ESI: Perceptrony proste i liniowe

[Matlab_1.1] Matlab2015b i nowsze

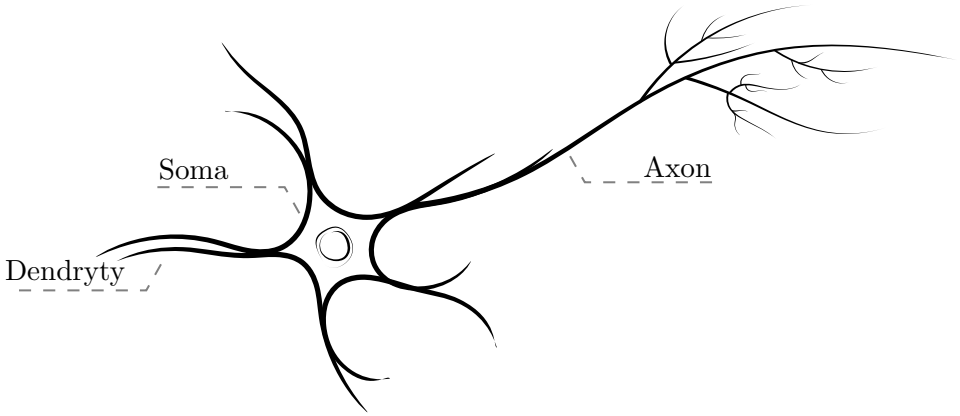
1 kwietnia 2019

1. **Cel ćwiczeń:** Celem ćwiczeń jest zapoznanie się studentów z podstawami zagadnieniami z zakresu sztucznych sieci neuronowych.

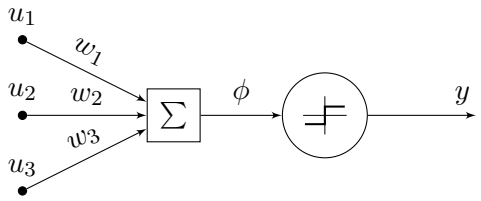
2. **Model neuronu McCullocha–Pittsa:** W roku 1943 McCulloch i Pitts zaproponowali sztuczny model neuronu [6], który w prosty sposób miał odtwarzać działanie neuronu biologicznego. Budowa biologiczna neuronu została przedstawiona na Rys. 1. Struktura neuronu może być podzielona na trzy funkcjonalne części: dendryty, somę oraz axon. Dendryty odpowiadają za zbieranie sygnałów z innych neuronów, soma odpowiedzialna jest za powstawanie nowego sygnału elektrochemicznego. Axon natomiast przekazuje sygnał do kolejnych neuronów. Kierując się tym tokiem rozumowania McCulloch i Pits zaproponowali tzw TLU (ang. Treshold Logic Unit), którego matematyczny opis przedstawia się następująco

$$y = \begin{cases} 1, & \sum_{i=1}^{n} u_i w_i > 0 \\ 0, & \sum_{i=1}^{n} u_i w_i \leqslant 0 \end{cases} \tag{1}$$

   a schemat przedstawia Rys. 2



Rysunek 1: Bilogiczny neuron

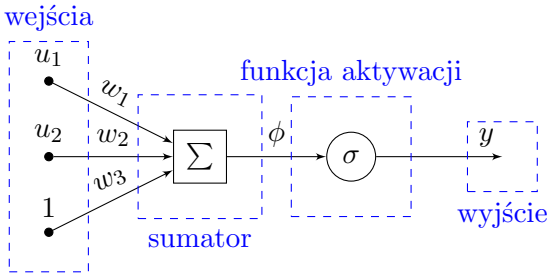


Rysunek 2: Perceptron McCullocha–Pittsa

3. **Ogólny model neuronu:** Model perceptronu McCullocha–Pittsa można uogólnić do postaci

$$y = \sigma\Big(\sum_{i=1}^{n} u_i w_i + w_{n+1}\Big), \tag{2}$$

gdzie n to liczba wejść, waga $w_{n+1} = b$ nazywana jest **biasem**, funkcja $\sigma(\cdot)$ nazywana jest **funkcją aktywacji** natomiast suma $\phi = \sum_{i=1}^{n} u_i w_i + w_{n+1}$ **potencjałem membranowym**. Od doboru funkcji aktywacji zależy zachowanie neuronu (tj. wartości wyjściowe) w odniesieniu do wartości potencjału membranowego. W konsekwencji od wybranej funkcji aktywacji zależą właściwości danej sieci neuronowej. Do popularnych funkcji aktywacji należą:



