AUTOMATYKA • 2006 • Tom 10 • Zeszyt 3

Marcin Kolibabka*, Andrzej Cader*

Metoda wymuszania wewnętrznych wzorców w jednokierunkowej sieci klasyfikującej

1. Wstęp

Używane powszechnie jednokierunkowe, wielowarstwowe nieliniowe sieci neuronowe zwane MLP (*Multi Layered Perceptron*), można też spotkać polską nazwę wielowarstwowe sieci perceptronowe [1] zawdzięczają swoja popularność prostocie implementacji. Jednak aby skutecznie wykorzystać ich zalety konieczne jest opracowanie odpowiedniej metody dla zadania struktury sieci oraz nauczenie jej działania. Zasady przedstawione pod koniec lat 80 XX wieku opisujące możliwości sieci neuronowych (każda ograniczona funkcja ciągła może być aproksymowana z dowolnie małym błędem przez sieć z jedną warstwą ukrytą [2, 5] ponadto dowolna funkcja może być aproksymowana z dowolną dokładnością przez sieć z dwoma warstwami ukrytymi [2, 4]) oraz opracowanie algorytmu wstecznej propagacji błędu EBP (*Error Back Propagation*) [4] bezpośrednio przyczyniły się do ich rozpowszechnienia po wcześniejszym, wieloletnim porzuceniu badań nad nimi. Do wielu zastosowań predysponuje je również stosunkowo prosta struktura nauczanej sieci, która w połączeniu z odpowiednim zrównolegleniem przetwarzania sygnałów pozwala na szybkie uzyskiwanie reakcji sieci na zmianę parametrów wejściowych.

Zadania klasyfikacji są jednym z podstawowych problemów rozwiązywanych przy pomocy sieci perceptronowych. W procesie uczenia sieć „zapamiętuje" wzorce z zbioru uczącego oraz uogólnia ich postacie tak, aby rozpoznawać również nowe dane wejściowe. Dzieje się tak oczywiście w idealnie przebiegającym procesie uczenia. W praktycznych zastosowaniach do takiego optymalnego rozwiązaniach trzeba dochodzić na drodze eksperymentowania z parametrami uczenia oraz strukturą sieci. Korzystne zatem jest każde ograniczenie ilości doświadczeń. Opisana w pracy metoda pozwala na ograniczenie wpływu nieoptymalnej struktury sieci oraz zwiększa zakres wartości parametru prędkości i pędu uczenia przy których uzyskuje się korzystne wyniki uczenia. Metoda ta stanowi poszerzenie klasycznej metody wstecznej propagacji błędów o wymuszenie wspólnego wzorca dla grupy wag.

* Instytut Kształcenia na Odległość, Wyższa Szkoła Humanistyczno-Ekonomiczna w Łodzi

2. Momentowa metoda wstecznej propagacji błędu

Dzięki swojej efektywności w połączeniu z względnie prostą strukturą algorytmu metody z grupy wstecznej propagacji błędu cieszą się dużą popularnością w wielu obszarach zastosowań. Podczas uczenia sieci dąży się do minimalizacji nieliniowej funkcji celu. W przypadku metody wstecznej propagacji błędu dąży się do minimalizacji błędu E będącego sumą błędów E_i obliczonych dla każdego wektora danych uczących. Oblicza się go ze wzoru dla i-tego wektora danych gdzie out_{ki} wartość na k-tym wyjściu sieci a $wzor_{ki}$ wartość oczekiwana na tym wyjściu

$$E_i = \frac{1}{2}\sum_{k=1}^{m}(out_{ki} - wzor_{ki})^2 \tag{1}$$

Po przetworzeniu przez sieć n-tego wektora ze zbioru uczącego następuje modyfikacja neuronów w sieci w oparciu o metodę największego spadku. Oznacza to, że wagi neuronów są modyfikowane na podstawie wzoru

$$w^{(n+1)} = w^{(n)} - \eta \nabla E_i\left(w^{(n)}\right) \tag{2}$$

gdzie:

$w^{(n+1)}$ – wektor wag w (n+1)-kroku algorytmu,
$w^{(n)}$ – wektor wag w n-tym kroku algorytmu,
η – współczynnik prędkości uczenia,
∇E_i – gradient funkcji E_i w punkcie $w^{(n)}$.

Modyfikacja tej metody poprzez wprowadzenie parametru zwanego momentum (pędem) pozwala na uczynienie metody bardziej równomierną i mniej wrażliwą na lokalne zmiany przebiegu błędu. Osiąga się to poprzez wprowadzenie do każdej zmiany czynnika powiązanego ze zmianą z poprzedniego kroku. Opisując to wzorem, otrzymujemy

$$\Delta w^n = -\eta \nabla E_i\left(w^n\right) + \mu \Delta w^{n-1} \tag{3}$$

gdzie μ to współczynnik pędu (momentum).

Tak zmodyfikowaną metodę nazywa się momentową metodą wstecznej propagacji błędu MEBP (*Momentum Error Back Propagation*).

