**Scenariusz 2**

Robert Białas, gr. 1

**Temat ćwiczenia:** Budowa i działanie sieci jednowarstwowej

**1. Cel ćwiczenia**

Celem ćwiczenia jest poznanie budowy i działanie jednowarstwowych sieci neuronowych orazuczenie rozpoznawania wielkości liter.

**2. Opis budowy sieci i algorytmów uczenia.**

Celem budowanej sieci jest rozpoznawanie wielkości liter. Do tego celu utworzony został zestaw liter (10 dużych i 10 małych), które są reprezentowane w postaci dwuwymiarowej tablicy 5x7 pikseli dla jednej litery.

Ćwiczenie zostało wykonane w języku C++ (własna implementacja), w którym zastosowano uczenie nadzorowane (z pomocą nauczyciela). Wykorzystano model sieci Perceptron oraz Adaline (ADAptive LInear NEuron) oraz model uczenia DeltaRule (sigmoidalna funkcja aktywacji).

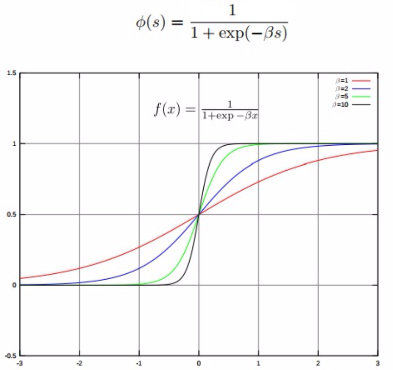
Duże litery wykorzystane w ćwiczeniu: ABCDEFGHIJ

Małe litery wykorzystane w ćwiczeniu: abcdefghij

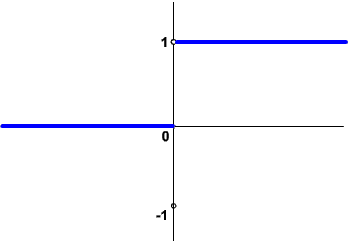
Tab. 1. Litery i ich reprezentacja

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 1 1 1 0  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  1 0 0 0 1 |  | 0 0 0 0 0  0 0 0 0 0  0 1 1 1 0  0 0 0 0 1  0 1 1 1 1  1 0 0 0 1  0 1 1 1 1 |
|  | 1 1 1 1 0  1 0 0 0 1  1 0 0 0 1  1 1 1 1 0  1 0 0 0 1  1 0 0 0 1  1 1 1 1 0 |  | 0 0 0 0 0  0 0 0 0 0  1 0 0 0 0  1 0 0 0 0  1 1 1 0 0  1 0 0 1 0  1 1 1 0 0 |
|  | 0 1 1 1 0  1 0 0 0 1  1 0 0 0 0  1 0 0 0 0  1 0 0 0 0  1 0 0 0 1  0 1 1 1 0 |  | 0 0 0 0 0  0 0 0 0 0  0 1 1 1 0  1 0 0 0 1  1 0 0 0 0  1 0 0 0 1  0 1 1 1 0 |
|  | 1 1 1 1 0  1 0 0 0 1  1 0 0 0 1  1 0 0 0 1  1 0 0 0 1  1 0 0 0 1  1 1 1 1 0 |  | 0 0 0 0 0  0 0 0 0 0  0 0 0 0 1  0 0 0 0 1  0 0 1 1 1  0 1 0 0 1  0 0 1 1 1 |
|  | 1 1 1 1 1  1 0 0 0 0  1 0 0 0 0  1 1 1 1 0  1 0 0 0 0  1 0 0 0 0  1 1 1 1 1 |  | 0 0 0 0 0  0 0 0 0 0  0 1 1 1 0  1 0 0 0 1  1 1 1 1 0  1 0 0 0 0  0 1 1 1 0 |
|  | 1 1 1 1 1  1 0 0 0 0  1 0 0 0 0  1 1 1 1 0  1 0 0 0 0  1 0 0 0 0  1 0 0 0 0 |  | 0 0 0 0 0  0 0 0 0 0  0 1 1 1 0  0 1 0 0 0  1 1 1 0 0  0 1 0 0 0  0 1 0 0 0 |
|  | 1 1 1 1 1  1 0 0 0 1  1 0 0 0 0  1 0 1 1 1  1 0 0 0 1  1 0 0 0 1  0 1 1 1 0 |  | 0 0 0 0 0  0 0 0 0 0  0 1 1 1 1  1 0 0 0 1  0 1 1 1 1  0 0 0 0 1  1 1 1 1 0 |
|  | 1 0 0 0 1  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  1 0 0 0 1 |  | 0 0 0 0 0  0 0 0 0 0  1 0 0 0 0  1 0 0 0 0  1 1 1 1 0  1 0 0 0 1  1 0 0 0 1 |
|  | 0 1 1 1 0  0 0 1 0 0  0 0 1 0 0  0 0 1 0 0  0 0 1 0 0  0 0 1 0 0  0 1 1 1 0 |  | 0 0 0 0 0  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  0 0 1 1 0 |
|  | 1 1 1 1 1  0 0 0 0 1  0 0 0 0 1  0 0 0 0 1  0 0 0 0 1  1 0 0 0 1  0 1 1 1 0 |  | 0 0 0 0 0  0 0 0 0 0  0 0 0 0 1  0 0 1 1 1  0 0 0 0 1  0 1 0 0 1  0 0 1 1 0 |

