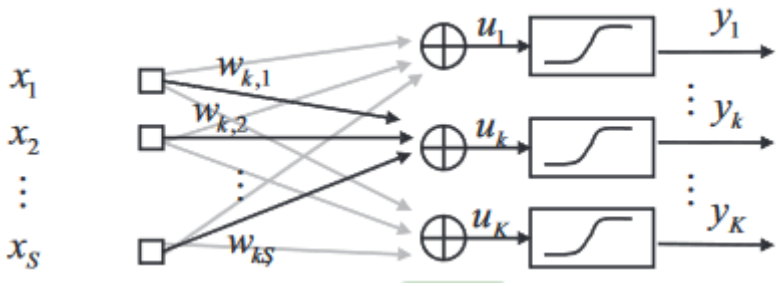
PODSTAWY SZTUCZNEJ INTELIGENCJI

Scenariusz 2
Maria Podkalicka, IS grupa 3

Temat ćwiczenia: Budowa i działanie sieci jednowarstwowej.

Sieć jednowarstwowa to zespół kilku neuronów, przetwarzających sygnały z tych samych wejść. Sieć taką, złożoną z K neuronów, przedstawia poniższy rysunek.



Zaznaczono na nim neuron o indeksie k, aby pokazać powszechnie stosowane oznaczenia. Pobudzenie k-tego neuronu w warstwie będziemy oznaczać przez uk, zaś jego wyjście przez yk. Natomiast wagi tego neuronu będziemy indeksować podając najpierw numer neuronu, do którego dana waga należy, a następnie numer wejścia, do którego ona prowadzi. Tak więc np.w3,2 oznacza wagę neuronu 3 do wejścia 2.

Reguła delty i metoda największego spadku

Rozważmy sieć jednowarstwową o liniowych elementach przetwarzających. Załóżmy, że mamy *P*-elementowy zbiór wzorców. Przy prezentacji m-tego wzorca na wejściu sieci, dla *j*-tego elementu wyjściowego możemy zdefiniować błąd:

$$\delta_j^{\mu} = y_j^{z\mu} - y_j^{\mu} = y_j^{z\mu} - \varphi_j^{\mu} = y_j^{z\mu} - \sum_{i=0}^{n_0} w_{ji} u_i^{\mu}$$

gdzie y_j^{zm}, y_j^m i j_j^m oznaczają odpowiednio oczekiwane i aktualne wartości wyjścia *j*-tego elementu oraz ważoną sumę wejść wyznaczoną w jego sumatorze przy prezentacji m-tego wzorca; u_i^m - *i*ta składowa m-tego wektora wejściowego ($u_0^m = 1$, wejście progowe); w_{ij} oznacza wagę połączenia pomiędzy *j*-tym elementem warstwy wyjściowej a *i*-tym elementem warstwy wejściowej; n_0 - liczba wejść.

Jako miarę błędu sieci x wprowadźmy sumę po wszystkich wzorcach błędów powstałych przy prezentacji każdego z nich:

$$\xi = \sum_{\mu=1}^{P} \xi_{\mu} = \frac{1}{2} \sum_{\mu=1}^{P} \sum_{j=1}^{n} (y_j^{z\mu} - \varphi_j^{\mu})^2$$

gdzie:

$$\xi_\mu = \frac{1}{2}\sum_{j=1}^{n}(\delta_j^\mu)^2$$

przy czym *n* oznacza liczbę elementów w warstwie wyjściowej.

Problem uczenia sieci to zagadnienie minimalizacji funkcji błędu x. Jedną z najprostszych metod minimalizacji jest gradientowa metoda największego spadku. Jest to metoda iteracyjna, która poszukuje lepszego punktu w kierunku przeciwnym do gradientu funkcji celu w danym punkcie. Stosując powyższą metodę do uczenia sieci, zmiana Dw_{ji} wagi połączenia w_{ji} winna spełniać relację:

$$\Delta w_{ji} = -\eta\frac{\partial\xi}{\partial w_{ji}} = -\eta\sum_{\mu=1}^{P}\frac{\partial\xi_\mu}{\partial w_{ji}} = -\eta\sum_{\mu=1}^{P}\frac{\partial\xi_\mu}{\partial y_j^\mu}\frac{\partial y_j^\mu}{\partial w_{ji}}$$

gdzie h oznacza współczynnik proporcjonalności.

