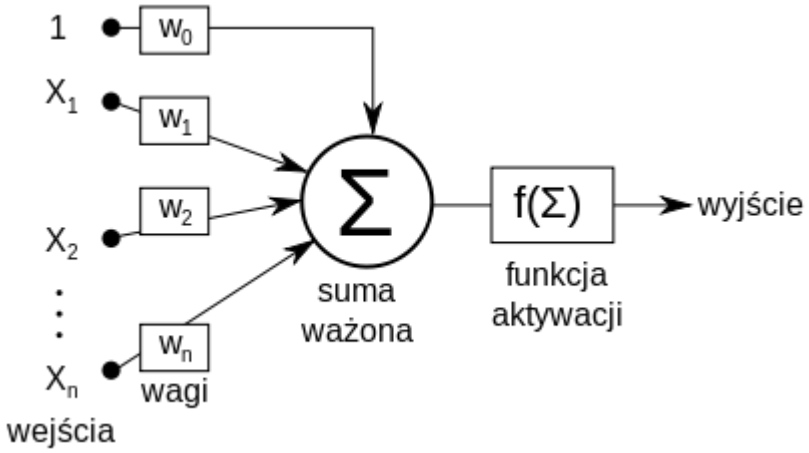
Tomasz Tomala
Podstawy Sztucznej Inteligencji
Sprawozdanie z projektu nr 2

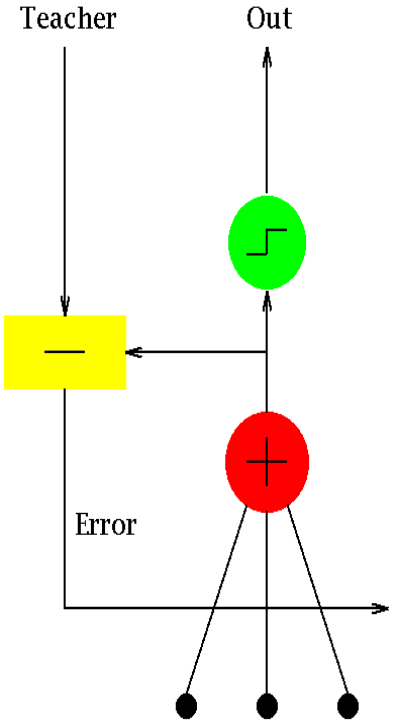
Celem ćwiczenia buło poznanie budowy i działania jednowarstwowych sieci neuronowych oraz uczenie rozpoznawania wielkości liter.

**1) Syntetyczny opis budowy wykorzystanego algorytmu uczenia:**

Aby wykonać ćwiczenie stworzyłem dwie jednowarstwowe sieci – każdą z wykorzystaniem innego algorytmu. Pierwsza z nich korzysta z modelu perceptronu McCullocha-Pittsa, druga z kolei wykorzystuje neuron typu Adaline.

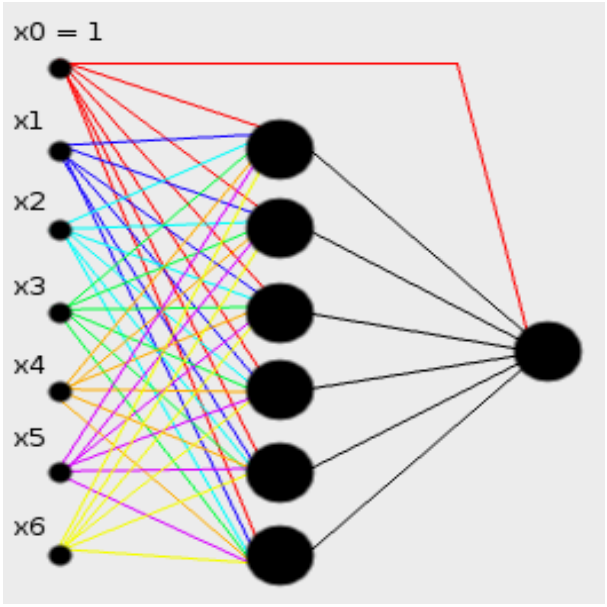



*Ilustracja 1: Model perceptronu McCullocha-Pittsa*




*Ilustracja 2: Model neuronu Adaline*

Obie sieci składają się z siedmiu neuronów. Sześć pierwszych stanowi pierwszą warstwę i przesyła sygnały wyjściowe do siódmego wyjściowego neuronu. Każdy z neuronów otrzymuje po siedem sygnałów wejściowych. Sposób w jaki wszystko zostało połączone przedstawiam na poniższej grafice:




