

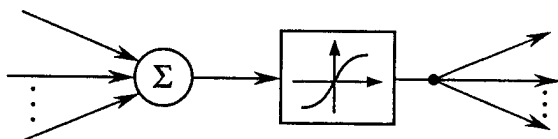
Seminarium Szkoleniowe — Optymalizacja Sieci Neuronowe

1 Wstęp

Sieć neuronowa (sztuczna sieć neuronowa)

1. klasa algorytmów matematycznych, pozwalająca na przeprowadzenie pewnych obliczeń
2. sztuczna sieć naśladująca biologiczne sieci neuronowe żywych organizmów
3. graf skierowany z odpowiednio określoną rolą węzłów i krawędzi
4. układ elementów przetwarzających, nazwanych neuronami, w których wyjścia każdego neuronu są połączone poprzez wagi z wejściami wszystkich neuronów, w tym także w jego własnym wejściu. Wagi są liczbami rzeczywistymi. Jeżeli waga jest równa 0, można powiedzieć że połączenie nie istnieje.

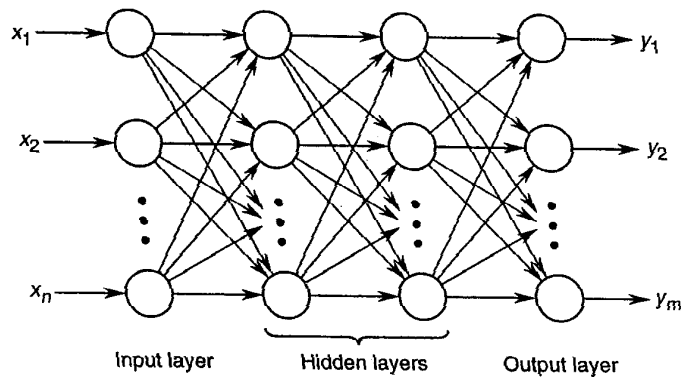
Pojedynczy neuron jest pewnym odwzorowaniem. Potencjał wejściowy (wewnętrzny) jest sumą wszystkich sygnałów wejściowych. Potencjał wyjściowy (zewnętrzny) jest pewną transformacją potencjału wejściowego — jest to funkcja aktywacji neuronu (rys 1).



Rysunek 1: schemat neuronu

Sieci neuronowe można podzielić na dwie podstawowe grupy ze względu na ich topologię:

- *Sieci jednokierunkowe* z elementarną konfiguracją bez sprzężenia zwrotnego. Neurony są ułożone w warstwach, wyjście neuronu z pewnej warstwy może być połączone tylko z wejściami neuronów znajdujących się w warstwie bezpośrednio następnej (rys. 2).



Rysunek 2: schemat sieci jednokierunkowej

Warstwa wejściowa (input layer) - pierwsza warstwa sieci;

Warstwa wyjściowa (output layer) - ostatnia warstwa sieci;

Warstwy ukryte (hidden layers) - pozostałe warstwy sieci,

- *Sieci rekurencyjne* - nie można mówić o ułożeniu w warstwy. Każdy neuron może być połączony z każdym.

Sieci neuronowe mogą być traktowane jako implementacje pewnego odwzorowania $\mathbb{R}^n \rightarrow \mathbb{R}^m$, gdzie n jest liczbą wejść x_1, \dots, x_n , a m jest liczbą wyjść y_1, \dots, y_m . Tak określone odwzorowanie zależy od wszystkich wag sieci neuronowej.

Najważniejszym procesem sieci neuronowych jest uczenie, które wymusza określone zachowanie na zadane sygnały wejściowe. Program sieci, czyli to co sieć robi, jest zapisane w jej wagach, dlatego też uczenie polega na adaptacji wartości wag związanych z krawędziami sieci.

Wyróżnia się kilka rodzajów uczenia:

- *Uczenie baczowe* - jednorazowy zapis. Wagi dostrajane są w wyniku jednej sesji treningowej podczas którego korzysta się z całego zbioru wejściowego.
- *Uczenie nadzorowane*, zwane również uczeniem z nauczycielem. Przy nowych danych wejściowych nauczyciel podpowiada pożądaną odpowiedź.
- *Uczenie nienadzorowane*, czyli uczenie bez nauczyciela. Pożądana odpowiedź nie jest znana. Taki typ uczenia stosuje się w sieciach, które mają wychwycić pewne podobieństwo w zbiorze wejściowym.

