Computer Science • 15 (1) 2014 http://dx.doi.org/10.7494/csci.2014.15.1.19

Zbigniew Kaleta

SEMANTIC TEXT INDEXING

Abstract *The following article presents a specific issue of semantic analysis of texts in natural language – text indexing and describes one field of its application (web browsing). The main part of this article describes a computer system assigning a set of semantic indexes (similar to keywords) to a particular text. The indexing algorithm employs a semantic dictionary to find specific words in a text that represent a text content. Furthermore, it compares two given sets of semantic indexes to determine similarities between texts (assigning a numerical value). The article describes the semantic dictionary – a tool essential to accomplish this task and its usefulness, the main concepts of the algorithm, and the test results.*

Keywords text subject, semantic analysis, indexing

1. Introduction

The turn of the Twenty-First Century was marked by the appearance and tremendous growth of the Internet as well as the World Wide Web. It evolved from a simple tool for sending messages into a sophisticated environment co-created by around two billion people all over the world. Such enormous collection of data requires an efficient means of finding, discovering, and retrieving information. A large part of that functionality is provided by dedicated search engines.

Internet search engines have co-evolved with the growing web. They started as simple programs relying on keyword information provided by the authors of the webpages. Nowadays, they consist of enormous databases with complex metrics that are capable of finding text information as well as graphics, videos, etc. Traffic generated by these search engines makes them profitable; thus, their progress has accelerated. However, they are still mostly based on statistical algorithms. The often-superfluous and irrelevant results provided by those engines create the need for search engines that "understand" content and the user's query in order to provide with results more suitable to the user's needs. Such "understanding" requires semantic analysis.

The following paper describes semantic dictionaries necessary for semantic analysis. The main part of this article is dedicated to one of the possible approaches to the creation of a semantic indexing system (a semantic tagger) – it describes the algorithm, its tests, and the results.

2. The need for semantic tagging

Efficient browsing of large data requires the information to be both comprehensive (to achieve a high quality of results) and short (to be obtained quickly). Naturally, one is usually obtained at cost of the other, so a satisfactory trade-off needs to be found.

An usual approach to large text corpora is from a quantitative point of view. Such an approach focuses on the content based on word frequency models or relations between texts. The former approach is represented by such models as the Space Vector Model (SVM) or Latent Semantic Analysis (LSA) [10]. An example of the latter is the PageRank algorithm [17]. The greatest advantages of statistical approaches are their independence from a specific language and relatively low computational cost. Still, this approach is insufficient for obtaining satisfactory results [4, 6, 7]. It yields too many false positive results due to operating only on the level of forms without prior knowledge of their meaning (more on this in section 3). This is the cost of language independence (LSA partially addresses this issue).

One way to improve the quality of search results containing superfluous results is to use relevance feedback. An available solution using this technique is *inter alia* Surf Canyon [11]. Its main idea is to provide the user with a large set of results, which is then gradually reduced based on information coming from user interests. Such interest may be expressed explicitly, *e.g.*, by clicking a button, or implicitly, *e.g.*, by

stopping for a given amount of time. However, during the Text Retrieval Conference (TREC) this approach was found to be not fully satisfactory [4, 6]. One of its main disadvantages is the time-consuming gathering of data about relevance: the user needs to analyze one of the results and decide whether it is relevant. Moreover, this data is valid only for one search – the next query may not be related to the one previously analyzed.

> In 1992 (concurrently with MUC [Message Understanding Conference]), the TREC (Text REtrieval Conference) started. This is an annual workshop aimed at adopting classic Information Retrieval algorithms to search extremely large text collections and Web data. It is important to notice that classic IR algorithms used a search pattern, which was a list of words. After years of ad hoc search testing (where classic algorithms were supported with various methods of automatic query [search pattern] expansion), a conclusion was attained: "One plausible reason that document retrieval has been unable to improve is that the nature of the task requires that systems adopt one size fits all" approaches. [...]. By ignoring the user (or, more accurately, by treating all users identically), systems cannot possibly advance beyond a particular level of accuracy on average for a specific user.' [...] "The goal of this track is to bring the user out of hiding, making him or her an integral part of both the search process and the evaluation." (Allan J., 2004). Further years of testing showed that user-provided relevance feedback can improve search accuracy, but there is a note of bitterness in the conclusion: "Relevance Feedback has been one of the successes of information retrieval research for the past 30 years. It has been proven to be worthwhile in a wide variety of settings, both when actual user feedback is available, and when the user feedback is implicit. However, while the applications of relevance feedback and type of user input to relevance feedback have changed over the years, the actual algorithms have not changed much. Most algorithms are either pure statistical word based (for example, Rocchio or Language Modeling), or are domain dependent. We should be able to do better now, but there have been surprisingly few advances in the area. In part, that's because relevance feedback is hard to study, evaluate, and compare." (Buckley C. and Robertson S., 2009). [13]