Upraszczając wzór na modyfikację pojedynczej wagi w tym algorytmie, do postaci stosowanej bezpośrednio w algorytmie otrzymujemy

$$w_{i+1} = w_i - \eta * err * \frac{df(u)}{du} * out \tag{4}$$

gdzie:

w_{i+1} – wartość wagi po (i+1)-kroku uczenia,
w_i – wartość wagi w otrzymana w poprzednim kroku,

err – wartość błędu na neuronie, do którego należy waga uzyskana z wstecznej propagacji błędu,
u – wartość podawana na funkcje aktywacji,
out – wartość wyjścia neuronu.

Jak widać wartość, wagi zależy bezpośrednio tylko od jej poprzedniej wartości. Pośrednio można powiedzieć, że dzięki wstecznej propagacji błędu również wagi z dalszych warstw mają wpływ na zmianę w trakcie uczenia. Jednak zmiana nie zależy od wag leżących w wcześniejszych warstwach ani w tej samej warstwie. W metodzie wymuszania wzorców proponujemy dodanie do procesu uczenia grupy wag powiązanych ze sobą.

3 . Metoda wymuszania wewnętrznych wzorców

Klasyczną metodę wstecznej propagacji błędu poszerzamy o dodanie warunku, który wprowadza dodatkową zależność w grupie wybranych wag. W niniejszej pracy przedstawiono postać algorytmu z warunkiem postaci

$$\sum_{w \in B} w = \text{const} \tag{5}$$

gdzie B oznacza zbiór wybranych do „blokady" wag.

Słowo blokada użyte zostało w cudzysłowie, ponieważ nie blokuje się pojedynczych wartości wag, a tylko ich sumę. Czyli w procesie uczenia będą one mogły się zmieniać jednak tak by ich suma pozostawała stała – zmiana zależy wtedy również od zmian pozostałych z grupy.

Dodanie warunku powoduje konieczność modyfikacji zależności na zmianę pojedynczej wagi. Warunek ten można uwzględnić przy pomocy metody mnożników Lagrange'a. Jeżeli poszukiwane jest ekstremum funkcji $y = f(x)$ przy warunku $g(x)=0$ gdzie f: $R^n \rightarrow R$ oraz g: $R^n \rightarrow R$, to należy zminimalizować funkcję Lagrange'a postaci

$$L(x,\lambda) = f(x) + \lambda g(x) \tag{6}$$

oznaczając jako zbiór A

$$A = \{x \in R^n : g(x) = 0\} \tag{7}$$

Można zauważyć, iż

$$\forall_{x \in A} \; L(x,\lambda) = f(x) \tag{8}$$

Obie te funkcje w zbiorze A (czyli przy warunku $g(x) = 0$) mają takie same ekstrema. Funkcją f jest funkcja ze wzoru (1), natomiast warunek g określa zależność (5), co daje nam funkcję L postaci

$$L_i = \frac{1}{2}\sum_{k=1}^{m}\left(out_{ki} - wzor_{ki}\right)^2 + \lambda\left(\sum_{w\in B} w - C\right) \tag{9}$$

gdzie C jest sumą wartości wybranych do blokowania wag sprzed procesu uczenia. Współczynnik λ jest mnożnikiem Lagrange'a.

Po przekształceniach analogicznych jak przy wyprowadzaniu zależności dla klasycznej metody wstecznej propagacji błędu oraz wykorzystaniu (5) otrzymujemy do rozwiązania układ równań liniowych:

$$\begin{aligned}
&w^p_{1,i+1} + w^p_{2,i+1} + \cdots + w^p_{n,i+1} = C \\
&w^p_{1,i+1} - \lambda = w_{1,i+1} \\
&w^p_{2,i+1} - \lambda = w_{2,i+1} \\
&\cdots \\
&w^p_{n,i+1} - \lambda = w_{n,i+1}
\end{aligned} \tag{10}$$

gdzie $w^p_{1,i+1}, w^p_{2,i+1}, ..., w^p_{n,i+1}$ to szukane wartości wag w kroku i+1, $w_{1,i+1}, w_{2,i+1}, ..., w_{n,i+1}$ wartości z zależności (4).

Układ można łatwo rozwiązać, a jego wynik daje nowe wartości wag.

4. Metody doboru blokowanych wag

Mając przedstawiony algorytm modyfikacji wag, można zastosować różne sposoby doboru wag do blokowania. W testach użyto dwóch algorytmicznych kryteriów do wyboru „blokowanych" wag oraz kliku sposobów wyboru ich ilości. Pierwszą metodą był wybór wag o największej wartości bezwzględnej (*maxAbs*) jako tych, które mają ilościowo największą możliwość wpływania na wynik działania sieci.

Druga metoda obejmowała wagi, których zmiana wartości powoduje największą zmianę błędu na wyjściu sieci w stosunku do wartości wagi (*maxRatio*). W testach używano pewnej wartości progowej *delta* o jaką zmieniano wagi, po czym przeprowadzano pełną epokę obliczeń dla danych uczących bez modyfikowania wartości wag. Uzyskany na końcu błąd średni kwadratowy na wyjściu sieci dzielono przez wartość wagi. Do blokady były wybierane te wagi, dla których tak otrzymana wartość była największa.

