong anomaly detection. *Knowledge-Based Systems*, page 108756, 2022.

[27] Roberto Corizzo, Michelangelo Ceci, and Nathalie Japkowicz. Anomaly detection and repair for accurate predictions in geo-distributed big data. *Big Data Research*, 16:18–35, 2019.

[28] Roberto Corizzo, Michelangelo Ceci, Gianvito Pio, Paolo Mignone, and Nathalie Japkowicz. Spatially-aware autoencoders for detecting contextual anomalies in geo-distributed data. In *International Conference on Discovery Science*, pages 461–471. Springer, 2021.

[29] Roberto Corizzo, Michelangelo Ceci, Eftim Zdravevski, and Nathalie Japkowicz. Scalable auto-encoders for gravitational waves detection from time series data. *Expert Systems with Applications*, 151:113378, 2020.

[30] Andrea Cossu, Antonio Carta, Vincenzo Lomonaco, and Davide Bacciu. Continual learning for recurrent neural networks: an empirical evaluation. *Neural Networks*, 143:607–627, 2021.

[31] Andrea Cossu, Gabriele Graffieti, Lorenzo Pellegrini, Davide Maltoni, Davide Bacciu, Antonio Carta, and Vincenzo Lomonaco. Is class-incremental enough for continual learning? *Frontiers in Artificial Intelligence*, 5, 2022.

[32] Gideon Creech and Jiankun Hu. Generation of a new ids test dataset: Time to retire the kdd collection. In *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 4487–4492, 2013.

[33] Marc'Aurelio Ranzato David Lopez-Paz. Gradient episodic memory for continual learning. *arXiv*, https://arxiv.org/abs/1706.08840, 2017.

[34] Dias, Daniel, Peres, Sarajane, Bscaro, and Helton. Libras Movement. UCI Machine Learning Repository, 2009.

[35] Tom Diethe, Tom Borchert, Eno Thereska, Borja Balle, and Neil Lawrence. Continual learning in practice. *NeurIPS Continual Learning Workshop*, 2018.

[36] Priyanka Dixit and Sanjay Silakari. Deep learning algorithms for cybersecurity applications: A technological and status review. *Computer Science Review*, 39:100317, 2021.

[37] Keval Doshi and Yasin Yilmaz. Continual learning for anomaly detection in surveillance videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 254–255, 2020.

[38] Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for continual learning. *arXiv preprint 1810.13166*, 2018.

[39] Yachuang Feng, Yuan Yuan, and Xiaoqiang Lu. Learning deep event models for crowd anomaly detection. *Neurocomputing*, 219:548–556, 2017.

[40] Giuseppe Fenza, Mariacristina Gallo, and Vincenzo Loia. Drift-aware methodology for anomaly detection in smart grid. *IEEE Access*, 7:9645–9657, 2019.

[41] Ricardo Ferreira, Andrea Martiniano, and Renato Sassi. Behavior of the urban traffic of the city of sao paulo in brazil. UCI Machine Learning Repository, 2018.

[42] Gianluigi Folino and Pietro Sabatino. Ensemble based collaborative and distributed intrusion detection systems: A survey. *Journal of Network and Computer Applications*, 66:1–16, 2016.

[43] Ahmed Frikha, Denis Krompaß, and Volker Tresp. Arcade: A rapid continual anomaly detector. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10449–10456. IEEE, 2021.

[44] Joao Gama. *Knowledge discovery from data streams*. CRC Press, 2010.

[45] Benoit Gaujac, Ilya Feige, and David Barber. Learning disentangled representations with the wasserstein autoencoder. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 69–84. Springer, 2021.

[46] Diptiben Ghelani, Tan Kian Hua, and Surendra Kumar Reddy Koduru. Cyber security threats, vulnerabilities, and security solutions models in banking. *Authorea Preprints*, 2022.

[47] Daniel Gibert, Carles Mateu, and Jordi Planes. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. *Journal of Network and Computer Applications*, 153:102526, 2020.

[48] Markus Goldstein and Seiichi Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, 11(4):e0152173, 2016.

[49] W. Haider, J. Hu, J. Slay, B.P. Turnbull, and Y. Xie. Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling. *Journal of Network and Computer Applications*, 87:185–192, 2017.

[50] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, 2017.