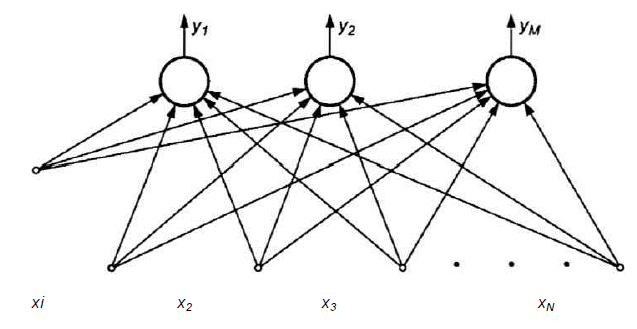
Ciąg przeznaczony do nauki w postaci pliku tekstowego data\_for\_learning.txt składa się z 20 wektorów (tablic jednowymiarowych) zawierających 35 pozycji oraz ciąg wyjściowy (zawierający 1 – duża litera, 0 – mała litera). Jako funkcję aktywacji sieci Perceptron oraz Adaline używa funkcji progowej unipolarnej, a sieć DeltaRule używa funkcji sigmoidalnej unipolarnej ze stałym współczynnikiem β.



Funkcja sigmoidalna unipolarna z różnymi współczynnikami β



Funkcja progowa unipolarna



Schemat sieci neuronowej jednowarstwowej

**2. Opis budowy sieci oraz wykorzystanych algorytmów**

Perceptron został zaimplementowany wg modelu McCullocha-Pittsa omówionego w książce „Sieci Neuronowe do przetwarzania informacji” Stanisława Osowskiego. Zastosowane jest uczenie z nauczycielem.

Dla każdego rodzaju sieci sygnały wejściowe xi (i = 1, 2, …, N) zawarte w pliku data\_for\_learning.txt są wczytywane w funkcji GetTestData().

W sieci Perceptron po rozpoczęciu uczenia funkcją Perceptron::Learn() sygnały xi są sumowane z odpowiednimi wagami wi  w sumatorze, tj. funkcji Perceptron::getResult() wg. wzoru:





Sygnał wyjściowy neuronu yj można opisać wzorem:

gdzie funkcje , czyli funkcję aktywacyjną wyrażamy wzorem:



Model neuronu McCullocha-Pittsa

Na koniec sprawdzany jest sygnał końcowy y przy pomocy funkcji Perceptron::getResult() – jeśli sygnał y nie zgadza się z podanymi wartościami oczekiwanymi, dochodzi do aktualizacji wag, a następnie nauczanie jest powtarzane.

Adaline został zaimplementowany poprzez modyfikację powyższego modelu.

Jak już zostało to wcześniej omówione, dane są wczytywane tą samą funkcją co Perceptron.

W sieci Adaline po rozpoczęciu uczenia funkcją Adaline::Learn() obliczana jest różnica między oczekiwanym wynikiem di a sumą sygnałów xi z odpowiednimi wagami wi , które są sumowane w sumatorze, tj. funkcji Adaline::getSum(). Różnica ta w kodzie oznaczona jest jako delta.

