|  |  |  |  |
| --- | --- | --- | --- |
| Imię i nazwisko | Przedmiot, kierunek, grupa | Temat projektu | Data |
| Paulina Reichel | Podstawy sztucznej inteligencji. Inżynieria obliczeniowa. Gr.2 | Budowa i działanie perceptronu | 26.10.2018 |

**Cel projektu**:

Poznanie budowy i działania perceptronu poprzez implementację oraz uczenie perceptronu realizującego wybraną funkcję logiczną dwóch zmiennych.

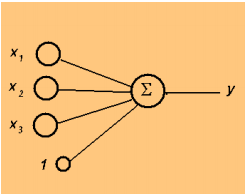
**Realizacja zadania**:

* Implementacja sztucznego neuronu według odpowiedniego algorytmu
* Wygenerowanie danych uczących i testujących funkcji logicznej dwóch zmiennych
* Uczenie perceptronu
* Testowanie perceptronu

**Opis teoretyczny**:

**Perceptron**- jest to najprostsza sieć neuronowa, składająca się z jednego bądź wielu niezależnych neuronów. Perceptron prosty ma budowę: d → 1 → 1, tzn. d neuronów wejściowych, 1 neuron ukryty i 1 wyjście. Aktywacja neuronu ukrytego jest bezpośrednio wyprowadzana na wyjście.

Graficzne przedstawienie perceptronu:



Postacie funkcji aktywującej perceptron:

* Funkcja progowa (perceptron progowy):
* Funkcja znakowa:
* Sigmoida
* Symetryczna sigmoida

**Algorytm uczenia sieci** to proces dochodzenia wag do wartości optymalnych to znaczyzapewniających odpowiednie reakcje sieci na pokazywane jej sygnały wejściowe, czyli prawidłowe rozwiązanie zadań. Istnieje kilka sposobów doborów wag i uczenia perceptronu, np. simple perceptron learning algorithm, pocket learning algorithm, pocket algorithm with ratchet. Projekt skupia się na perceptronie Rosenblatta, więc wystarczy zastosować algorytm uczenia perceptronu prostego.

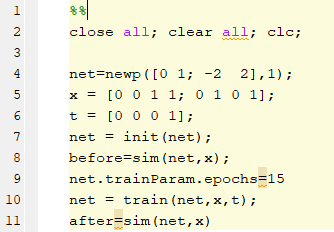
**Kroki wykonanego programu**:

* Utworzenie za pomocą funkcji newp(pr,s,tf,lf) nowego perceptronu, gdzie pr to macierz wartości, s-liczba neuronów, tf -funkcja transferu,a lf to funkcja uczenia.
* Wygenerowanie na podstawie działania bramki logicznej AND parametru określającego wagi perceptronu uczącego

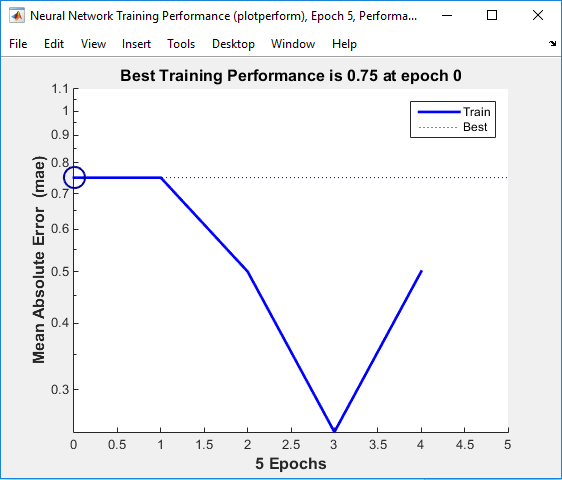
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1 | 0 | 0 | 1 | 1 |
| X2 | 0 | 1 | 0 | 1 |
| t | 0 | 0 | 0 | 1 |