Ostatnią metodą, już nie algorytmiczną, był ręczny wybór wag. Pokazała ona, iż blokowanie wszystkich wag w sąsiadujących ze sobą neuronach nie polepsza wyników uczenia.

5. Wyniki testów na przykładowych sieciach klasyfikujących

Testy metody były prowadzone na sieciach o strukturze optymalnej dla danego zadania jak i na strukturze nadmiarowej. W pierwszym przypadku różnica pomiędzy metodą bez blokowania, a uczeniem z wymuszaniem wzorca nie była znacząca. Jednak na etapie poszukiwania optymalnej struktury sieci uczenie z wymuszaniem wzorca poprawiało w zauważalny sposób efektywność uczenia. Sieci o zbyt dużej strukturze posiadają mniejszą zdolność klasyfikowania danych, nie występujących w zbiorze uczącym czyli gorzej generalizują problem [6]. Blokowanie zmienia tę sytuację.

Skuteczność uczenia była silnie uzależniona zarówno od doboru ilości wag, jak i sposobu ich wyboru. Testy były prowadzone w sposób następujący: generowano strukturę sieci, która następnie była uczona metodą klasyczną oraz z wymuszaniem wzorca przy takich samych parametrach, czyli z tą samą prędkością uczenia oraz pędem. Kryterium poprawy stanowiła ilość rozpoznanych próbek ze zbioru testowego. Testy prowadzone były na problemie klasyfikacji irysów oraz klasyfikacji „Zoo" (na podstawie 16 cech dzielono zwierzęta na 7 grup). Pliki testowe zawierały odpowiednio 45 i 30 próbek. Eksperymenty dla każdego zestawu blokowanych wag były powtarzane kilkukrotnie i w pracy podane są wyniki średnie z grupy testów.

W przypadku obu zagadnień klasyfikacyjnych blokowanie zestawu zawierającego ponad 90% wag z sieci powodowało, że sieć prawie całkowicie przestawała się uczyć. Brak wpływu blokady był obserwowany przy blokowaniu około połowy wag z sieci wszystkimi metodami. Siec uczyła się porównywalnie do przypadku braku blokady.

Kolejna próba polegała na zwiększaniu ilości blokowanych wag co 5000 iteracji. Przy sposobie *maxAbs* również nie uzyskano poprawy działania sieci. Natomiast przy *maxRatio* i sieci klasyfikującej dla problemu „Zoo" zaobserwowany został mniejszy wpływ doboru prędkości uczenia na działanie sieci. O ile przy klasycznej metodzie ilość rozpoznanych próbek zawierała się w przedziale od 10 do 25 w zależności od dobranych prędkości uczenia n i pędu, to przy takich samych parametrach dla metody z wymuszaniem wzorca przedział ten był od 20 do 25.

Efekt ten miał miejsce również w przypadku, gdy w pierwszym kroku blokowano połowę wag w sieci wybierając za pomocą *maxRatio*, a w kolejnych krokach zmniejszano ich ilość o połowę. Tym razem zysk był widoczny obu zagadnieniach. Najlepsze wyniki uzyskiwano przy blokowaniu połowy wag w pierwszym cyklu uczenia (wybrane metodą *maxRatio*), a w kolejnych krokach zbiór był zmniejszany poprzez usuniecie połowy wag. Uzyskane wyniki były średnio o 8,7% lepsze od metody klasycznej.

6. Wnioski i dalsze kierunki badań

Blokując wagi, można uczyć równie skutecznie sieci o strukturze nadmiarowej co sieci o strukturze optymalnej. Szczególnie było to widoczne w problemie klasyfikacyjnym „Zoo" gdzie różnica była mniejsza od błędu wynikającego z zaokrąglenia wyników. Ponadto przy blokowaniu wag można uzyskać dużo wyższą stabilizację procesu uczenia dla większego spektrum wartości prędkości i pędu uczenia.

Kolejne etapy pracy będą obejmowały wyszukanie innych kryteriów wybierania wag do blokady oraz zaimplementowanie możliwości blokowania na raz więcej niż jednej grupy wag.

Literatura

[1] Tadeusiewicz R.: *Sieci neuronowe*. Warszawa, Akademicka Oficyna Wydaw. RM 1993
[2] Cybenko G.: *Approximation by Superpositions of a Sigmoidal Function.* Mathematics of Control, Signals, and Systems, vol. 2, 1989, 303–314
[3] Rutkowska D., Piliński M., Rutkowski L.: *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte*. Warszawa, PWN 1997
[4] Rumelhart D., Hinton G., Williams R.: *Learning Internal Representations by Error Propagation*. Parallel Distributed Processing, vol. 1, 1986, 318–362
[5] Hornik K., Stinchcombe M., White H.: *Multilayer feedforward networks are universal approximators*. Neural Networks, 2, 1989, 359–366
[6] Rutkowski L.: *Metody i techniki sztucznej inteligencji.* Warszawa, PWN 2005