The conclusion of the TREC conference is that statistical analysis is insufficient, even if enriched with relevance feedback. It was suggested to more-thoroughly analyze browsed data using semantic analysis instead of analyzing users and their variable needs. It should be noted that semantic analysis does not exclude the use of other techniques (such as the aforementioned statistical analysis and relevance feedback), but rather supplements them. Its advantage is the possibility of clarifying the user query by the means of inexpensive dialogue.

Semantic indexing of data, including textual, is also required by the Semantic Web. It is an idea introduced in 2001 by Berners-Lee, Hendler, and Lassila [5]. Its main goal is to enrich the Web with semantic data (such as rules of logic and relations

between data), thus enabling the creation of semi-intelligent agents that "understand" the processed data and are able to replace humans in some everyday tasks (like making a doctor's appointment). In order to achieve that, an efficient means of indexing large sets of data is required. Köhler et al. [12] present an algorithm for semantic indexing based on ontologies (such as WordNet) designed for English. The SENSEVAL 2 workshop provides much more of them. However, the authors note that all of those algorithms and ontologies are very language specific. To date, no such algorithm or ontology has been developed specifically for the Polish language besides the plWordNet 2.0, which is still a work-in-progress. There are means of extracting data from texts (*e.g.* Conceptual Dependency [8]); however, they require domain-specific templates. A solution based on the ontology could be much more versatile. Such an ontology is known as the *semantic dictionary*.

3. Semantic dictionaries

Cognitive science introduced the concept of semiotic triangle (also known as the triangle of meaning, see Fig. 1) for the purpose of describing the process of understanding a text or speech by humans. It represents the association between a certain form (a word, *e.g.* "dog"), its meaning (a certain kind of animal) ,and an object or a set of objects it applies to (*e.g.* any real dog).

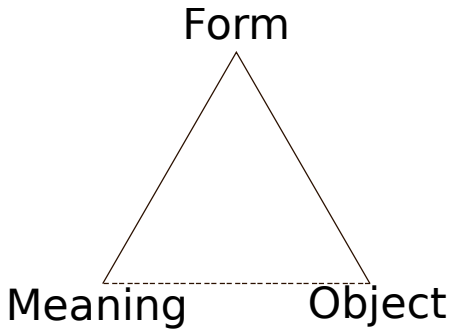

Figure 1. The semiotic triangle.

The human mind can also distinguish connections between abstract concepts. For example, it is usually easy for a human to realize the relation between a car and a driver. Statistical algorithms work only on the level of forms (usually written words). Semantic analysis adds an abstract concept (meaning) to the picture. However, it is impossible to automatically derive the aforementioned connections from forms. They need to be provided by humans. Such ontology is usually called the semantic dictionary.

Technically speaking, semantic dictionaries may take on different forms. Their common characteristic is that they contain concepts and relations between them. The concept is represented by its form (which is usually the base form of the word).

The term "concept" is used to highlight the one word that may represent more than one entity in the dictionary (*e.g.* type of a flying machine and a flat, two-dimensional surface are distinct concepts, both represented by the form "plane"). Some dictionaries may also allow one concept to be represented by more than one form (*e.g.* "mobile phone" and "cell phone"). The relation in the most-basic approach is a triple of two concepts and type of relation. Such a dictionary can be interpreted as a directed graph, also known as a semantic network: concepts are vertices and relations are arcs.

The best-known semantic dictionary is WordNet. The algorithm presented in the latter part of this paper uses Semantic Dictionary of Polish Language.

3.1. WordNet

WordNet was created in 1985 and is still being developed at Princeton University. It was founded (and, for a long time, directed) by the psychology professor George A. Miller [2]. It is available free of charge for browsing and downloading at the project's webpage [1].

WordNet contains about 207 000 word-sense pairs. It includes nouns, verbs, adjectives, and adverbs.