Popularna metoda uczenia sieci neuronowych, nazywana metodą *propagacji wstecznej* (*backpropagation*) opiera się na problemie optymalizacji i wykorzystuje informację o gradientach.

2 Uczenie pojedynczego neuronu

Rozważmy pojedynczy neuron. Odwzorowuje on następującą funkcję liniową: $\mathbb{R}^n \rightarrow \mathbb{R}$:

$$y = \sum_{i=1}^n w_i x_i = x^T w$$

gdzie $x = [x_1, \dots, x_n]^T \in \mathbb{R}^n$ jest wektorem sygnałów wejściowych, $y \in \mathbb{R}$ jest sygnałem wyjściowym, a $w = [w_1, \dots, w_n]^T \in \mathbb{R}^n$ jest wektorem wag.

Dla danej funkcji $F : \mathbb{R}^n \rightarrow \mathbb{R}$ chcemy znaleźć wartości wag, tak aby neuron jak najlepiej przybliżał odwzorowanie F . By to osiągnąć używamy zbioru p par treningowych $\{(x_{d,1}, y_{d,1}), \dots, (x_{d,p}, y_{d,p})\}$, gdzie $x_{d,i} \in \mathbb{R}^n$ oraz $y_{d,i} \in \mathbb{R}$ dla $i = 1, \dots, p$. Dla każdego i , $y_{d,i} = F(x_{d,i})$ jest oczekiwaną wartością, jeżeli na wejściu dane było $x_{d,i}$.

Zatem problem uczenia może być przedstawiony jako następujące zadanie optymalizacyjne:

$$\min \frac{1}{2} \sum_{i=1}^p (y_{d,i} - x_{d,i}^T w)^2$$

gdzie minimalizacja jest po $w = [w_1, \dots, w_n]^T \in \mathbb{R}^n$.

Minimalizowana funkcja jest sumą kwadratów błędów między wyjściem oczekiwanym a rzeczywistym. Czynniki $\frac{1}{2}$ dodany jest ze względu na wygodę.

Zdefiniujmy następującą macierz $X_d \in \mathbb{R}^{n \times p}$, $X_d = [x_{d,1} \dots x_{d,p}]$ oraz wektor $y_d \in \mathbb{R}^p$, $y_d = [y_{d,1}, \dots, y_{d,p}]^T$. Wtedy problem optymalizacji można

zapisać jako

$$\min \frac{1}{2} \|y_d - X_d^T w\|^2$$

Rozpatrzmy dwa przypadki: $p \leq n$ oraz $p > n$.

2.1 $p \leq n$

Dla ułatwienia zakładamy że rząd $X_d^T = p$. W tym przypadku nieskończenie wiele punktów spełnia równanie $y_d = X_d^T w$. Możliwym kryterium wyboru rozwiązania optymalnego jest minimalizacja normy rozwiązania. W tym przypadku rozwiązaniem jest $w^* = X_d(X_d^T X_d)^{-1} y_d$. Efektywnym iteracyjnym algorytmem był algorytm Kaczmarza.

$$w^{(k+1)} = w^{(k)} + \mu \frac{e_k x_{d,R(k)}}{\|x_{d,R(k)}\|^2}$$

gdzie $w^{(0)} = 0$ i

$$e_k = y_{d,R(k)} - x_{d,R(k)}^T w^{(k)}$$

$R(k)$ jest unikalną liczbą całkowitą ze zbioru $\{0, \dots, p-1\}$ taką, że $k = lp + R(k)$ dla pewnej liczby całkowitej l .

Ten algorytm uczenia neuronów liniowych był używany przez Widrowa i Hoffa. Metoda ta uczenia pojedynczego neuronu jest często nazywana metodą uczenia Adaline (adaptive linear element).

2.2 $p > n$

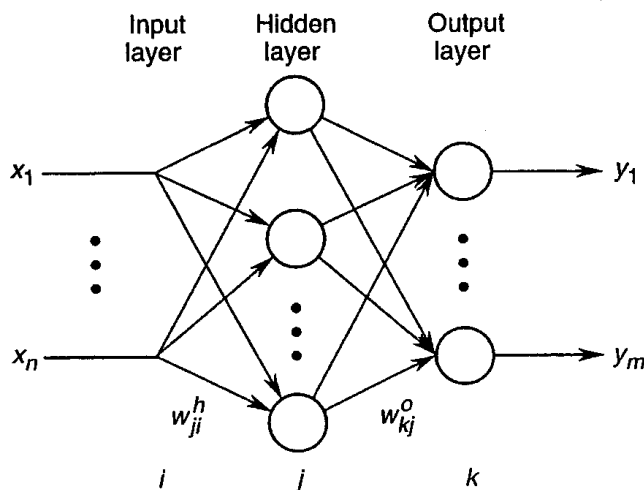
Zakładamy że rząd $X_d^T = n$. W tym przypadku funkcja

$$\frac{1}{2} \|y_d - X_d^T w\|^2$$

jest zwykłą, wypukłą funkcją kwadratową na w , ponieważ $X_d X_d^T$ jest macierzą dodatnio określoną. To zadanie można rozwiązać przy pomocy wielu nieograniczonych algorytmów optymalizacyjnych, np. algorytmu gradientowego.