Rysunek 3: Ogólny model sztucznego neuronu

funkcja liniowa $y = ax + b$, funkcja różniczkowalna, gładka i monotoniczna

funkcja unipolarna skoku jednostkowego $y = \begin{cases} 0, & \phi \leqslant 0 \\ 1, & \phi > 0 \end{cases}$, funkcja przedziałami różniczkowalna, monotoniczna, zwana funkcją Heaviside'a

funkcja bipolarna skoku jednostkowego $y = \begin{cases} -1, & \phi \leqslant 0 \\ 1, & \phi > 0 \end{cases}$, funkcja przedziałami różniczkowalna, monotoniczna

funkcja sigmoidalna unipolarna $y = \frac{1}{1+e^{\beta\phi}}$, funkcja różniczkowalna, gładka i monotoniczna

funkcja sigmoidalna bipolarna $y = \frac{1-e^{\beta\phi}}{1+e^{\beta\phi}}$, funkcja różniczkowalna, gładka i monotoniczna

funkcja Gaussa $y = ae^{-\frac{(x-b)^2}{2c^2}}$, funkcja różniczkowalna, gładka, niemonotoniczna

W kontekście rodzajów sztucznych sieci neuronowych mówimy o sieciach liniowych, jeżeli funkcje aktywacji są funkcjami liniowymi lub o sieciach nieliniowych w przeciwnym wypadku.

4. **Cechy i właściwości sieci neuronowych:**

   uczenie – podstawową cechą sieci neuronowych jest możliwość uczenia. Istnieją dwie metody uczenia sieci neuronowych:

   - z nauczycielem – w tej metodzie wykorzystuje się zbiory uczące składające się z z par (p, t) w których p jest zadaną wartością wejściową, natomiast t jest podpowiadaną przez nauczyciela wartością wyjściową.
   - bez nauczyciela – w tej metodzie sieć uczy się samodzielnie jedynie na podstawie sygnałów wejściowych p, uczenie polega wówczas na odkrywaniu analogii i asocjacji w danych wejściowych.

   uogólnianie – sieci neuronowe posiadają zdolność uogólniania. Oznacza to, iż potrafi generalizować rozwiązywane problemy i dawać oczekiwaną odpowiedź nawet w przypadku gdy dane wejściowe nie były w zbiorze danych uczących,

   uniwersalny aproksymator – w teorii [1] sieć neuronowa może być uniwersalnym aproksymatorem. Oznacza to, iż sieć o skończonej liczbie neuronów może przybliżyć dowolną funkcję matematyczną z dowolną dokładnością. W praktyce jednak ograniczeniem jest "skończona liczba neuronów", która może być na tyle duża iż wystąpią problemy pamięciowe i numeryczne w procesie uczenia,

   odporność na zakłócenia – sieć neuronowa jest w stanie wygenerować oczekiwany wynik nawet dla niepewnych, uszkodzonych lub niepełnych danych wejściowych.

   pamięć – odpowiednia konstrukcja sieci (rekurencyjna) pozwala na uzyskanie efektu pamięci. Dzięki temu sieć może rozwiązywać problemy trwające w czasie (rozpoznawanie mowy), a w ogólności funkcjonować jak maszyna Touringa [7, 3]

5. **Uczenie sieci neuronowych:** Jak już wspomniano proces uczenia może odbywać się pod nadzorem nauczyciela (uczenie z nauczycielem, wówczas dane uczące zawierają zarówno próbki wejściowe jak i wyjściowe), lub bez nauczyciela (wówczas istotne jest odkrywanie reguł asocjacji przez sztuczną sieć neuronową).

   5.1. **Reguła Hebba:** oparta o o teorię Donalda Hebba [2] dotyczącą aktywacji neuronów oraz powstawania engramów. Hebb postulował, iż połączenie pomiędzy dwoma neuronami wchodzącymi w stan pobudzenia w wyniku tego samego sygnału jest wzmacniane. Odzwierciedleniem tego jest mechanizm uczenia w następującej postaci

$$w_i(k+1) = w_i(k) + \Delta w_i(k), \tag{3}$$

$$\Delta w_i(k) = \eta p_i(k) y(k), \tag{4}$$

   gdzie p to wartości wejściowe zbioru uczącego, t oczekiwane wartości wyjściowe a η jest **krokiem uczenia**. Krok uczenia to parametr pozwalający na zmianę tempa uczenia.

