Daniel Wysowski
Nr. albumu: 286136
Inżynieria Obliczeniowa
Wydział Inżynierii Metali i Informatyki Przemysłowej
Akademia Górniczo-Hutnicza w Krakowie

Sprawozdanie z przedmiotu: Podstawy sztucznej inteligencji

Scenariusz 2 – Budowa i działanie sieci jednowarstwowej

Spis treści:

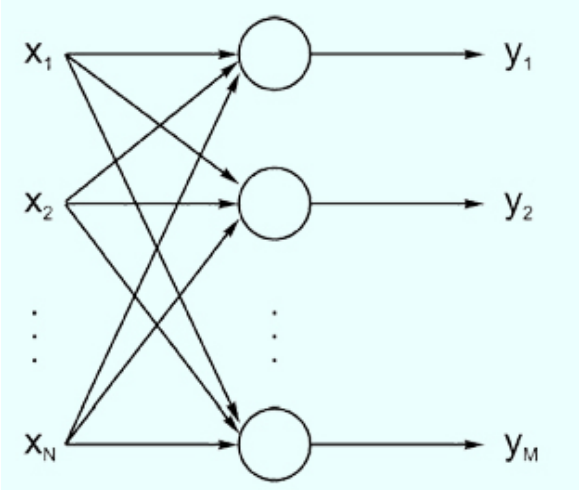
I.WSTĘP TEORETYCZNY

Aby móc mówić o sieci jednowarstwowej w ramach wstępu przytoczę kilka definicji:

Neuron: budowa i zasada działania tego modelu została oparta na swoim biologicznym odpowiedniku. Założeniem było tu zastąpienie neuronu jednostką binarną.

Sieć neuronowa: Zbiór neuronów, realizujących różne cele. W przypadku sztucznych sieci neuronowych jest to sztuczna struktura, zaprojektowana i zbudowana w taki sposób, aby modelowała działanie naturalnego układu nerwowego, w szczególności mózgu.

Sieci jednokierunkowe jednowarstwowe – w sieciach tego typu neurony ułożone są w jednej warstwie, która jest zasilana z węzłów wejściowych. Przepływ sygnału w tego typu sieciach przebiega zawsze w ściśle określonym kierunku: od warstwy wejściowej do warstwy wyjściowej. Na węzłach wchodzących nie znajdują się warstwy neuronów, gdyż nie zachodzi w nich żaden proces obliczeniowy. Dobór wag następuje tu w procesie uczenia sieci, czyli dopasowania sygnałów wyjściowych yi do wartości, której oczekujemy di .



II. CEL ĆWICZENIA

Celem zrealizowanego projektu jest poznanie budowy i działania jednowarstwowych sieci neuronowych oraz uczenie rozpoznawania wielkości liter.

Kroki, które należy wykonać:

a) Wygenerowanie danych uczących i testujących, zawierających 10 dużych i 10 małych liter dowolnie wybranego alfabetu w postaci dwuwymiarowej tablicy .
b) Przygotowanie dwóch jednowarstwowych sieci - każda według. innego algorytmu
c) Uczenie sieci dla przy różnych współczynnikach uczenia.
d) Testowanie sieci.

III. WYKONANIE ZADANIA + OPIS DZIAŁANIA

Najpierw wygenerowałem dane uczące i testujące, które będą zawierać po 10 dużych i 10 małych liter alfabetu.

Litery wybrane przeze mnie: A a D d E e F f N n P p R r T t U u Y y
Po wykorzystaniu danych uczących otrzymujemy liniowe rozwinięcie matryc powstałych po wykorzystaniu wspomnianych wcześniej danych uczących, reasumując przedstawione poniżej liczby to wektory wejściowe ciągu uczącego:

```
in=[0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
    1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0
    1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 1 0 0 0 0
    1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 0 0 0 0
    0 0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 1 0 1 0
    1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 1 1
    0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0
    0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    1 0 1 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 1 1
    1 1 1 0 1 0 1 1 1 1 1 1 1 0 0 0 1 0 0 1
    0 1 0 0 0 1 0 1 1 1 0 1 0 1 0 1 0 0 1 0
    0 1 0 0 0 1 0 1 0 1 0 1 0 1 1 1 0 0 0 0
    0 0 0 1 0 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1
    1 0 1 0 0 0 0 0 1 0 1 0 1 1 0 0 1 0 0 1
    1 0 1 0 1 1 1 0 1 1 1 1 1 0 0 0 1 1 0 0
    1 0 0 0 1 0 1 1 0 0 1 0 1 0 0 0 0 0 0 1
    1 0 0 0 1 0 1 0 1 0 1 0 1 1 1 1 0 0 1 1
    1 1 0 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 0 1
    1 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 1
    1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 0 1 1 0 1
    0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0
    0 1 0 1 0 1 0 0 0 0 0 1 1 1 1 1 0 0 1 0
    0 1 0 1 0 1 0 0 1 1 0 1 0 0 0 0 0 1 0 1
    1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0
    1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 0 1 1 0 0
    0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1
    0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0
    0 1 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1
    1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0
    1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 0 0 0 0 0
    0 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 1 1 0 0
    0 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 1 1
    0 1 1 1 1 1 1 0 0 0 1 0 0 0 0 0 1 1 0 0 0
    1 1 0 0 1 0 0 0 1 1 0 0 1 0 0 0 0 1 0 0]
  %A a D d E e F f N n P p R r T t U u Y y
```