The main idea behind WordNet is the synset. It is an unordered set of all words sharing similar meaning (connected by relation of synonymy; hence, the name). These synsets are in turn connected by the following relations:

- Super-subordinate (hyponymy and hyperonymy) – it denotes that one concept (synset) is more specific than the other. For example, "lion" is more specific than "predator". This relation is transitive, thus arranging concepts into a tree: if "armchair" is a "chair", then it is also "part of furniture". Whatever can be said about a concept is also true for all its children. Tree leaves may be specific entities (*e.g.* people, places). If so, they are called Instances (*e.g.* "Eifle Tower", "Albert Einstein", "Poland"). Other concepts are called Types.
- Part-whole (meronymy) – it denotes that one concept consists of one or more instances of another. For example such relation connects "couch" and "leg", since all couches have legs (usually four of them).
- Troponymy – this relation may connect only verbs. It expresses increasingly-specific manners characterizing an event.
- Antonymy – connects words (mostly adjectives) of opposite meaning, *e.g.* "light" and "dark" or "light" and "heavy".
- Semantic similarity – it reflects any other type of similarity that cannot be described using the above set of relations. It means that connected concepts have a similar meaning, but not exactly the same (*e.g.* "dry" and "parched").
- Morphosemantic relation – connects words (often of different parts of speech) that have a stem of the same meaning (*e.g.* "observe", "observer", "observatory"). A noun-verb pair connected morphosemantically may have an additional label attached, specifying the semantic role of the noun with respect to the verb. For

example, the verb "to paint" may be connected to "a painter", which is agent (a subject of activity) and to "a painting" which is the result (object).

The main disadvantage of WordNet is the very-limited set of relations it contains. It lacks information about the co-occurrence of words and their relations in texts. Such data would be very useful for disambiguation, which is crucial for semantic analysis. However, it is worth noting that there are works which focus on disambiguation based solely on WordNet (*e.g.* [3]). For more information on WordNet, please refer to [15] and [9].

There are numerous versions of WordNet for languages other than English. Some of them are simply translations of the Princeton University version. Others are created from scratch for better focus on lexico-semantic specifics of a language. Among the latter group is a Polish version, named plWordNet 2.0 or *Słowosieć* 2.0, created by the Language Technology Group at the Wrocław University of Technology [14].

3.2. SSJP – The Semantic Dictionary of Polish Language

The Semantic Dictionary of Polish Language (SSJP is the acronym of its Polish name: *Słownik Semantyczny Języka Polskiego*) was created by Computer Linguistics Group at the AGH University of Science and Technology and the Department of Management and Social Communication at Jagiellonian University. Its main idea is similar to that of WordNet. The most basic element of SSJP is called a concept. Each concept corresponds to one meaning of a certain word – it contains the base form of that word and an unique ID. Unlike in WordNet, words with the same meaning are different concepts connected by the synonymy relation [16].

There is one relation distinguished from the others, and it is the category. Currently, there are 14 categories. The remaining relations in SSJP are divided into two groups. The first group consists of paradigmatic relations that correspond to the concepts' meanings. These relations are similar to that of WordNet. The full set contains so far: synonymy, similarity, meronymy (IS A PART OF), holonymy (CONSISTS OF), hyponymy (IS A KIND OF), hiperonymy (IS A – relation opposite to the former one), and conceptual source (determined mostly for abstract objects such as places and characters existing in literature only).

The second group consists of syntagmatic relations. They describe the co-occurrence of words in texts and their relative roles. These relations vary depending on the concept's category. If a concept is an event, it may have the following relations: actor, object, source (FROM) and destination (TO), instrument, time, place, and mood. Source and destination apply to actions with some kind of transfer involved and refer to the direction of this transfer. They do not depend on the initiator of an action. For instance, both "robbery" and "gift" are from the person who loses an object to person who acquires it (although they differ in actor). Concepts from all other categories may have the following relations: destination (purpose of existence), role (special role an object may fulfill, consistent with its general purpose, *e.g.* "throne" is a role of "chair", while "electric chair" is not), action, and state. Both action

and state may also be (but not necessarily) marked as positive (if consistent with the object's destination or expectations) or negative. Action may also be marked as passive if it may be executed on (instead of by) the described object. States are passive by their nature.

Most of the syntagmatic relations may be specified as "related to" another concept. Such a relation between two concepts is true only if this third concept is also involved. For example, a chair may fulfill the role of a throne only when a king is involved.

And important aspect of syntagmatic relations is the possibility of discovering information present implicitly in the text. For example, it enables a recognition (with big probability) that, in the sentence "Jack barked nervously and tried to bite the man's leg", Jack is the name of a dog. This is because two words: "to bark" and "to bite" are both at the right side of a dog's syntagmatic relations (of type action). Since "to bark" is related to a smaller amount of concepts than "to bite" it contains much more information.

4. Proposed algorithm

The following section presents one possible approach to the creation of an automatic text indexing system. It uses semantic analysis based on an ontology.