Następnie aktualizowane są wagi przy pomocy funkcji Adaline::changeWeights() w taki sam sposób jak w Perceptron, jednak nieco inaczej zapisany: w Perceptron różnica nie jest zapisywana jako delta przez co w changeWeights() jest ona obliczana bezpośrednio, a w Adaline różnica jest policzona przed użyciem changeWeights. Następnie obliczany jest błąd średniokwadratowy wg wzoru:



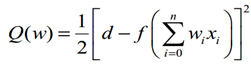
Uczenie trwa dopóki błąd średniokwadratowy jest większy od ustalonego błędu maksymalnego Emax.

Po zakończeniu uczenia sygnał wyjściowy yi , taki sam jak w Perceptron jest przekazywany funkcji aktywacyjnej Adaline::activationFunction() w celu ustalenia czy litera jest mała czy duża.

Został zaimplementowany jeszcze jeden rodzaj neuronu – Adaline z metodą nauczania Delta Rule.

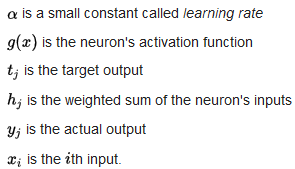
Różnica pierwsza w uczeniu polega na zastosowaniu innej funkcji aktywacji na sumie xi z wi podczas obliczania delty. Funkcja aktywacyjna to funkcja sigmoidalna unipolarna.

Zastosowanie funkcji aktywacji podczas liczenia delty powoduje drugą różnicę – inny wzór błędu średniokwadratowego:



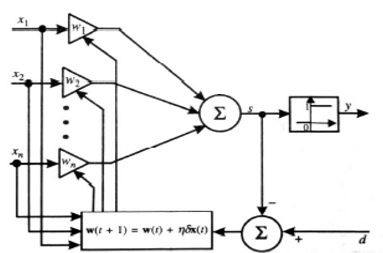
Trzecia różnica polega na zastosowaniu pochodnej funkcji aktywacji podczas aktualizacji wag – co zmienia wzór aktualizacji wag na:

, gdzie:



Warunek uczenia pozostaje taki sam – sieć uczy się, dopóki błąd średniokwadratowy jest większy od ustalonego EMax.

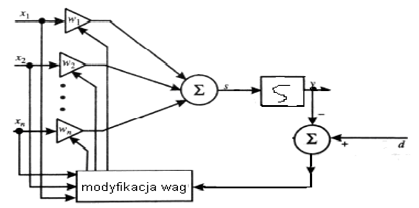
Schemat sieci Adaline



Model sieci Adaline

Model Adaline ma zbliżoną budowę do perceptronu. Różnią się one algorytmami uczenia. W modelu Adaline nie jest uwzględniana funkcja aktywacji przy porównywaniu sygnału wyjściowego z sygnałem wzorcowym.

Schemat sieci DeltaRule



Model sieci DeltaRule

Model DeltaRule ma analogiczną budowę do modelu Adaline, jednakże funkcją aktywacji jest funkcja sigmoidalna, a przy aktualizacji wag uwzględnia się pochodną tejże funkcji, obliczaną w funkcji derivativeActivationFunction().

**3. Zestawienie wyników**

*Przykładowe działanie programu*

Emax = 0.01

Współczynnik uczenia = 0.01

**Perceptron - wyniki przed uczeniem:**

**A - 0 1 1 1 0 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 1 0 0 0 1 1**

**B - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**C - 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**D - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**E - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1**

**F - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1**

**G - 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**H - 1 0 0 0 1 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 1 0 0 0 1 1**

**I - 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0 1**

**J - 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**a - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 1**

**b - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1**

**c - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0**

**d - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 1**

**e - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0 0**

**f - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0**

**g - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0**

**h - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1**

**i - 0 0 0 0 0 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 0 0 1 1 0 0**

**j - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0 1 0 0 1 1 0 1**

**Perceptron - wyniki po uczeniu:**

**Liczba epok: 88**

**A - 0 1 1 1 0 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 1 0 0 0 1 1**

**B - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**C - 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**D - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**E - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1**