$$w^{(k+1)} = w^{(k)} + \alpha_k X_d e^{(k)}$$

gdzie $e^{(k)} = y_d - X_d^T w^{(k)}$.



Rysunek 3: schemat sieci jednokierunkowej

3 Algorytm propagacji wstecznej

Dla uproszczenia będziemy rozważać algorytm z 3 warstwami: warstwą wejściową, ukrytą i wyjściową (rys. 3).

Oznaczenia:

n — liczba neuronów w warstwie wejściowej

l — liczba neuronów w warstwie ukrytej

m — liczba neuronów w warstwie wyjściowej

x_i — sygnał wejściowy sieci, $i = 1, \dots, n$

y_s — sygnał wyjściowy sieci, $s = 1, \dots, m$

z_j — sygnał wyjściowy warstwy ukrytej, $j = 1, \dots, l$

w_{ji}^h — wagi pomiędzy warstwą wejściową a ukrytą, $i = 1, \dots, n$, $j = 1, \dots, l$

w_{sj}^o — wagi pomiędzy warstwą ukrytą a wyjściową, $j = 1, \dots, l$, $s = 1, \dots, m$

f_j^h — funkcja aktywacji j neuronu warstwy ukrytej

f_s^o — funkcja aktywacji s neuronu warstwy wyjściowej

Neurony warstwy wejściowej nie wykonują żadnych obliczeń, przekazują tylko sygnały do warstwy ukrytej.

Każda funkcja aktywacji f_j^h , f_s^o jest funkcją $\mathbb{R} \rightarrow \mathbb{R}$.

Dla danych wag w_{ji}^h oraz w_{sj}^o sieć neuronowa odwzorowuje funkcję $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$.

Wyjście neuronu j z warstwy ukrytej jest równe:

$$z_j = f_h^j \left(\sum_{i=1}^n w_{ji}^h x_i \right)$$

Sygnał wyjściowy y_s jest równy

$$y_s = f_o^s \left(\sum_{j=1}^l w_{sj}^o z_j \right) = f_o^s \left(\sum_{j=1}^l w_{sj}^o f_h^j \left(\sum_{i=1}^n w_{ji}^h x_i \right) \right) = F_s(x_1, \dots, x_n)$$

co można zapisać jako:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} F_1(x_1, \dots, x_n) \\ \vdots \\ F_m(x_1, \dots, x_n) \end{bmatrix}$$

Rozważmy teraz problem uczenia sieci neuronowej. Zbiór uczący jest zbiorem par (x_d, y_d) , gdzie $x_d \in \mathbb{R}^n$ oraz $y_d \in \mathbb{R}^m$.

Uczenie sieci polega na dobraniu wag, tak by sygnał wyjściowy generowany przez sieć był możliwie bliski oczekiwanej wartości. Formalnie można to zapisać jako:

$$\min \frac{1}{2} \sum_{s=1}^m (y_{ds} - y_s)^2$$

gdzie y_s $s = 1, \dots, m$ są sygnałami wyjściowymi generowanymi przez sieć jeżeli na wejściu podano x_{d1}, \dots, x_{dn} .

$$y_s = f_o^s \left(\sum_{j=1}^l w_{sj}^o f_h^j \left(\sum_{i=1}^n w_{ji}^h x_i \right) \right)$$

Dla ułatwienia zapisu jako w oznaczmy wektor wag:

$$w = \{w_{ij}^h, w_{sj}^o : i = 1, \dots, n, j = 1, \dots, l, s = 1, \dots, m\}$$

Zadanie optymalizacji można zatem przedstawić jako:

$$E(w) = \frac{1}{2} \sum_{s=1}^m (y_{ds} - y_s)^2$$

$$E(w) = \frac{1}{2} \sum_{s=1}^m \left(y_{ds} - f_o^s \left(\sum_{j=1}^l w_{sj}^o f_h^j \left(\sum_{i=1}^n w_{ji}^h x_i \right) \right) \right)^2$$