There are two main tasks set before this system:

1. Assigning a set of semantic indexes to each analyzed text based on its subject.
2. Comparing two sets of semantic indexes, either automatically or manually assigned, in order to specify the similarity of their subjects. This step must not require the full content of the text, but only the aforementioned indexes.

It is assumed that processed texts are written in Polish and are lexically correct.

Although semantic indexes are often called keywords, it is important to note the difference between them. Keywords are simply words from natural language that are the best representation of the text's content. Semantic indexes are concepts, interpreted as in the semantic dictionary. Their roles are the same, but they have two main advantages over traditional keywords. The first one is that they are never ambiguous. While a keyword still may have multiple meanings, a semantic index represents only one of those meanings. The second one is that they form a hierarchical structure contained in the semantic dictionary, which allows better automatic processing. For example, the English term "a cat" may name an animal as well as a part of a ship and (archaically) a type of a ship. If a text is assigned a keyword "a cat", it is impossible to certainly distinguish between those meanings without access to the full text. The semantic index, on the other hand, represents only one of the mentioned meanings, and it is easy to access additional data in the semantic dictionary such as "cat is a mammal, which is an animal" (different types of hierarchies are possible based on different paradigmatic relations).

The proposed algorithm consists of the following steps:

1. Division into words – a text is divided into single words. White spaces and punctuation marks are used as word boundaries. A more-sophisticated algorithm may be used, although most errors caused by such a simple approach are irrelevant during further processing.
2. Transformation into base forms – each word is looked up in the inflectional dictionary (namely CLP or SJP) and replaced by its base (aka. dictionary) form. If there is more that one possible base form, it is replaced by a list of all of them. During our tests, two inflectional dictionaries were used: Polish Inflection Lexicon (commonly refered to as CLP) and Polish Language Dictionary (SJP – *Słownik Języka Polskiego*).
3. Transformation into concepts – each base form is looked up in the semantic dictionary (SSJP) and is replaced by the corresponding concept (or a list of them).
4. Disambiguation – at this point, each word from the original text is replaced by a (flat) list of concepts. If the list is empty, the word may not be processed – it may have been misspelled or is proper name not present in one of the dictionaries. If it has more than one concept, one of them is chosen and the others are discarded. Details of this step are described later in this section.
5. Removal of non-nouns – keywords should be nouns. All concepts that are not nouns are removed from further processing. Part of speech is determined by a heuristic algorithm, since there is not enough data in the semantic dictionary to do this unambiguously.
6. Creation of candidates' set: Each list of concepts in the processed text is turned into a set. Each concept in the set is labeled with its number of occurrences appearing on the list.
7. Reduction into keywords' set – if two or more concepts in the candidates' set are closely related, they are reduced into one concept. The details of this process are described further in this section.
8. Applying a stoplist – if a word (concept) appears in very little of keywords' sets they may be useful for the purpose of searching, but not for text comparison. Keywords appearing in very large number of keywords' sets are useless for both of these purposes. Therefore, the most-frequent keywords are deleted from all sets in which they appear. It is believed that the threshold should be adjusted to delete words with a total frequency of about 20%. The stoplist (the list of concepts to be deleted) may be provided externally or computed (if there is a sufficient number of texts to do so).

4.1. Words disambiguation

This is one of the most important and most problematic parts of the algorithm. Each word from the original text is replaced with a list of concepts. The correct one must be selected (there is always exactly one). To do this for each candidate, concepts that confirm it are counted. A concept is considered to confirm a candidate if it appears

in the text and is related (either directly or indirectly) with the candidate. Types of relations that are considered (most likely syntagmatic relations), and the maximum number of intermediate concepts are parameters of the algorithm. It is also possible to weigh the relations. The candidate with the most confirmations is used for further processing. Left to right processing of text may lead to situations where a concept is confirmed by another one that will be deleted later during this step (but this is unlikely). The way to avoid this problem is to administer a complete search, but this would greatly increase the computational complexity of the solution.

4.2. Reduction of index sets

In order to find possible reductions, a relations graph is built. Each concept that belongs to a candidate set receives a weight equal to its number of occurrences in the original text, while all remaining concepts have a weight of 0. Some kind of dampening function may be used (*e.g.* square root, logarithm, signum). For each concept in candidate set (c), weights of all its direct and indirect neighbors are increased by $w/\sqrt{n+1}$, where n is number of edges between c and said concept, and w is the weight of c. Then, every candidate is replaced with nearest concept with greater weight (if such exists). This process continues until no replacement can be made. During this step, only certain relations are considered (the default behavior is to use only paradigmatic relations that do not narrow the meaning).