*Ilustracja 3: Graficzne zilustrowanie połaczeń w sieci*

Do nauki perceptronów oraz adaline wykorzystałem algorytm Widrow-Hoffa.

Do budowy perceptronu wykorzystałem podany na wykładzie model McCullocha-Pittsa. Zaimplementowana przeze mnie klasa Perceptron składa się z trzech metod: active, process oraz learn.

Metoda ***active*** wykorzystuje unipolarną funkcję progową która zwraca wynik 0 lub 1 dla podanego argumentu.

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

Metoda ***process*** sumuje iloczyn sygnałów wejściowych i odpowiadających im wag. Uruchamia metodę activate z otrzymaną sumą iloczynów jako parametrem, oraz zwraca wynik tej metody.

$$\sum_{i=1}^{m} w_i x_i$$

Metoda ***learn*** wywołuje metodę process dla otrzymanych wejść jako parametrów, po czym na podstawie otrzymanego wyniku modyfikuje wszystkie wagi dla odpowiednich wejść.

$$\mathbf{W_i = W_i + ( y - y' ) * x_i * \alpha}$$

Budowa adaline jest bardzo podobna do budowy perceptronu. Różni się tylko tym, że modyfikowanie wag wykonuje się z pominięciem funkcji aktywacji. Zaimplementowana przeze mnie klasa Adaline składa się z czterech metod: active, process, learn oraz test.

Metoda ***active*** wykorzystuje unipolarną funkcję progową która zwraca wynik -1 lub 1 dla podanego argumentu.

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ -1 & \text{otherwise} \end{cases}$$

Metoda ***process*** sumuje iloczyn sygnałów wejściowych i odpowiadających im wag, oraz zwraca ten wynik.

$$\sum_{i=1}^{m} w_i x_i$$

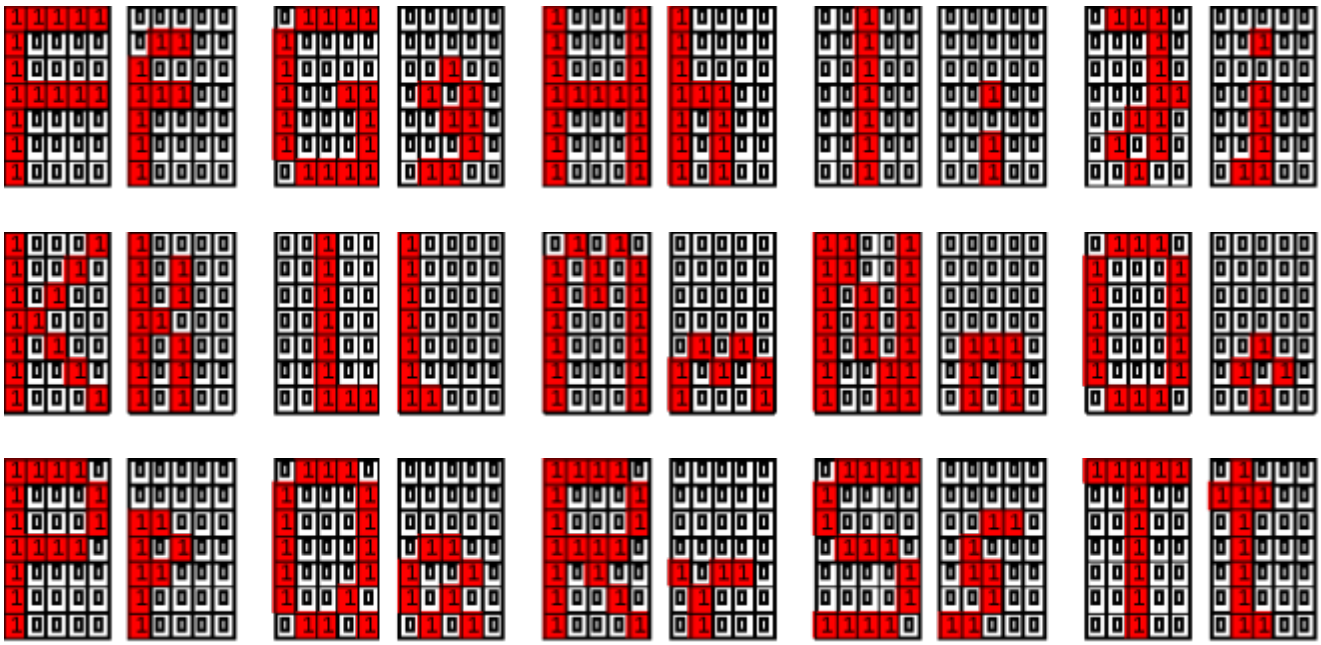
Metoda ***learn*** wywołuje metodę process dla otrzymanych wejść jako parametrów, po czym na podstawie otrzymanego wyniku modyfikuje wszystkie wagi dla odpowiednich wejść.