[51] D.M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[52] Waleed Hilal, S Andrew Gadsden, and John Yawney. Financial fraud: a review of anomaly detection techniques and recent advances. *Expert systems With applications*, 193:116429, 2022.

[53] Rui Hou, Yifan Peng, Lars J. Grimm, Yinhao Ren, Maciej A. Mazurowski, Jeffrey R. Marks, Lorraine M. King, Carlo C. Maley, E. Shelley Hwang, and Joseph Y. Lo. Anomaly detection of calcifications in mammography based on 11,000 negative cases. *IEEE Transactions on Biomedical Engineering*, 69(5):1639–1650, 2022.

[54] Ning Huyan, Dou Quan, Xiangrong Zhang, Xuefeng Liang, Jocelyn Chanussot, and Licheng Jiao. Unsupervised outlier detection using memory and contrastive learning. *IEEE Transactions on Image Processing*, 31:6440–6454, 2022.

[55] Christian Janiesch, Patrick Zschech, and Kai Heinrich. Machine learning and deep learning. *Electronic Markets*, 31(3):685–695, 2021.

[56] Haeyong Kang, Rusty J. L. Mina, Sultan Rizky, Hikmawan Madjid, Jaehong Yoon, Mark Hasegawa-Johnson, Sung Ju-Hwang, and Chang D. Yoo. Forget-free continual learning with winning subnetworks. *International Conference on Machine Learning (ICML)*, 1, 2022.

[57] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

[58] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, 2(1), 2019.

[59] Junbong Kim, Kwanghee Jeong, Hyomin Choi, and Kisung Seo. Gan-based anomaly detection in imbalance problems. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 128–145. Springer, 2020.

[60] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[61] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[62] Bartosz Krawczyk, Bernhard Pfahringer, and Michał Woźniak. Combining active learning with concept drift detection for data stream mining. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2239–2244. IEEE, 2018.

[63] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[64] Dhireesha Kudithipudi, Mario Aguilar-Simon, Jonathan Babb, Maxim Bazhenov, Douglas Blackiston, Josh Bongard, Andrew P Brna, Suraj Chakravarthi Raja, Nick Cheney, Jeff Clune, et al. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210, 2022.

[65] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[66] Rikard Laxhammar and Göran Falkman. Online learning and sequential anomaly detection in trajectories. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1158–1173, 2013.

[67] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. 2015.

[68] Hongzu Li and Pierre Boulanger. A survey of heart anomaly detection using ambulatory electrocardiogram (ecg). *Sensors*, 20(5):1461, 2020.

[69] Yi-Fan Li, Yang Gao, Gbadebo Ayoade, Hemeng Tao, Latifur Khan, and Bhavani Thuraisingham. Multistream classification for cyber threat data with heterogeneous feature space. In *The World Wide Web Conference*, WWW '19, page 2992–2998, New York, NY, USA, 2019. Association for Computing Machinery.

[70] Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: copula-based outlier detection. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 1118–1123. IEEE, 2020.

[71] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018.

[72] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944. IEEE, 2017.

[73] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[74] Vincenzo Lomonaco. Continual learning with deep architectures. *PhD Thesis*, 2019.

[75] Vincenzo Lomonaco, Davide Maltoni, and Lorenzo Pellegrini. Rehearsal-free continual learning over small non-i.i.d. batches. *1st Workshop on Continual Learning in Computer Vision at CVPR2020*, 2019.

[76] Christopher J MacLellan, Erik Harpstead, Vincent Aleven, Kenneth R Koedinger, et al. Trestle: a model of concept formation in structured domains. *Advances in Cognitive Systems*, 4:131–150, 2016.

[77] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

[78] Dhrumil Malani, Jimit Modi, Siddharth Lilani, Yukta Desai, and Ritesh Dhanare. Implementation of intrusion detection systems in distributed architecture. In *International Conference on Cyber Warfare, Security and Space Research*, pages 144–155. Springer, 2021.

[79] Benjamin Maschler, Thi Thu Huong Pham, and Michael Weyrich. Regularization-based continual learning for anomaly detection in discrete manufacturing. *Procedia CIRP*, 104:452–457, 2021.

[80] David S Matteson and Nicholas A James. A nonparametric approach for multiple change point analysis of multivariate data. *Journal of the American Statistical Association*, 109(505):334–345, 2014.