5.2. **Reguła perceptronowa:** Regułę tę stosuje się do uczenia perceptronu prostego ze skokową funkcją aktywacji. Korekcja wag odbywa się w/g wzoru

$$w_i(k+1) = \begin{cases} w_i(k), & y(k) = t(k), \\ w_i(k) + u_i(k), & y(k) < t(k), \\ w_i(k) - u_i(k), & y(k) > t(k), \end{cases} \tag{5}$$

gdzie p to wartości wejściowe zbioru uczącego, t oczekiwane wartości wyjściowe.

5.3. **Reguła Widrowa–Hoffa:** Regułę tę stosuje się, gdy ze względu na nieciągłość funkcji aktywacji nie można wyznaczyć pochodnej. Korekcja wag odbywa się w/g wzoru [4]

$$w_{i,j}(k+1) = w_{i,j}(k) + \Delta w_{i,j}(k), \tag{6}$$

$$\Delta w_{i,j}(k) = \eta p_j(k)\Big(t_i(k) - y_i(k)\Big), \tag{7}$$

gdzie p to wartości wejściowe zbioru uczącego, t oczekiwane wartości wyjściowe a η jest krokiem uczenia. Reguła Widrowa–Hoffa jest uogólnieniem reguły perceptronowej. Jednocześnie można zauważyć, iż reguła Widrowa–Hoffa minimalizuje kwadratową **funkcję celu**

$$E(k) = \sum_{i=1}^{n} (t_i(k) - y_i(k))^2. \tag{8}$$

Funkcja celu to funkcja która jest minimalizowana przez algorytm uczenia. Znalezienie minimum gwarantuje znalezienie najlepszego dopasowania parametrów (wag) sieci.

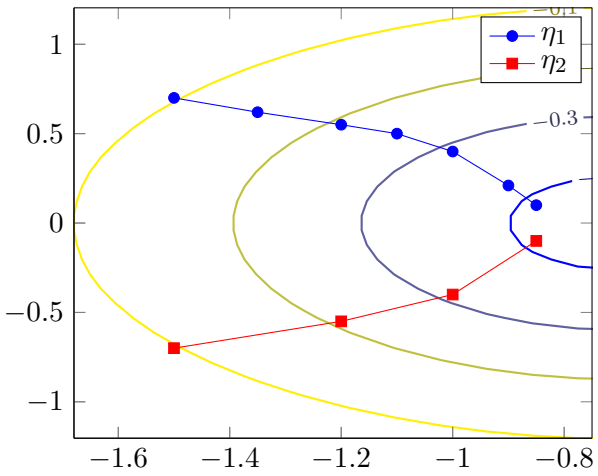
5.4. **Algorytmy gradientowe i wsteczna propagacja:** Przyjmując do rozważań, iż minimalizacja pewnej określonej funkcji zapewni minimalizację różnicy pomiędzy wartością oczekiwaną a otrzymaną $e_i = t_i - y_i$, można przedstawić problem uczenia sieci jako problem optymalizacyjny. Bezpośrednim następstwem takiego rozumowania jest **Reguła Delty** w której zmiana wag jest zdefiniowana następująco

$$\Delta w_{i,j}(k) = -\eta \frac{\partial E(k)}{\partial w}, \tag{9}$$

będąc uogólnieniem dla neuronów o ciągłej, gładkiej funkcji aktywacji. Krok uczenia η definiuje jak szybko algorytm "zbiega" wzdłuż kierunku spadku gradientu. Rys. 4 przedstawia zachowanie algorytmu dla $\eta_1 < \eta_2 < 1$, gdzie η_1 zaznaczono kolorem niebieskim, zaś η_2 czerwonym. Inną popularną optymalizacji nieliniowej stosowanej do uczenia sieci neuronowych jest **metoda największego spadku**. Jest to metoda gradientowa polegająca na wyznaczenie minimum zadanej funkcji celu, będąca rozszerzeniem reguły delty, η nie jest bowiem wybierane arbitralnie a optymalizowane w każdym kroku, tak aby wyznaczyć kierunek największego spadku.

$$W(k+1) = W(k) - \eta \nabla E(k), \tag{10}$$

gdzie W jest wektorem (lub macierzą) wag, a $\nabla E(k) = \left[\frac{\partial E(k)}{\partial w_1}, \frac{\partial E(k)}{\partial w_2}, \ldots, \frac{\partial E(k)}{\partial w_n}\right]$ jest gradientem



Rysunek 4: Algorytm największego spadku dla różnych η

funkcji celu.