W przypadku liniowych elementów przetwarzających mamy:

$$\frac{\partial\xi_\mu}{\partial y_j^\mu} = -(y_j^{z\mu} - y_j^\mu) = -\delta_j^\mu$$

$$\frac{\partial y_j^\mu}{\partial w_{ji}} = \frac{\partial\varphi_j^\mu}{\partial w_{ji}} = u_i^\mu$$

Stąd otrzymamy:

$$\Delta w_{ji} = \eta\sum_{\mu=1}^{P}\delta_j^\mu u_i^\mu$$

Ostatecznie pełną regułę wyznaczania wag połączeń zapiszemy:

$$w_{ji}^n = w_{ji}^s + \Delta w_{ji}$$

gdzie górne indeksy *n* i *s* oznaczają odpowiednio "nową" i "starą" wartość współczynnika wag w_{ji}.

Konsekwentna realizacja metody największego spadku wymaga dokonywania zmian wag w_{ji} dopiero po zaprezentowaniu sieci pełnego zbioru wzorców, m = 1, 2, ..., *P*. W praktyce stosuje się jednak zmiany wag po każdej prezentacji pojedynczego wzorca zgodnie ze wzorem:

$$\Delta^{\mu} w_{ji} = -\eta \frac{\partial \xi_{\mu}}{\partial w_{ji}} = \eta \delta_j^{\mu} u_i^{\mu}$$

co pozwala istotnie uprościć praktyczną realizację algorytmu.

Zasada zmiany wartości współczynników wag w_{ji}, określona powyższym wzorem, nazywana jest regułą delty. Owa modyfikacja wprowadza wprawdzie pewne zaburzenie do metody największego spadku, ale jest ono zaniedbywalnie małe dla odpowiednio niewielkich wartości h, czyli dla małych zmian wag. Reguła delty, jako pewne przybliżenie metody największego spadku przy wystarczająco małym współczynniku h, poszukuje zbioru wag minimalizującego funkcję błędu sieci liniowej.

Ogólny schemat procesu trenowania sieci:

1. W raz z każdym wektorem wejściowym X do neuronu podawany jest sygnał z. Neuron odpowiada na sygnał X sygnałem $y = W * X$
2. J eśli neuron nie jest nauczony, sygnał ten jest inny niż wymagany ($y \neq z$). Wewnątrz neuronu istnieje blok oceniający wielkość błędu, $\delta = z - y$
3. Blok ten składa się inwertora oraz sumatora. Na podstawie sygnału błedu oraz wektora wejściowego X możliwe jest takie skorygowanie wektora wag W, by neuron lepiej realizował zadaną funkcję y= f(X).
4. Nowy wektor wag W' obliczany jest ze wzoru:

$W' = W + \eta \delta X$, gdzie η jest współczynnikiem liczbowym, decydującym o szybkości nauki.

Reguła propagacji wstecznej

Algorytm wstecznej propagacji błędu jest uogólnieniem reguły delta.

Ogólnie algorytm propagacji wstecznej wymaga dla każdego wzorca uczącego wykonania następujących działań:

1. Podaj wektor uczący $\mathbf{u}^m$ na wejście sieci.
2. Wyznacz wartości wyjść u_j^{mm} każdego elementu dla kolejnych warstw przetwarzania neuronowego, od pierwszej warstwy ukrytej do warstwy wyjściowej.
3. Oblicz wartości błędów d_j^{Mm} dla warstwy wyjściowej.
4. Oblicz wartość sumy kwadratów błędów x_m.
5. Dokonaj propagacji wstecznej błędu wyjściowego d_j^{Mm} do poszczególnych elementów warstw ukrytych wyznaczając d_j^{mm} według wzoru:

$$\delta_j^{m\mu} = f'(\varphi_j^{m\mu}) \sum_{l=1}^{n_{m+1}} \delta_l^{(m+1)\mu} w_{lj}^{(m+1)}$$

6. Dokonaj aktualizacji wag kolejno pomiędzy warstwą wyjściową i ukrytą, a następnie pomiędzy warstwami ukrytymi przesuwając się w kierunku warstwy wejściowej.