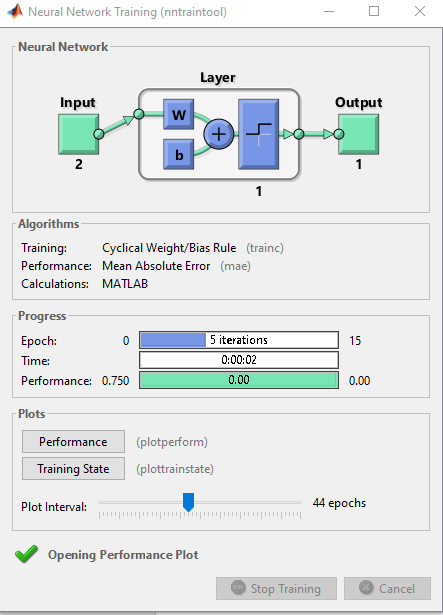
* Inicjalizacja wag oraz wykorzystanie funkcji „init” do inicjalizacji dowolnymi parametrami
* Wykorzystanie funkcji simulink, przyjmującej jako argumenty model (zainicjowany funkcją „init”) oraz wagę, do stworzenia symulacji
* Ustalenie liczby epok (maksymalnie mogą wynosić 20)
* Wykorzystanie funkcji train do uczenia perceptronu
* Otrzymanie wyniku symulacji przed i po uczeniu

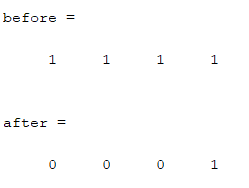
**Kod programu (Matlab)**



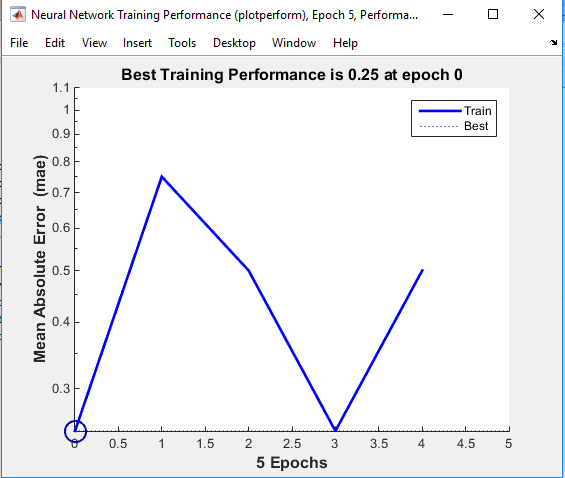
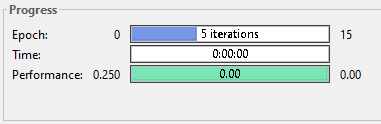
**Otrzymane wyniki**:

Dla parametru t=[0 0 0 1] (bramka logiczna AND)

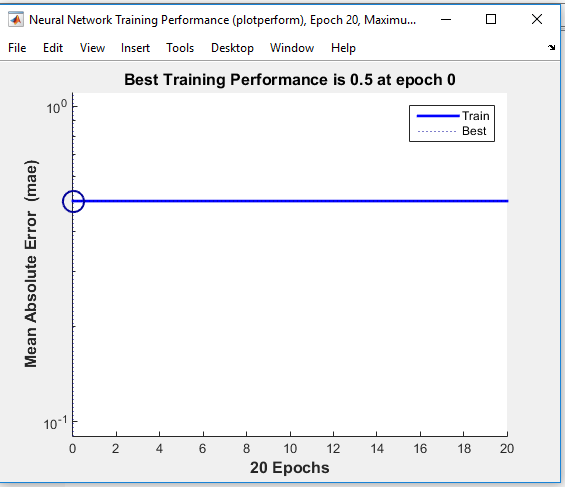
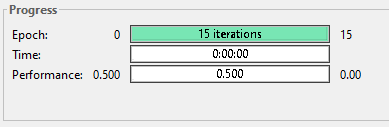
****