The decision whether a concept replacement should occur during indexing is a major question. The algorithm described above reduces only multiple concepts into a single one. However, sometimes it is desired to replace a single concept with another with a wider meaning. For instance, "alsatian" could be replaced with "dog", but not necessarily with "animal" (although both would be correct). It is also possible to provide keywords in a tree-like fashion, which would provide a higher abstraction level without loss of more-precise information.

4.3. Keywords' comparison

The comparison of sets of keywords is an independent part of the system. For every two sets of keywords, the system returns one number describing their similarity. Identical sets receive a mark of 0, and completely different receive a mark of 1. The simmilarity function satisfies the conditions of mathematical definition of distance. The purpose of this part of the algorithm is finding texts of similar topics (very similar keywords' sets) based on the keywords assigned either automatically (*e.g.* by the algorithm described above) or by hand.

Comparison is done in two steps. First is to assign distance to every pair of concepts, where one is from the first set and the second is from the second set. This distance ranges from 0 to 1 and depends on the number and type of relations connecting these concepts. If there is more than one path that connects these concepts, the shortest one is used (with the lowest distance, not necessarily containing the least number of edges). In the second step, for each concept in one set (smaller one, if their

sizes differ) algorithm selects a concept from second set with the smallest distance. The final result is a mean value of distances between the chosen pairs.

This part of the algorithm makes strong use of the semantic dictionary's paradigmatic relations. They allow text about airplanes (with index "plane") to be associated with texts about helicopters rather than about mathematical planes. Such a conclusion would be much more difficult for statistical algorithms operating on the level of forms (written words), not meanings (semantic concepts).

5. Mathematical formulation

Let C be a set of all concepts.

Each semantic relation between concepts in the semantic dictionary is an ordered tuple (c_1, c_2, r), where c_1 and c_2 are concepts ($c_1, c_2 \in C$) and r is a type of relation ($r \in RT$, where $RT := \{CATEGORY, SYNONYMY, IS\_A\_PART\_OF, \ldots, STATE\_NEGATIVE\}$). R is the set of all semantic relations.

5.1. Words disambiguation

Concept c_1 confirms concept c_2 ($c_1 CONF c_2$) $:\Leftrightarrow \exists_{r_C \in RT_C}(c_2, c_1, r_C) \in R$, where $RT_C \subseteq RT$. Typically RT_C consists of the syntagmatic relations.

Every text is a sequence of words: $T = w_1, w_2, \ldots w_n$

For every word composing a text, a number of candidate concepts is assigned based on word's base form, thus transforming a text into sequence of sets of concepts: $T^I = C_1, C_2, \ldots, C_n$, where

$$\forall_{i \in 1 \ldots n} \colon C_i = \{c_{i1}, c_{i2}, \ldots c_{ik_i}\} \wedge k_i \in \mathbb{N} \wedge \forall_{j \in 1 \ldots k_i} : c_{ij} \in C$$

The next step is to reduce every candidate set C_i into a single concept c_i. It is the candidate that has the most neighboring concepts and candidates confirming it.

$conf(c_{ij}) = |\{k \colon kCONFc_{ij} \wedge k \in \{c_1, c_2, \ldots, c_{i-1}\} \cup C_{i+1} \cup C_{i+2} \cup \ldots \cup C_n\}|$

$c_i = \mathrm{argmax}_{c_{ij} \in C_i}(conf(c_{ij}))$

$T^{II} = c_1, c_2, \ldots c_n$

5.2. Reduction of index sets

After this step, the sequence of concepts is transformed into a set of concepts with associated weights: $T^{III} = \{(k_1, w_1), (k_2, w_2), \ldots (k_m, w_m)\}$ in such a way that $\forall_{i=1 \ldots m}$ exists exactly w_i ($\in \mathbb{N}_+$) concepts $c_j \in T^{II} : c_j = k_i$

The reducibility relation is defined as follows:

$c_1 RED c_2 \colon \Leftrightarrow \exists_{r \in RT_R}(c_1, c_2, r) \in R$, where $RT_R \subseteq RT$. Typically RT_R consist of those paradigmatic relations that do not narrow the meaning.