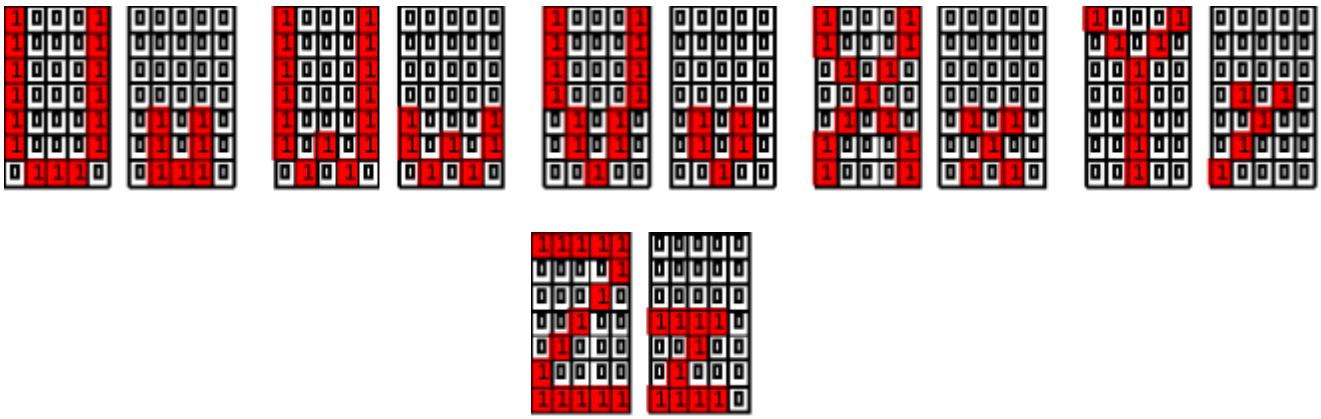
$$\mathbf{W_i = W_i + ( y - y' ) * x_i * \alpha}$$

Metoda ***test*** uruchamia metodę active z wynikiem metody process jako parametrem.

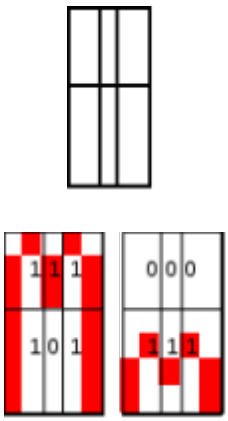
**2) Zestawienie otrzymanych wyników:**

Jako dane uczące i testujące wykorzystałem własnoręcznie stworzone litery alfabetu. Są one przedstawione jako dwuwymiarowa tablica o rozmiarach 7 x 5. Przedstawiam je poniżej:

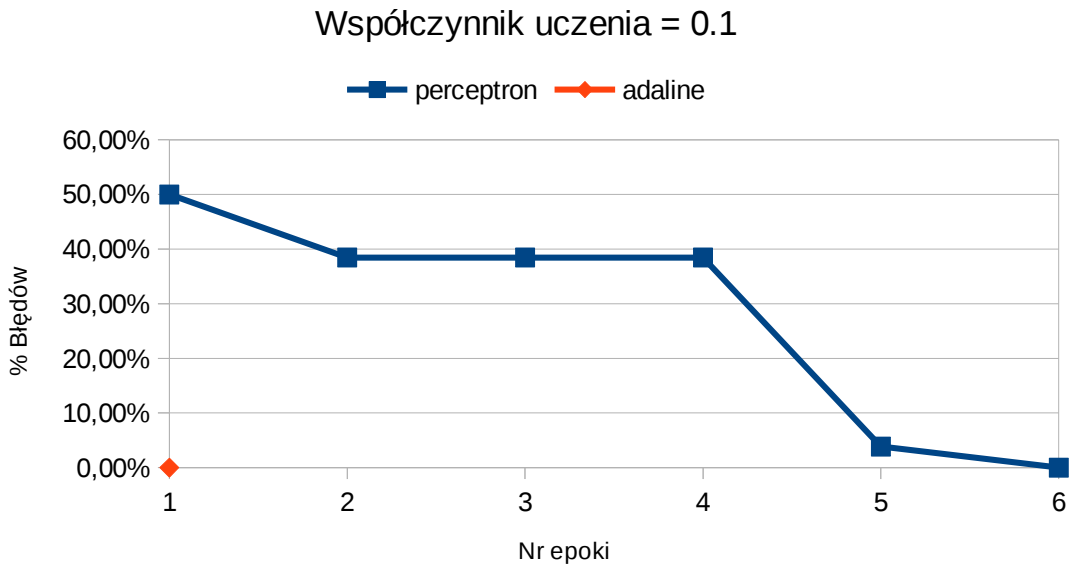


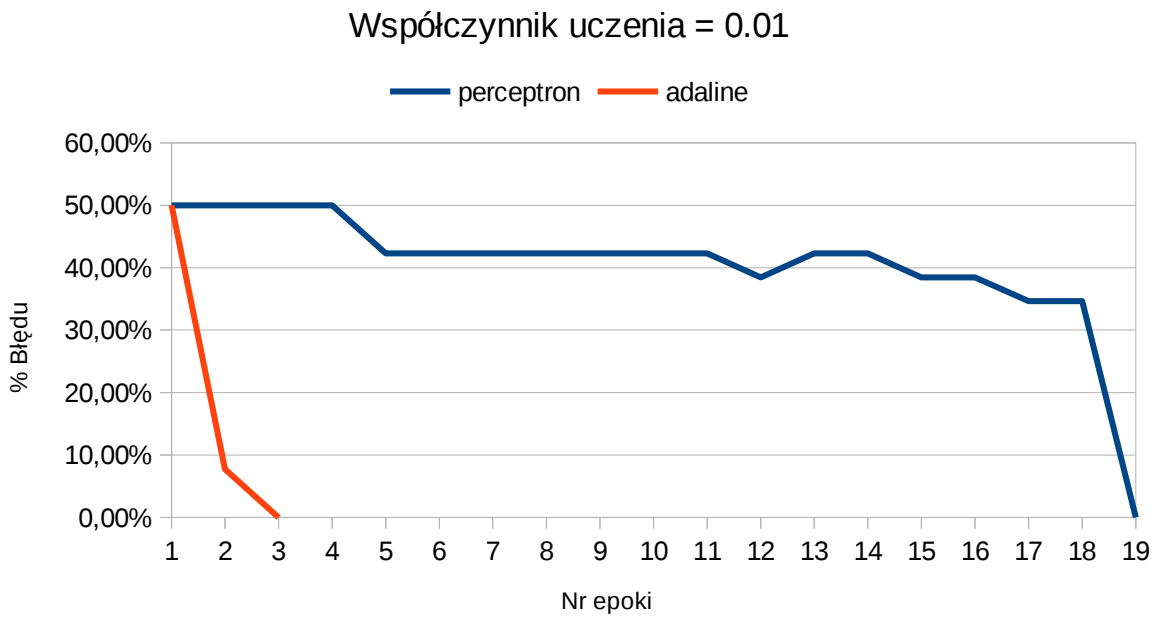


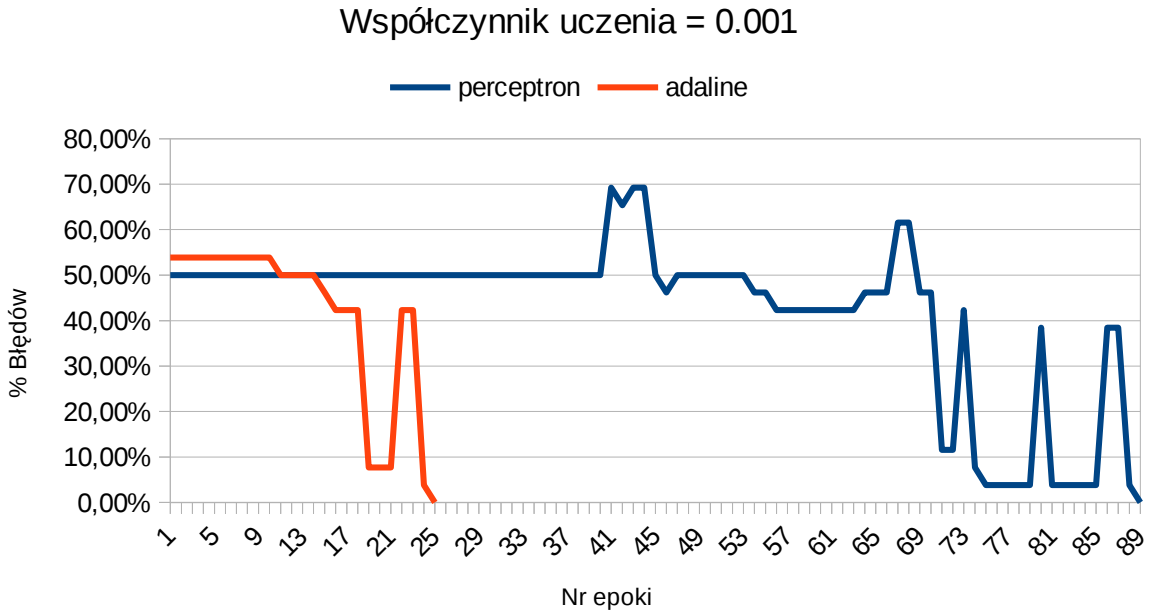
Tablicę 7 x 5 podzieliłem na 6 podobszarów. Każdy z nich to jeden sygnał wejściowy do neuronów. Jeżeli w danym obszarze pojawił się choćby jeden piksel z litery to obszar zwracał sygnał 1, w przeciwnym wypadku zwracał sygnał 0. Poniżej sposób podziału tablicy na obszary, oraz przykład wyniku dla dużej i małej litery „M”:

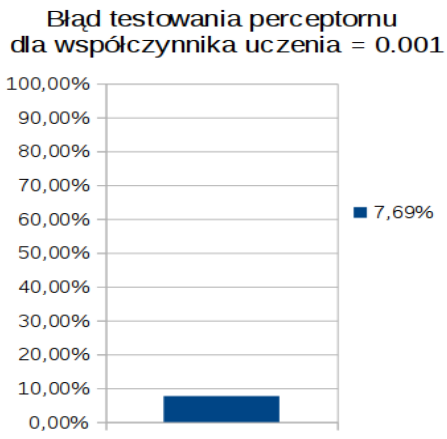



Przygotowałem 26 dużych i małych liter alfabetu. Pierwszą połowę z nich użyłem jako dane uczące a drugą połowę jako dane testujące. Proces uczenia przeprowadziłem kilkukrotnie dla różnych współczynników uczenia. Wagi początkowe dobierane są w sposób losowy, co wpływa na wyniki uczenia się, dlatego przy każdym współczynniku uczenia wykonałem kilka testów i wybrałem najlepsze rezultaty. Poniżej przedstawiam wyniki:




Współczynnik uczenia = 0.01
perceptron
adaline
60,00%
50,00%
40,00%
30,00%
20,00%
10,00%
0,00%
% Błędu
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
Nr epoki


Współczynnik uczenia = 0.001
perceptron
adaline
80,00%
70,00%
60,00%
50,00%
40,00%
30,00%
20,00%
10,00%
0,00%
% Błędów
1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89
Nr epoki


Błąd testowania perceptornu
dla współczynnika uczenia = 0.001
100,00%
90,00%
80,00%
70,00%
60,00%
50,00%
40,00%
30,00%
20,00%
10,00%
0,00%
7,69%

**3) Analiza i dyskusja błędów uczenia i testowania opracowanych sieci w zależności od wartości współczynnika uczenia oraz wybranego algorytmu:**

Zarówno w przypadku neuronu typu perceptron oraz adaline widać, że im mniejszy współczynnik uczenia, tym więcej epok zajęło nauczenie się rozpoznawania wielkości liter. Jeśli jednak porównać oba te algorytmy, to widać sporą różnicę. Przy współczynniku uczenia:

- 0.1 - perceptron 6 epok, adaline 1 epoka
- 0.01 – perceptron 19 epok, adaline 3 epoki
- 0.001 – perceptron 89 epok, adaline 25 epok

W każdym przypadku uczenia wykonałem również testowanie na pozostałej połowie liter alfabetu. W prawie wszystkich przypadkach błąd przy testowaniu wynosił 0%. Jedynie w przypadku współczynnika uczenia 0.001 błąd przy testowaniu perceptronu wynosił 7,69%. Mogły to spowodować znacznie różniące się od innych litery takie jak np. litera „s". Dane te zostały wybrane jako najlepsze ze wszystkich jakie uzyskałem w trakcie testów.
Dodatkowo jeżeli spojrzeć na wykresy, można zauważyć, głównie przy tych niższych współczynnikach uczenia, jak % błędów czasami skacze z niższych na wyższe wartości. Prawdopodobnie spowodowane było to podobieństwem między niektórymi wielkimi i małymi literami takimi jak np.: B b, F f, L l.

**4) Sformułowanie wniosków:**

Z powyższych danych jasno wynika, iż neuron typu adaline jest o wiele wydajniejszy niż perceptron. Niezależnie od współczynnika uczenia zawsze uczył się kilkukrotnie szybciej, dodatkowo podczas testowania jego błąd zawsze wynosił 0%. Wynika to z tego w jaki sposób uczy się adaline. Otóż modyfikowanie wag następuje przed funkcją aktywacji. Oznacza to, że jeżeli wartość oczekiwana będzie się znacznie różnić od wartości wyliczonej, to wagi również zostaną znacznie zmodyfikowane. Z kolei jeżeli popełniony błąd będzie niewielki, to wagi również zostaną niewiele zmienione. Co więcej, adaline różni się od perceptronu jeszcze w jednej kwestii. W przypadku perceptronu, gdzie sygnały wejściowe, tak jak i sygnały wyjściowe to 0 lub 1, modyfikacja wag będzie się odbywać tylko i wyłącznie poprzez zwiększanie ich wartości. Jeśli chodzi o adaline, to sygnały wejściowe jak i wyjściowe to -1 lub 1. Dzięki temu modyfikacja wag może odbywać się nie tylko poprzez zwiększanie ich wartości, ale także poprzez ich zmniejszanie, co znacznie poprawia dokładność.

**5) Listing całego kodu**

**Bibliografia:**

https://pl.wikipedia.org/wiki/Neuron_McCullocha-Pittsa
https://en.wikipedia.org/wiki/ADALINE
https://en.wikipedia.org/wiki/Perceptron