**F - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1**

**G - 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**H - 1 0 0 0 1 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 1 0 0 0 1 1**

**I - 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0 1**

**J - 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**a - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0**

**b - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 0**

**c - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0**

**d - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0**

**e - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0 0**

**f - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0**

**g - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0**

**h - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 0**

**i - 0 0 0 0 0 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 0 0 1 1 0 0**

**j - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0 1 0 0 1 1 0 0**

**Adaline - wyniki przed uczeniem:**

**A - 0 1 1 1 0 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 1 0 0 0 1 1**

**B - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**C - 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**D - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**E - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1**

**F - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1**

**G - 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**H - 1 0 0 0 1 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 1 0 0 0 1 1**

**I - 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0 1**

**J - 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**a - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 1**

**b - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1**

**c - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**d - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 1**

**e - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0 1**

**f - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 1**

**g - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 1**

**h - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1**

**i - 0 0 0 0 0 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 0 0 1 1 0 1**

**j - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0 1 0 0 1 1 0 1**

**Adaline - wyniki po uczeniu:**

**Liczba epok: 34**

**A - 0 1 1 1 0 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 1 0 0 0 1 1**

**B - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**C - 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**D - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**E - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1**

**F - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1**

**G - 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**H - 1 0 0 0 1 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 1 0 0 0 1 1**

**I - 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0 1**

**J - 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**a - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0**

**b - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 0**

**c - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0**

**d - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0**

**e - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0 0**

**f - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0**

**g - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0**

**h - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 0**

**i - 0 0 0 0 0 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 0 0 1 1 0 0**

**j - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0 1 0 0 1 1 0 0**

**Delta Rule - wyniki przed uczeniem:**

**A - 0 1 1 1 0 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 1 0 0 0 1 1**

**B - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**C - 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**D - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**E - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1**

**F - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1**

**G - 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**H - 1 0 0 0 1 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 1 0 0 0 1 1**

**I - 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0 1**

**J - 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**a - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 1**

**b - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1**

**c - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**d - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 1**

**e - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0 1**

**f - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 1**

**g - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 1**

**h - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1**

**i - 0 0 0 0 0 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 0 0 1 1 0 1**

**j - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0 1 0 0 1 1 0 1**

**Delta Rule - wyniki po uczeniu:**

**Liczba epok: 1240**

**A - 0 1 1 1 0 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 1 0 0 0 1 1**

**B - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**C - 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1**

**D - 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1**

**E - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1**

**F - 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1**

**G - 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**H - 1 0 0 0 1 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 1 0 0 0 1 1**

**I - 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0 1**

**J - 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1**

**a - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0**

**b - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 0**

**c - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0**

**d - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0**

**e - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0 0**

**f - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0**

**g - 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0**

**h - 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 0**

**i - 0 0 0 0 0 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 0 0 1 1 0 0**

**j - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0 1 0 0 1 1 0 0**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Współczynniki uczenia | | | | | | | | | |
|  | 0.0001 | 0.001 | 0.005 | 0.01 | 0.025 | 0.05 | 0.1 | 0.25 | 0.5 | 0.75 |
| Perceptron – l. epok | 9164 | 1137 | 220 | 89 | 38 | 19 | 12 | 4 | 2 | 1 |
| Adaline – l.  epok | 4611 | 463 | 94 | 48 | 21 | 13 | 11 | 101 | 36 | 26 |
| Delta Rule – l.  epok | 40379 | 4039 | 809 | 405 | 163 | 82 | 42 | 17 | 9 | 7 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stały współczynnik uczenia h = 0.01 | | | | | |
| Współczynnik β | 0.15 | 0.25 | 0.5 | 0.75 | 1.0 |
| Liczba epok | 20360 | 11782 | 1581 | 656 | 405 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Współczynnik  Uczenia | 0.0001 | 0.001 | 0.005 | 0.01 | 0.025 | 0.05 | 0.1 | 0.25 | 0.5 | 0.75 |
| MSE Adaline | 1.499 | 1.495 | 1.489 | 1.431 | 1.452 | 1.421 | 1.028 | 0.813 | 0.135 | 0.048 |
| MSE DeltaRule | 1.499 | 1.498 | 1.494 | 1.483 | 1.465 | 1.443 | 1.429 | 1.394 | 1.293 | 1.171 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Emax | 1.0 | 0.75 | 0.5 | 0.25 | 0.1 | 0.01 | 0.001 | 0.0001 |
| Błędne litery Adaline | 3 | 2 | 2 | 1 | 0 | 0 | 0 | 0 |
| Błędne litery DeltaRule | 5 | 4 | 2 | 2 | 1 | 0 |  | 0 |