Aby rozwiązać to zadanie, będziemy używać metody gradientowej z ustalonym krokiem. Obliczmy najpierw pochodną częściową po w_{sj}^o :

$$E(w) = \frac{1}{2} \sum_{p=1}^m \left(y_{dp} - f_o^p \left(\sum_{q=1}^l w_{pq}^o z_q \right) \right)^2$$

dla każdego $q = 1, \dots, l$ $z_q = f_h^q(\sum_{i=1}^n w_{qi}^h x_{di})$

$$\frac{\partial E}{\partial w_{sj}^o}(w) = -(y_{ds} - y_s) f_s^{o'} \left(\sum_{q=1}^l w_{sq}^o z_q \right) z_j$$

$f_s^{o'} : \mathbb{R} \rightarrow \mathbb{R}$ jest pochodną funkcji f_s^o . Dla uproszczenia przyjmijmy że:

$$\delta_s = (y_{ds} - y_s) f_s^{o'} \left(\sum_{q=1}^l w_{sq}^o z_q \right)$$

δ_s może być traktowane jako błąd wyjścia przeskalowany przez $f_s^{o'}(\sum_{q=1}^l w_{sq}^o z_q)$ zatem

$$\frac{\partial E}{\partial w_{sj}^o}(w) = -\delta_s z_j$$

Policzmy teraz częściową pochodną po w_{ji}^h

$$\frac{\partial E}{\partial w_{ji}^o}(w) = \sum_{p=1}^m (y_{dp} - y_p) f_o^{p'} \left(\sum_{q=1}^l w_{pq}^o z_q \right) w_{pj}^o f_j^{h'} \left(\sum_{r=1}^n w_{jr}^h x_{dr} \right) x_{di}$$

gdzie $f_j^{h'} : \mathbb{R} \rightarrow \mathbb{R}$ jest pochodną funkcji f_j^h .

Zgodnie z metodą gradientową wagi w_{sj}^o oraz w_{ji}^h są uaktualniane zgodnie z wzorami:

$$w_{sj}^{o(k+1)} = w_{sj}^{o(k)} + \eta \delta_s^{(k)} z_j^{(k)}$$

$$w_{ji}^{h(k+1)} = w_{ji}^{h(k)} + \eta \left(\sum_{p=1}^m \delta_p^{(k)} w_{pj}^{o(k)} \right) f_j^{h'}(v_j^{(k)}) x_{di}$$

gdzie η jest ustalonym krokiem,

$$v_j^k = \sum_{i=1}^n w_{ji}^{h(k)} x_{di}$$

$$z_j^{(k)} = f_j^h(v_j^{(k)})$$

$$y_s^{(k)} = f_s^o \left(\sum_{g=1}^l w_{sg}^{o(k)} z_g^{(k)} \right)$$

$$\delta_s^{(k)} = (y_{ds} - y_s) f_s^{o'} \left(\sum_{q=1}^l w_{sq}^o z_q \right)$$

Powyższy algorytm nosi nazwę algorytmu „propagacji wstecznej” (back-propagation).

4 Zastosowanie Sieci Neuronowych do Zadań Optymalizacyjnych

4.1 Sieć Hopfielda...

Kluczowym pojęciem związanym z rozwiązywaniem problemów optymalizacyjnych jest pojęcie energii.

$$E = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N t_{ij} v_i v_j - \sum_{i=1}^N I_i v_i + \frac{1}{RC} \sum_{i=1}^N C(v_i)$$

$$C(x) = \int_{\frac{1}{2}}^x g^{-1}(\eta) d\eta$$

Energia jest funkcją nierosnącą w czasie, zatem po pewnym czasie zostanie znaleziony pewien stabilny punkt — minimum funkcji. Sieć realizuje algorytm minimalizacji kierunkowej wzdłuż największego spadku E , który nie gwarantuje znalezienia minimum globalnego. Znalezione minimum zależy w sposób deterministyczny od wyboru punktu startowego.

4.2 Problem komiwojażera (TSP)

Dla n miast, połączonych ze sobą znaleźć najkrótszą trasę, która przechodzi przez wszystkie miasta dokładnie raz, kończy i zaczyna się w tym samym mieście. (Dla danego niezorientowanego grafu pełnego K_n z wagami znaleźć cykl Hamiltona o minimalnej wadze)

4.2.1 kodowanie miasto-kolejność

Reprezentacja — sieć złożona z n^2 neuronów reprezentowanych przy pomocy macierzy kwadratowej V o wymiarach $n \times n$, której elementy należą do zbioru $\{0, 1\}$. Jeżeli $v_{ij} = 1$, to miasto i -te jest odwiedzane jako j -te w kolejności.