5.5. **Uczenie adaptacyjne:** Uczenie adaptacyjne polega na inteligentnym dopasowaniu parametrów uczenia do kontekstu w którym uczenie następuje [5]. Techniki adaptacyjne mają na celu poprawić zbieżność (zarówno złożoność czasową, obliczeniową jak i osiągalność) algorytmów szukania minimum funkcji celu. Dopasowaniu może podlegać krok uczenia

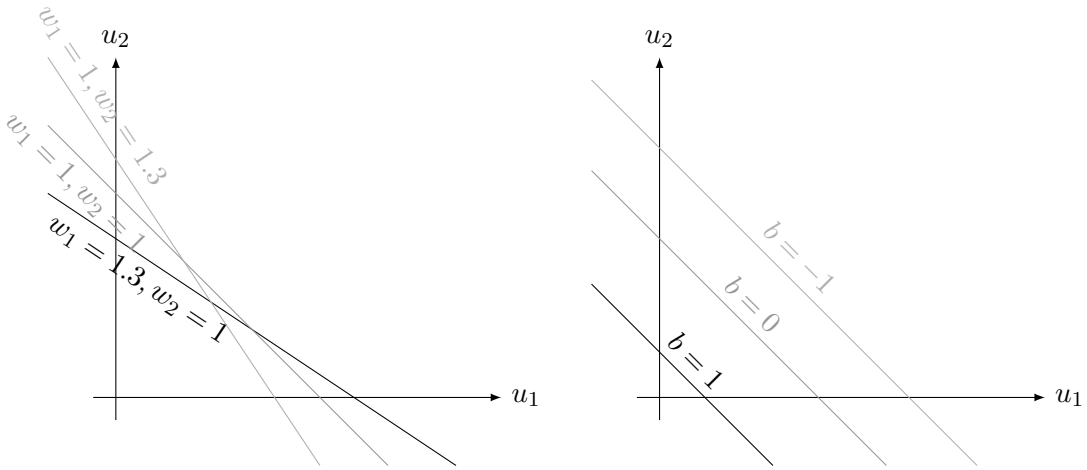
$$\eta(k) = \frac{E(k)}{\nabla E(k) d(k)}, \tag{11}$$

gdzie $d(k)$ jest kierunkiem przeszukiwań, a całość oparta jest o metodę Newtona. Istnieją również inne metody dopasowania kroku uczenia [5].

6. **Granica decyzyjna, bias:** Rozważmy model perceptronu w postaci

$$y = \begin{cases} 1, & \phi > 0 \\ 0, & \phi \leqslant 0, \end{cases} \tag{12}$$

gdzie $\phi = u_1 w_1 + u_2 w_2 + b$. Jest to model McCullocha–Pittsa o dwóch wejściach, jednym wyjściu i biasie. Położenie prostej ϕ w na płaszczyźnie zmiennych u_1, u_2 będzie zależne od wartości w_1, w_2, które będą współczynnikami kierunkowymi prostej oraz b, czyli biasu (wyrazu wolnego) Rys. 5a przedstawia położenie prostej ϕ w zależności od wartości wag w_1, w_2, natomiast Rys. 5b wpływ biasu. Załóżmy, iż chcemy zastosować perceptron (12) jako bramkę logiczną OR. Aby zastosować perceptron jako bramkę OR najpierw należy poddać go procesowi uczenia, np. metodą perceptronową. Tabela 1 przedstawia tablicę prawdy funkcji logicznej OR. Wartości u_1, u_2 stanowić będą wejściowy zbiór uczący a y oczekiwane wartości wyjściowe. Po nauczeniu perceptronu prosta $u_2 = -\frac{u_1 w_1 + b}{w_2}$ rozdzieli płaszczyzne zmiennych wejściowych na dwie półpłaszczyzny. Na jednej półpłaszczyźnie $u_1 w_1 + u_2 w_2 + b > 0$ będą leżały elementy przynależne $y = 1$, na drugiej $u_1 w_1 + u_2 w_2 + b \leqslant 0$ przynależne $y = 0$ (patrz (12)). Prostą $u_2 = -\frac{u_1 w_1 + b}{w_2}$ nazywamy **granicą decyzyjną**, a perceptron w takiej postaci