[81] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. Kitsune: An ensemble of autoencoders for online network intrusion detection. In *Network and Distributed Systems Security (NDSS) Symposium*, 2018.

[82] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, 2015.

[83] Martin Mundt, Yongwon Hong, Iuliia Pliushch, and Visvanathan Ramesh. A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. *Neural Networks*, 160:306–336, 2023.

[84] Yuval Netzer, Tao Wang, Adam Coates, A. Bissacco, Bo Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011.

[85] Xiangli Nie, Zhiguang Deng, Mingdong He, Mingyu Fan, and Zheng Tang. Online active continual learning for robotic lifelong object recognition. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2023.

[86] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[87] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54 – 71, 2019.

[88] Jitendra Parmar, Satyendra Chouhan, Vaskar Raychoudhury, and Santosh Rathore. Open-world machine learning: applications, challenges, and opportunities. *ACM Computing Surveys*, 55(10):1–37, 2023.

[89] KJ Prabuchandran, Nitin Singh, Pankaj Dayama, Ashutosh Agarwal, and Vinayaka Pandit. Change point detection for compositional multivariate data. *Applied Intelligence*, pages 1–26, 2021.

[90] Sowmya Ramapatruni, Sandeep Nair Narayanan, Sudip Mittal, Anupam Joshi, and Karuna Joshi. Anomaly detection models for smart home security. In *2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, pages 19–24. IEEE, 2019.

[91] Oliver Roesler. EEG Eye State. UCI Machine Learning Repository, 2013.

[92] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[93] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *NIPS 2016 Deep Learning Symposium*, 2016.

[94] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, 15(9):1779–1797, 2022.

[95] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588, 2000.

[96] Sumaiya Shaikh, Ch. Rupa, Gautam Srivastava, and Thippa Reddy Gadekallu. Botnet attack intrusion detection in iot enabled automated guided vehicles. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 6332–6336, 2022.

[97] Iman Sharafaldin, Arash Habibi Lashkari, Ali A Ghorbani, et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116, 2018.

[98] Shiven Sharma, Anil Somayaji, and Nathalie Japkowicz. Learning over subconcepts: Strategies for 1-class classification. *Computational Intelligence*, 34(2):440–467, 2018.

[99] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, volume 30. Curran Associates, Inc., 2017.

[100] Vinicius MA Souza, Farhan A Chowdhury, and Abdullah Mueen. Unsupervised drift detection on high-speed data streams. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 102–111. IEEE, 2020.

[101] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[102] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6, 2009.

[103] Sebastian Thrun and Tom M. Mitchell. Lifelong robot learning. *Robotics and Autonomous Systems*, 15(1):25–46, 1995. The Biology and Technology of Intelligent Autonomous Agents.

[104] Nikko Tinbergen. The hierarchical organization of nervous mechanisms underlying instinctive behaviour. In *Symposia of the Society for Experimental Biology*, volume 4, pages 305–312, 1950.

[105] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[106] G. J. J. Van den Burg and C. K. I. Williams. An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222*, 2020.

[107] Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57, mar 1985.

[108] Siqi Wang, Yijie Zeng, Xinwang Liu, En Zhu, Jianping Yin, Chuanfu Xu, and Marius Kloft. Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[109] Felix Wiewel and Bin Yang. Continual learning for anomaly detection with variational autoencoder. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3837–3841. IEEE, 2019.

[110] Shuang Wu, Jingyu Zhao, and GuangJian Tian. Understanding and mitigating data contamination in deep anomaly detection: A kernel-based approach. In *IJCAI*, 2022.

[111] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[112] LeCun Yann. The MNIST database of handwritten digits. Online: http://yann.lecun.com/exdb/mnist/, 1998.

[113] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017.

[114] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. In *1996 ACM SIGMOD International Conference on Management of Data*, page 103–114, 1996.

[115] Yue Zhao, Xiyang Hu, Cheng Cheng, Cong Wang, Changlin Wan, Wen Wang, Jianing Yang, Haoping Bai, Zheng Li, Cao Xiao, et al. Suod: Accelerating large-scale unsupervised heterogeneous outlier detection. *Proceedings of Machine Learning and Systems*, 3, 2021.