Funkcja energii przyjmuje postać:

$$\begin{aligned}
E = & \frac{A}{2} \sum_{x=1}^n \sum_{i=1}^n \sum_{j=1, j \neq i}^n v_{xi} v_{xj} + \\
& + \frac{B}{2} \sum_{i=1}^n \sum_{x=1}^n \sum_{y=1, y \neq x}^n v_{xi} v_{yi} + \\
& + \frac{C}{2} \left(\sum_{x=1}^n \sum_{i=1}^n v_{xi} - (n + \sigma) \right)^2 + \\
& + \frac{D}{2} \sum_{x=1}^n \sum_{y=1, y \neq x}^n \sum_{i=1}^n d_{xy} v_{xi} (v_{y,i+1} + v_{y,i-1}) + \\
& - E \sum_{x=1}^n \sum_{i=1}^n (v_{xi} - 1)
\end{aligned}$$

Współczynniki A , B , C , D są dodatnimi stałymi. Składnik mnożony przez A jest minimalny wtedy i tylko wtedy, gdy w każdym wierszu x macierzy V znajduje się nie więcej niż 1 element niezerowy. Podobnie, składnik mnożony przez B jest minimalny wtedy i tylko wtedy, gdy w każdej kolumnie i macierzy V znajduje się nie więcej niż 1 element niezerowy. σ oraz składnik mnożony przez E wymuszają, by liczba jedynek była równa n . Składnik mnożony przez D oblicza koszt drogi.

Wady takiego rozwiązania:

- utykanie w minimach lokalnych
- dobór współczynników

4.2.2 kodowanie miasto-miasto

W reprezentacji *miasto-miasto* zarówno wiersze jak i kolumny macierzy V indeksowane są numerami miast, a warunek $v_{ij} = 1$ oznacza, że odcinek z miasta i do miasta j jest częścią rozwiązania. Macierz V jest symetryczna oraz w każdym wierszu i każdej kolumnie znajdują się dokładnie 2 elementy równe 1.

Funkcja energii jest następująca:

$$E = \frac{A}{2} \sum_{x=1}^n \left(\sum_{i=1, i \neq x}^n v_{xi} - 2 \right)^2 +$$

$$\begin{aligned}
& + \frac{B}{2} \sum_{i=1}^n \left(\sum_{x=1, x \neq i}^n v_{xi} - 2 \right)^2 + \\
& + \frac{C}{2} \sum_{x=1}^n \sum_{i=1}^n v_{xi} d_{xi}
\end{aligned}$$

Współczynniki A , B , C , D są dodatnimi stałymi. Składnik mnożony przez A jest minimalny wtedy i tylko wtedy, gdy w każdym wierszu x macierzy V znajdują się 2 elementy niezerowe. Podobnie, składnik mnożony przez B jest minimalny wtedy i tylko wtedy, gdy w każdej kolumnie i macierzy V znajdują 2 element niezerowe. Ostatni składnik reprezentuje długość rozwiązania. Wady takiego rozwiązania:

- problem poddróg — kilka cykli
- słaba skalowalność

4.3 Symulowane Wyżarzanie

Skuteczność sieci Hopfielda można poprawić między innymi poprzez wzbogacenie jej o mechanizmy umożliwiające wychodzenie z minimów lokalnych.

Można zastosować algorytmy symulowanego wyżarzania (Simulated Annealing — SA).

- wprowadzenie do opisu sieci dodatkowego parametru nazywanego temperaturą, która reguluje częstotliwość i intensywność przejść sieci do konfiguracji zwiększających wartość energii
- w wysokich temperaturach przeszukiwana jest przestrzeń stanów sieci i możliwe jest wykonanie przejścia w jednym kroku pomiędzy stanami o bardzo różnych wartościach energii
- stopień akceptacji przez sieć potencjalnej zmiany stanu powodującej zwiększenie energii jest tym większy im mniejsza jest potencjalna zmiana energii spowodowana tą zmianą stanu
- stopniowo temperatura jest obniżana
- w miarę obniżania temperatury układ wybó konfiguracji zwiększającej wartość energii następuje coraz rzadziej.
- w temperaturze $T=0$ akceptowane są jedynie przejścia do stanów o niższej wartości energetycznej