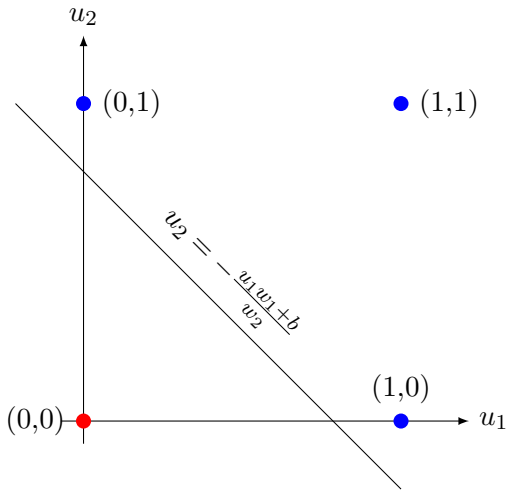
(a) Wpływ wag w_1, w_2 (b) Wpływ biasu b

Rysunek 5: Wpływ wag w_1, w_2, b na położenie prostej ϕ

u_1	u_2	y
0	0	0
0	1	1
1	1	1
1	0	1

Tabela 1: Tablica prawdy funkcji OR

jest klasyfikatorem binarnym.



Rysunek 6: Przykładowy przebieg granicy decyzyjnej

7. **Separowalność liniowa:** O problemie klasyfikacji można powiedzieć, iż jest **separowalny liniowo** jeżeli można wyznaczyć hiperpłaszczyznę rozdzielającą przestrzeń elementów na dwa podzbiory(patrz Rys. 6). Przykładami problemów separowalnych liniowo w przestrzeni dwuwymiarowej są problem

OR i AND.

8. **Zadania:**

   8.1. Korzystając z funkcji *ezplot*(Matlab 2017a i nowsze*fplot*) narysować wykresy funkcji liniowej oraz unipolarnego oraz bipolarnego skoku jednostkowego w przedziale domkniętym $\langle -1, 1\rangle$.

   8.2. Sprawdzić wartości funkcji *hardlim*, *hardlims* i *heaviside* w punktach $\{-1, 0, 1\}$.

   8.3. Sprawdzić działanie **biasu** na przykładzie funkcji liniowej ($y = ax + b$) i skoku jednostkowego ($hardlim(x + b)$). Narysować wykresy dla wszystkich $b = \{-1, -0.3, 0.4, 1\}$ wykorzystując dodatkowe parametry polecenia *plot*.

   8.4. Samodzielnie zaimplementować uczenie dwu-wejściowego perceptronu prostego z biasem z zastosowaniem reguły perceptronowej. Nauczyć perceptron rozwiązywania problemu OR.

   8.5. Wykorzystując polecenia *perceptron* i *train* sprawdzić działanie algorytmu z zad 4.

   8.6. Samodzielnie zaimplementować uczenie jedno-wejściowego perceptronu liniowego z biasem z zastosowaniem reguły Widrowa–Hoffa.

   8.7. Wykorzystując polecenia *newlind* i *train* nauczyć liniową sieć neuronową wyznaczania n-tej liczby nieparzystej. Porównać wyniki dla algorytmu z zad 6.

   8.8. Napisać skrypt który będzie wyświetlał granicę decyzyjną perceptronu prostego. Sprawdzić działanie skryptu dla problemów OR, AND i XOR dla pojedynczego neuronu uczonego algorytmem z zad. 4.

Literatura

[1] M H Hassoun. *Fundamentals of Artificial Neural Networks*. A Bradford book. MIT Press, 1995.

[2] D Hebb. *Organization of Behaviour*. J. Wiley, New York, 1949.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[4] Stefen Hui and Stanislaw H. Zak. The Widrow-Hoff Algorithm for McCulloch-Pitts Type Neurons. *IEEE Transactions on Neural Networks*, 5(6):924–929, 1994.

[5] GEORGE D. MAGOULAS and MICHAEL N. VRAHATIS. Adaptive Algorithms for Neural Network Supervised Learning: a Deterministic Optimization Approach. *International Journal of Bifurcation and Chaos*, 16(07):1929–1950, 2006.

[6] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[7] Alan Turing. Intelligent machinery: a report. *London: National Physical Laboratory*, 1948.