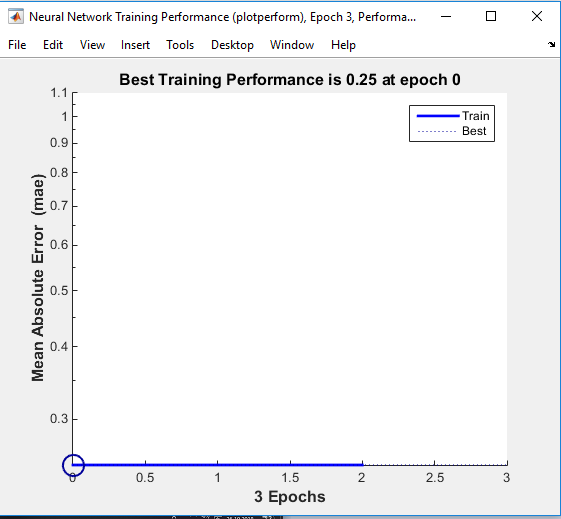
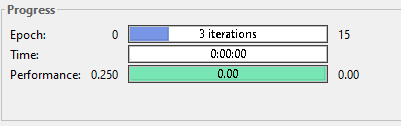
Dla parametru t=[1 1 1 0]



Dla parametru t=[0 1 1 0] oraz t=[1 0 0 1]



Dla parametru t= [0 1 1 1]



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parametr t | czas | Liczba neuronów | Wsp.mae | Ilość inicjalizacji potrzebnych do nauczenia percptronu |
| [0 0 0 1] | 0:00:02 | 1 | 0.75 | 5 |
| [1 1 1 0] | 0:00:00 | 1 | 0.25 | 5 |
| [1 0 0 1] | 0:00:00 | 1 | 0.5 | 15 |
| [0 1 1 1] | 0:00:00 | 1 | 0.25 | 3 |
| [0 0 0 1] | 0:00:02 | 6 | 0.75 | 5 |
| [1 1 1 0] | 0:00:00 | 6 | 0.25 | 5 |
| [1 0 0 1] | 0:00:00 | 6 | 0.5 | 15 |
| [0 1 1 1] | 0:00:00 | 6 | 0.25 | 3 |

**Analiza błędów**:

Analizę błędów można przeprowadzić za pomocą funkcji MAE, która jest funkcją wydajności sieci. Mierzy wydajność sieci jako średnią bezwzględnych błędów. W przypadku programu wywoływanie funkcji w kodzie jest zbędne, a wartość funkcji błędu absolutnego wystarczy odczytać z wyżej zamieszczonego wykresu. Używając bramki logicznej AND w epoce zerowej współczynnik błędu wynosi 0.75, w pierwszej epoce błąd zaczyna maleć, dochodząc do zera w epoce trzeciej. To miejsce, jest momentem nauczenia sieci. Przy użyciu bramki NAND współczynnik był dużo niższy i wynosił 0,25, jednak podobnie jak w przypadku AND nauczenie perceptronu zaszło w 3 epoce. Użycie bramki XOR nie dało wiarygodnych wyników.

**Wnioski:**

* Zadania zamieszczone w projekcie zostały zrealizowane
* Wykonanie zadania umożliwiło poznanie ogólnej zasady algorytmu uczenia perceptronu.
* Bramka logiczna ma duży wpływ na MAE.
* Za pomocą perceptronu prostego można rozwiązywać tylko problemy separowalne liniowo (np. bramka XOR powoduje powstanie konfliktu rozwiązywalnego).
* Funkcja train zawarta w pakiecie Matlab znacznie ułatwiła problem nauczenia sieci
* Uczenie perceptronu nastąpiło w 3 epoce.
* Na algorytm przeznaczone było 15 epok, lecz do nauczenie nastąpiło już 5 inicjalizacji.
* Maksymalne MAE wyniosło 0.75 (w epoce zerowej)
* W pierwszej epoce MAE zaczęło maleć, do osiągnięcia zera w 3 epoce.
* Podany algorytm okazał się wystarczający do nauczenia perceptronu prostego, lecz w przypadku bardziej skomplikowanych sieci neuronowych, może okazać się zbyt prymitywny
* Dzięki temu, że wejścia i wyjścia mogą przyjmować tylko kilka wartości, są tylko 4 możliwości zmiany wag, a każda zmiana wagi skutkuje tym, że po ponownym podaniu tego samego przykładu odpowiedź była bliższa pożądanej (uczenie sieci neuronowej).
* W zależności od zastosowanej bramki logicznej następuje zmiana wszystkich wyników- czasu, epoki, w której zostaje nauczony perceptron oraz wartość funkcji błędu absolutnego.
* Liczba neuronów nie ma wpływu na szybkość uczenia się perceptronu