Przygotowany przeze mnie zestaw uczący zawiera 10 dużych i 10 małych liter utworzonych na matrycy 5x5 czyli na 25 polach. Liniowym rozwinięciem tych matryc będą wektory wejściowe ciągu uczącego. Ciąg uczący składa się z 20 następujących wektorów uczących :

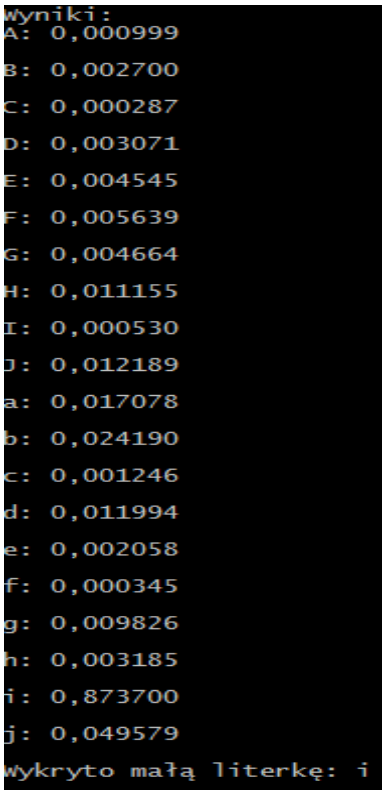
A= {1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1};
B= {1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0};
C= {1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1};
D= {1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0};
E= {1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1};
F= {1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0};
G= {1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0};
H= {1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1};
I= {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0};
J= {1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0};

a= {1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0};
b= {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0};
c= {0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0};
d= {0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0};
e= {0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0};
f= {0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0};
g= {1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0};
h= {0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0};
i= {0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0};
j= {0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0};

Przykładowa literka i która w programie została wyszukana:

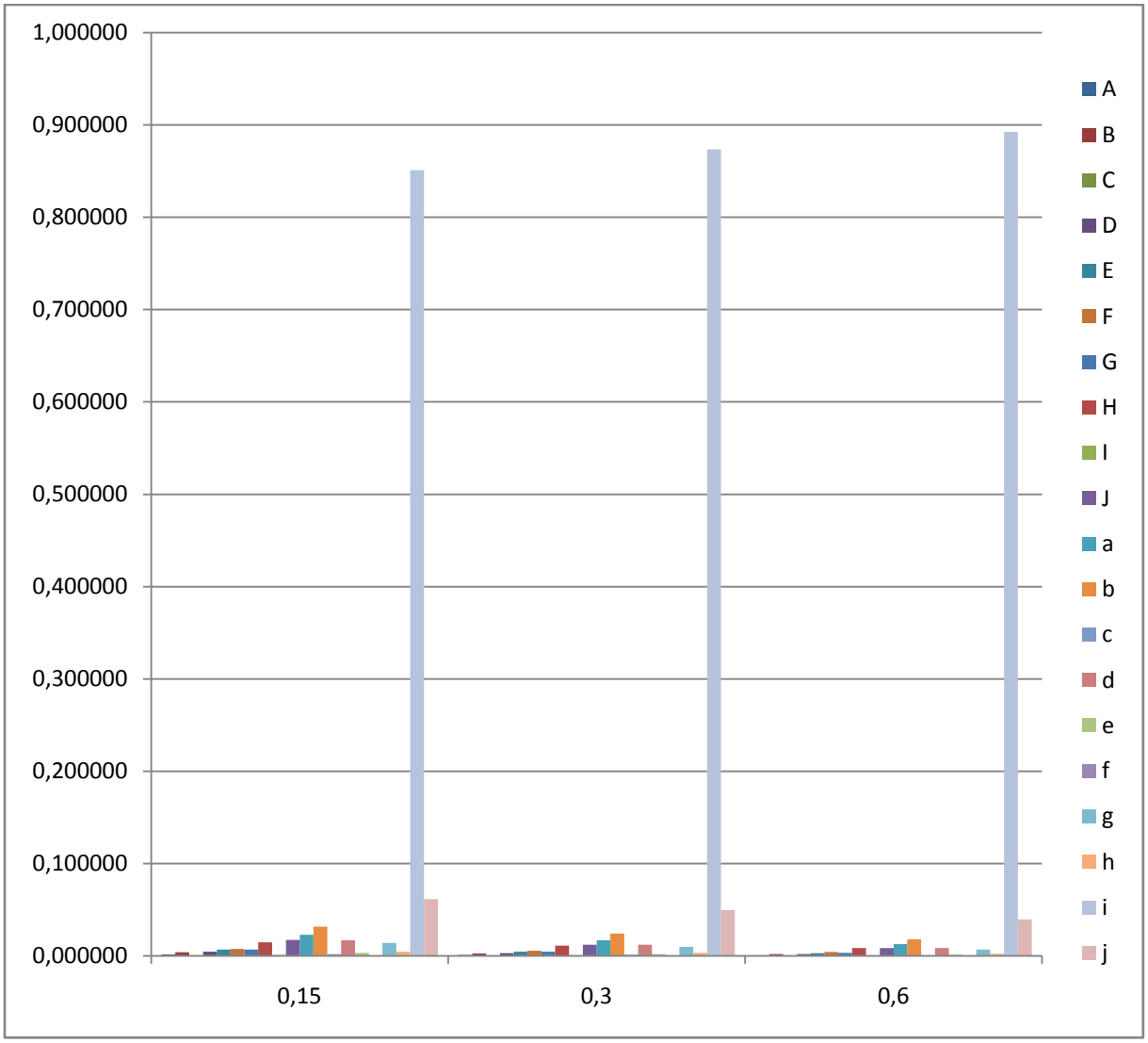
00000
00100
00000
00100
00100

Screen z wykonania testu poprawności działania dla współczynnika uczenia = 0,3 :



	0,15	0,3	0,6
A	0,001578	0,000999	0,000633
B	0,004004	0,002700	0,001823
C	0,000516	0,000287	0,000160
D	0,004540	0,003071	0,002014
E	0,006780	0,004545	0,003077
F	0,007672	0,005639	0,004111
G	0,006772	0,004664	0,003252
H	0,014569	0,011155	0,008566
I	0,000888	0,000530	0,000316
J	0,017416	0,012189	0,008561
a	0,022933	0,017078	0,012860
b	0,031780	0,024190	0,018111
c	0,002088	0,001246	0,000740
d	0,017146	0,011994	0,008549
e	0,003164	0,002058	0,001321
f	0,000604	0,000345	0,000196
g	0,014146	0,009826	0,006728
h	0,004481	0,003185	0,002298
i	0,850898	0,873700	0,892411
j	0,061266	0,049579	0,039632

W tabeli powyżej zaprezentowano otrzymane wyniki. W pierwszej kolumnie pokazane są litery, a w odpowiadających im rzędach wartości, jakie uzyskano dla odpowiednich współczynników uczenia tj. : 0.15, 0.30, 0.60.



Wykres przedstawia zależność pomiędzy wartościami uzyskanymi dla odpowiednich liter a wartościami współczynników uczenia.

Analiza wyników:

Widać, iż nauczona sieć bardzo poprawnie rozpoznaje prezentowane litery. Na podstawie wykresu można zauważyć, że nie biorąc pod uwagi wyniku dla małej literki i, spośród pozostałych wyróżnia się też wynik dla małej litery j. A więc na tej podstawie można zauważyć jakie cechy liter zostały uznane przez siec za istotne dla opisu litery, a ponieważ mała literka j ma najwięcej wspólnych cechy z małą literką i, dlatego j ma wyższy wynik od reszty liter.

Zwiększenie precyzji rozpoznawania liter a dokładniej - znaków i symboli opartych na matrycy prezentowanej sieci można uzyskać poprzez np. powiększenie jej rozmiarów.
Wybranie małego współczynnika uczenia doprowadza do bardzo wolnego procesu uczenia. Trzeba wykonać dużo kroków, ponieważ były one poprawiane o małe wartości.
Nie można także wybrać zbyt dużego współczynnika uczenia, gdyż może to prowadzić do bardzo gwałtownych zmian parametrów sieci.