Wyjściową (out) macierzą będą liczby 1 symbolizujący literę dużą lub 0 literę małą.

```
out=[ 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 ];
```

ANALIZA KODU:

KOD: net = newp(PR,S,TF,LF);

net = init(net);

PR - macierz o wymiarach Rx2, gdzie R jest liczbą wejść sieci - pierwsza kolumna tej macierzy zawiera minimalne wartości zaś druga maksymalne tych współrzędnych.
S - liczba neuronów sieci.
TF - nazwa funkcji aktywacji neuronów (zmienna tekstowa).
LF - nazwa funkcji trenowania sieci perceptronowej (zmienna tekstowa);

KOD: net=newlin(PR,S,ID,LF)

PR - identycznie jak w poprzednim.
S - liczba neuronów sieci, tj. wyjść sieci (neurony tworzą automatycznie warstwę wyjściową)
ID - wektor opóźnień poszczególnych elementów wektora wejść sieci.
LF - stała szybkości uczenia (treningu) sieci liniowej.

OSTATNIM KROKIEM WYKONANIA ĆWICZENIA BYŁO UCZENIE SIĘ SIECI DLA RÓŻNYCH WSPÓŁCZYNNIKÓW:

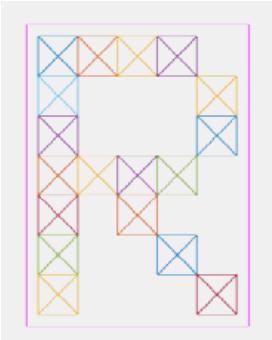
net.trainParam.epochs = 2500; - maksymalna liczba epok uczenia
net.trainParam.goal = 0.01; - oczekiwana końcowa wartość funkcji celu
net.trainParam.mu = 0.001; - współczynnik uczenia
Net=train(net, in, out);

Literka = testujacyR; - wywołanie litery przykładowej
Plotchar(literka); - wyświetleni jej

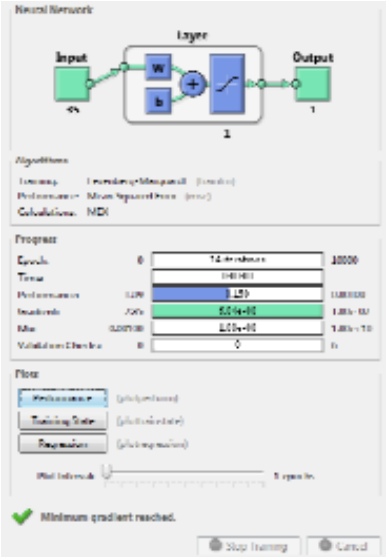
```
a=sim(net,literka);
If round(a) == 0
      disp (,mała litera');
Else
      disp(,duza litera');
End
Disp (a)
```

Literka- wybrana litera do tekstu
Plotchar - wyświetlanie
sim() - funkcja wyznaczająca wartość sieci neuronowej.
Round()- zaokrągla liczbę rzeczywistą

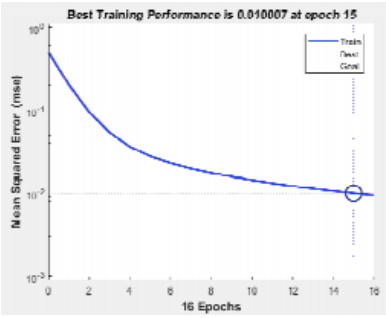
Wywołanie litery R.
Wyświetlenie litery R:



Otrzymuję „Neural Network Training”:



Plots -> performance



Dostaję wynik o wielkości literki oraz wynik dokładności:1.000

IV. PODSUMOWANIE

Ilość Iteracji (epochs) potrzebnych do nauki wynosi 0.5.
Program zwraca dokładnie wartość jeden lub zero, w zależności od wyniku (0 - mała, 1 - duża).

Program jest bardzo szybki, można to dostrzec po ilości iteracji które potrzebuje - zastosowanie innej metody może sprawić że czas uczenia bardzo drastycznie wzrośnie. Z tego wynika, że funkcja której użyliśmy (newp) jest szybsza (nie potrzebuje dużej ilości iteracji).