$c_1 RED^k c_2 \colon \Leftrightarrow \exists_{c \in C} : c_1 RED^{k-1} c \wedge cRED c_2$

$c_1 RED^+ c_2 \colon \Leftrightarrow \exists_{k \in \mathbb{N}_+} : c_1 RED^k c_2$

c_1 is reducible to c_2 when $c_1 RED^+ c_2$

$\rho(c) = \sum_{i \in 1 \ldots n} d(c_i, c)$ where $c \in C, c_i \in T^{II}$ and

$$d(c_i, c) = \begin{cases} 0 & , \neg cREDc_i \\ \frac{1}{\sqrt{k+1}} & , k = \min_{l \in \mathbb{N}_+} cRED^l c_i \end{cases}$$

$T^{IV} = \{\gamma_1, \gamma_2, \ldots \gamma_m\}$ and $\forall_{i=1 \ldots m} \exists_{(c_j, w_j) \in T^{III}} \colon c_j RED \gamma_i \wedge \rho(\gamma_i) > w_j \wedge \nexists_{\gamma_i'} \colon (c_j RED^+ \gamma_i' \wedge \rho(\gamma_i') > \rho(\gamma_i))$

T^{IV} is final set of keywords.

For the sake of clarity, removing concepts that are not nouns was omitted here.

5.3. Index comparison

Let there be weighting function $\omega \colon RT \ni r \to \mathbb{R}_+ \cup \{0\}$ and $b \in \mathbb{R}_+$

p is a concept path: $\Leftrightarrow p = c_1 r_1 c_2 r_2 c_3 \ldots r_{n-1} c_n \wedge \forall_{i=1 \ldots n-1} (c_i, c_{i+1}, r_i) \in R$

Concept path p leads from c_1 to c_n.

$P(c, c') := \{p \colon p$ is a concept path $\wedge\, p$ leads from c to $c'\}$

Length of concept path is $len(p) = \sum_{i=1 \ldots n-1} \omega(r_i)$

Distance of concepts:

$$\delta \colon C^2 \ni (c, c') = \begin{cases} 1 & , P(c, c') = \emptyset \\ \min(1, \min_{p_i \in P(c,c')} \frac{len(p_i)}{b}) & , P(c, c') \neq \emptyset \end{cases}$$

$K_1 = \{\gamma_1^1, \gamma_2^1, \ldots \gamma_m^1\}; K_2 = \{\gamma_1^2, \gamma_2^2, \ldots \gamma_n^2\}; 0 < m \leqslant n$

Distance of keywords' sets:

$\Delta(K_1, K_2) := \frac{\sum_{i=1 \ldots m} \min_{j=1 \ldots n} \delta(\gamma_i^1, \gamma_j^2)}{m}$

6. Tests and results

The algorithm presented above has been implemented in Ruby. The program was tested using a corpus of more than fifty thousand short press notes from the Polish Press Agency. The main advantage of this corpus is its variety and high level of correctness of the contained texts. However, the texts do not have assigned keywords; therefore, any evaluation of the algorithm's results is subjective and selective.

The results were analyzed on the following four criteria:

1. The number of keywords assigned. Each text should have 3 to 7 keywords assigned. If there are less then 3, it means that most words in text could not be *understood* (recognized by the semantic dictionary). If there are more than 7, it means that subject of whole text have not been recognized (too little reductions occurred). Neither too little nor too many keywords is desirable, both from the standpoint of the user as well as further analysis.
   As seen in Figure 2, the system tends to assign too many keywords. The maximum of the histogram is set at 7 keywords, but its right side is much bigger (1300 notes have 40 or more keywords assigned). This problem points to an improvement required of both the algorithm itself and the semantic dictionary.

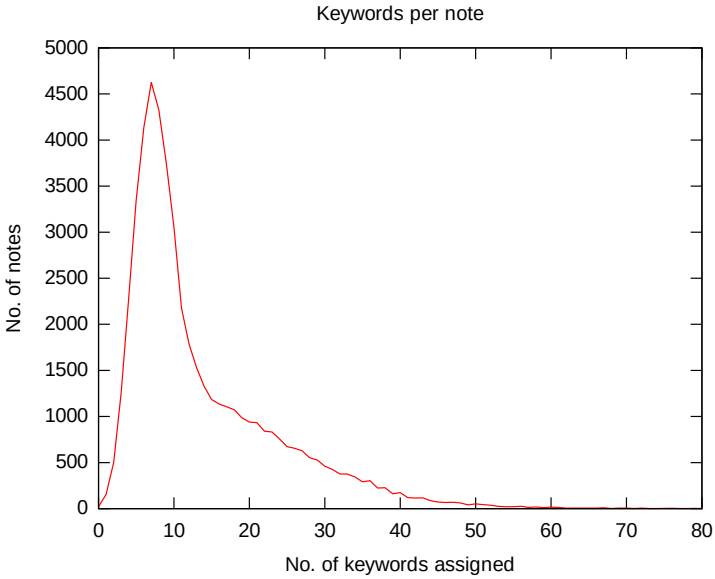

Figure 2. Histogram of keywords count.