**4. Analiza**

Proces uczenia sieci jednowarstwowej przebiega analogicznie do uczenia perceptronu. Analizując wykres można zauważyć, że im wyższy współczynnik uczenia, tym liczba potrzebnych epok była niższa. Dla współczynnika uczenia > 0.01 sieć została wyćwiczona już po kilkunastu epokach. Dla współczynnika uczenia = 0.0001 proces uczenia zajął ponad 40000 epok dla metody DeltaRule.

W przypadku metody Adaline sieć dla bardzo małych współczynników uczenia nie potrzebowała aż tak dużej liczby, lecz ich ilość była zauważalna.

Dla współczynnika uczenia = 0.01 liczba epok uczenia praktycznie się wyrównała, a od 0.25 w górę liczba epok rosła dla Adaline, a zmniejszała się dalej dla DeltaRule.

Analizując powyższy wykres można zauważyć, że na efektywność uczenia sieci ma wpływ funkcja aktywacji. Im mniejsza wartość współczynnika β, tym potrzeba więcej epok nauki. Akceptowalne wartości współczynnika β wynoszą powyżej 0.5, gdzie uczenie przebiega w miarę na tym samym poziomie.

Z powyższego wykresu można odczytać, że współczynnik uczenia silnie wpływa na błąd średniokwadratowy tego uczenia. Im wyższy współczynnik uczenia, tym powstały błąd jest mniejszy.

Dla mniejszych współczynników uczenia błąd jest większy, co może wpływać na samo uczenie się sieci (nauka może trwać dłużej właśnie ze względu na powstały błąd).

Dla przypadku z DeltaRule błąd uczenia nie zmniejsza się równie szybko jak przy modelu Adaline.

Z powyższego wykresu wynika, że sieć wykorzystująca model Adaline daje poprawniejsze wyniki aniżeli sieć z modelem DeltaRule. Model Adaline miał zawsze mniej lub taką samą ilość błędów co model DeltaRule.

**5. Wnioski**

* Skuteczność procesu uczenia zależy od współczynnika uczenia. Wraz z jego wzrostem proces uczenia jest poprawniejszy. Jest to wytłumaczalne z tego względu, że im ta wartość jest większa tym przyrost wag, które na samym początku są niewielkie jest szybszy, więc proces uczenia przebiega szybciej.
* Sieć jednowarstwowa jest odporna na zaszumienie, jednakże tylko dla kilku bitów. Zbyt duże zaszumienie powoduje błędne odpowiedzi dawane przez sieć (w szczególności dla modelu DeltaRule).
* Przy projektowaniu sieci neuronowej trzeba wybrać odpowiedni jej model. Oprócz samego modelu na jej efektywność ma wpływ sama jej struktura, np. zastosowana funkcja aktywacji.
* Model Adaline pozwala na o wiele sprawniejsze uczenie sieci w porównaniu do modelu DeltaRule ze względu na mniejszą ilość epok wymaganych do nauczenia w większości przypadków oraz mniejsze błędy.
* Model Adaline pozwala na o wiele sprawniejsze uczenie sieci w porównaniu do modelu Perceptron ze względu na mniejszą ilość epok wymaganych do nauczenia w większości przypadków.
* W samym modelu DeltaRule wpływ na efektywność uczenia sieci ma współczynnik β w sigmoidalnej funkcji aktywacji (akceptowalną wartością współczynnika jest ten >=0.5, który powoduje, że sieć zachowuje się stabilnie i jej uczenie jest w miarę akceptowalne. Dla naszego przypadku wartość współczynnika, wynosząca 0.5, była wystarczająca, lecz dla bardziej zaawansowanych sieci współczynnik ten musi być wyższy.