Podstawy Sztucznej Inteligencji

Scenariusz 2: **Budowa​​ i​​ działanie​​ sieci​​ jednowarstwowej**

Wykonał:

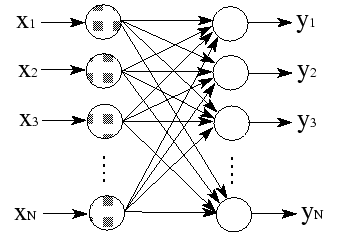
Bartłomiej Leja 254493

**1.Cel ćwiczenia**

Celem drugiego projektu było stworzenie na podstawie wcześniej zaimplementowanego perceptronu jednowarstwowej sieci neuronowej. Poznanie jej budowy i działania oraz nauczenie rozpoznawania liter.

**2.Sieć jednowarstwowa**

Sieć jednowarstwowa to najprostsza z sieci, polegająca na wykorzystaniu więcej niż jednego neuronu. Dane wejściowe przekazywane są do każdego neuronu należącego do sieci. Sieć taka zobrazowana jest na rysunku rys.1.



Rys.1

**3.Uczenie sieci jednowarstwowej**

Uczeni w sieci jednowarstwowej realizowane jest takim samym schematem jak uczenie perceptronu. Różnica polega na tym iż w przypadku sieci mamy do czynienia z wieloma neuronami. Każdy z neuronów dostaje na wejściu ten sam zestaw danych. Dodatkowo każdy neuron uczy się tylko jednej litery. W projekcie rozpoznawane jest 10 dużych liter oraz 10 małych z czego wynika iż sieć składa się z 20 neuronów.

**4.Uzyskane wyniki**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numer wariantu | Współczynnik uczenia η | Liczba danych uczących | Liczba potrzebnych iteracji do wyuczenia litery A | Liczba potrzebnych iteracji do wyuczenia litery B |
| 1 | 1 | 20 | 9 | 10 |
| 2 | 0,5 | 20 | 12 | 30 |
| 3 | 0,25 | 20 | 30 | 42 |
| 4 | 0,1 | 20 | 99 | 209 |

Tab. 1

Rys.2

Bardzo trudno przedstawić wyniki w postaci tabelarycznej ponieważ rozpoznajemy 20 liter z których każdą obsługuje inny neuron ze swoja liczbą iteracji. Jak widać w tabeli 1 liczba iteracji jest inna dla każdej litery w tym przypadku mniejsza dla litery A. Dodatkowo jak można było się spodziewać czym większy współczynnik uczenia tym niej iteracji. Dodatkowo zauważono, iż czym mniejszy współczynnik uczenia tym więcej razy algorytm rozpoznaje nawet zniekształconą literę A.

Literatura:

<http://wwwold.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node15.html>

**http://www.ai.c-labtech.net/sn/litery.html**