2. Frequency of keywords. Figure 3 shows the number of times that each concept has been assigned as a keyword. Concepts never assigned are not included. The x-axis shows the concept's rank, and the y-axis its number of assignments. According to Zipf's law, the number of occurrences of each concept in the corpus should be inversely proportional to its rank. The results presented here show a similar dependency (with $\beta = 7071.32$). It proves that the algorithm does not considerably change the relations present in original texts. However, it also shows that there are many keywords that are not appropriate for this purpose.
3. Accuracy of keywords. As noted before, accuracy could only be evaluated by humans. Most keywords are assigned properly. There are, however, certain repeating errors:

    - Almost all nouns from a note are assigned as keywords. This problem is visible also in Figure 2. Such a situation is caused (inter alia) by a lack of connections between concepts in the semantic dictionary. It is necessary to add such relations wherever possible; however, an algorithmic solution also needs to be developed.
    - A word is not recognized properly. Some keywords are erroneously recognized as nouns. There are two ways to deal with this problem. First is to join the semantic and the inflectional dictionary. Second is to remove all words but nouns before recognizing concepts.

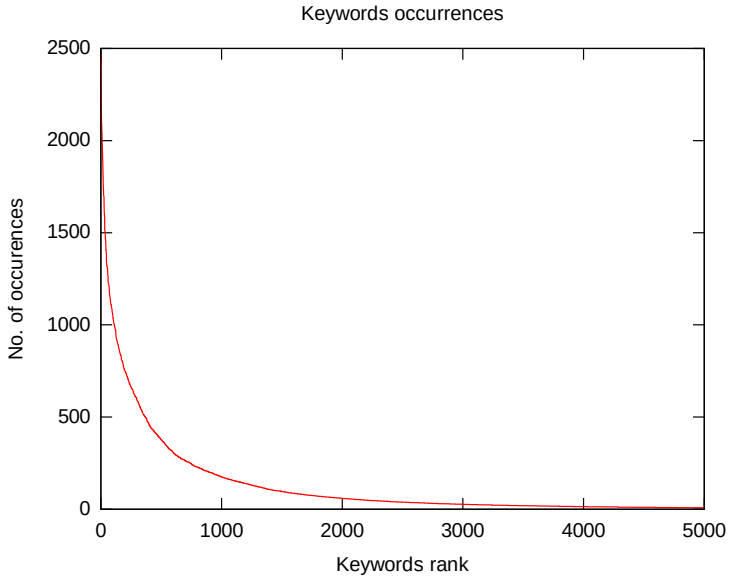

Figure 3. Histogram of keywords frequency.

4. Accuracy of index comparison
   This part also needs to be evaluated subjectively. The results of text comparison based on assigned keywords are satisfactory, even in those areas where keywords are not entirely correct.
   The results of this part of the algorithm were also compared to the results of the SVM algorithm using a cosine measure with two different weigths: binary and tf-idf[1]. A human selected a text (a standard) and then a group of twelve texts most similar to the first one. This group was ordered from most to least similar to the standard. A similarity measure of each text from the selected group to the standard by all three algorithms is presented in Table 1. For increased readability, result '0' is the best (most similar texts) and '1' the worst (in all three cases).
   As it can be seen, the semantic algorithm performed slightly better than the two other algorithms. The most important difference is the spread of values. In the case of the semantic algorithm, they range from 0.3 to 0.5, for the SVM with tf-idf from 0.163 to 0.630 and for the SVM with binary weights from 0.211 to 0.9. When searching for texts similar to a given one, a larger spread increases the number of false positives.

[1]text frequency-inverse document frequency

Table 1
Algorithms' results.

rank	semantic	binary	tf-idf
1	0.300	0.388	0.570
2	0.300	0.211	0.292
3	0.333	0.396	0.518
4	0.333	0.396	0.624
5	0.400	0.910	0.163
6	0.444	0.431	0.223
7	0.462	0.302	0.353
8	0.500	0.295	0.238
9	0.500	0.403	0.528
10	0.444	0.457	0.612
11	0.449	0.317	0.311
12	0.500	0.512	0.630

7. Computational complexity

Each text in a corpus is processed completely independently. Therefore, the computational effort needed for an analysis of the corpus is a sum of efforts needed for each text. The time of a single text's analysis depends on its length (namely: the word count) in the following way:

- The time of splitting the text into word tokens, their transformation to base forms and to concepts depends linearly on the size of the problem (the number of words in a text). Each word in the original text is transformed into a set of concepts. The size of this set is bounded; therefore, every dependency on the total size of these sets may be treated as dependency on a number of words.
- The time of the concepts disambiguation of a single word depends linearly on the length of the sentence. The only certain limit of this length is the length of the full text, so it should be assumed that disambiguation of all words depends on a square of number of words.
- The time of the selection of nouns from the list and changing it into a set depends linearly on the size of the list.
- The time of the reduction of candidates' set depends linearly on its size, therefore approximately linearly on the text's length.

In summary: the time of analysis of a single text depends on a square of its length. The time of comparison between two sets of the semantic indexes depends linearly on their total size.

8. Conclusions

Semantic algorithms are the next step in browsing large sets of textual data. Their most important advantages are the creation of unambiguous indexes and indexes not explicitly present in analyzed text. They also allow the creation of multi-level, hierarchical indexes, *e.g.* animal, dog, alsatian.

The analysis of the system presented in this paper proves that the chosen algorithmic solution is correct and promising, as well as the structure and relations of the Semantic Dictionary of Polish Language. However, it points to necessary improvements in the algorithm itself and the content of the the semantic dictionary.

References

[1] *Princeton University About WordNet.* [online] URL: http://wordnet.princeton.edu/, 2010.

[2] *WordNet.* [online] URL: http://en.wikipedia.org/wiki/WordNet, 2013. [accessed: 2013-04-11 15:11].

[3] Agirre E., Rigau G.: Word sense disambiguation using conceptual density. In: *Proceedings of the 16th International Conference on Computational Linguistics.* Copenhagen, 1996.

[4] Allan J.: HARD Track Overview. In: *TREC 2003, The Twelfth Text REtrieval Conference (TREC 2003) Proceedings.* NIST, 2004.

[5] Berners-Lee T., Hendler J., Lassila O.: The Semantic Web. *Scientific American*, 2001.

[6] Buckley C., Robertson S.: Relevance Feedback Track Overview. In: *TREC 2008. The Seventeenth Text REtrieval Conference (TREC 2008) Proceedings.* NIST, 2009.

[7] Collins M., Singer Y.: *Relational Learning of Pattern-matching Rules*, 1999.

[8] Dorosz K., Korzycki M.: Latent semantic analysis evaluation of conceptual dependency driven focused crawling. In: *The Indect project: Multimedia Communications, Services & Security Conference.* 2012.

[9] Fellbaum C. E.: *WordNet: An Electronic Lexical Database*, 1988.

[10] Figiel A.: *Tekst jako wzorzec informacyjny – automatyczna ocena podobieństwa tematycznego tekstów za pomocą Latent Semantic Analysis.* In: W. Lubaszewski (Ed.), *Słowniki komputerowe i automatyczna ekstrakcja informacji z tekstu*, chap. 9, pp. 165–178, Uczelniane Wydawnictwa Naukowo-Dydaktyczne AGH, Kraków, 2009.

[11] Hardtke D., Wertheim M., Cramer M.: *Demonstration of Improved Search Result Relevancy Using Real-Time Implicit Relevance Feedback*, 2009.

[12] Köhler J., Philippi S., Specht M., Rüegg A.: Ontology based text indexing and querying for the semantic web. *Knowledge-Based Systems*, 19(8): 744–754, 2006.

[13] Lubaszewski W., Dorosz K., Korzycki M.: D4.4. System for Enhanced Search: A Tool for Pattern Based Information Retrieval. In: *The Indect project: European Seventh Framework Programme FP7-218086-Collaborative Project*, 2009.

[14] Maziarz M., Piasecki M., Szpakowicz S.: Approaching plWordNet 2.0. In: *Proceedings of the 6th Global Wordnet Conference*. 2012.

[15] Miller G. A.: WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11): 39–41, 1995.

[16] Pohl A.: *Rozstrzyganie wieloznaczności, maszynowa reprezentacja znaczenia wyrazu i ekstrakcja znaczeń.* In: W. Lubaszewski (Ed.) *Słowniki komputerowe i automatyczna ekstrakcja informacji z tekstu*, 241–256. Kraków, 2009.

[17] Xing W., Ghorbani A.: Weighted PageRank Algorithm. In: *Second Annual Conference on Communication Networks and Services Research, 2004. Proceedings*, pp. 305–314. 2004.

Affiliations

Zbigniew Kaleta
AGH University of Science and Technology, Krakow, Poland, zkaleta@agh.edu.pl

Received: 8.04.2013
Revised: 17.05.2013
Accepted: 20.12.2013