**Kamil Wasilonek**

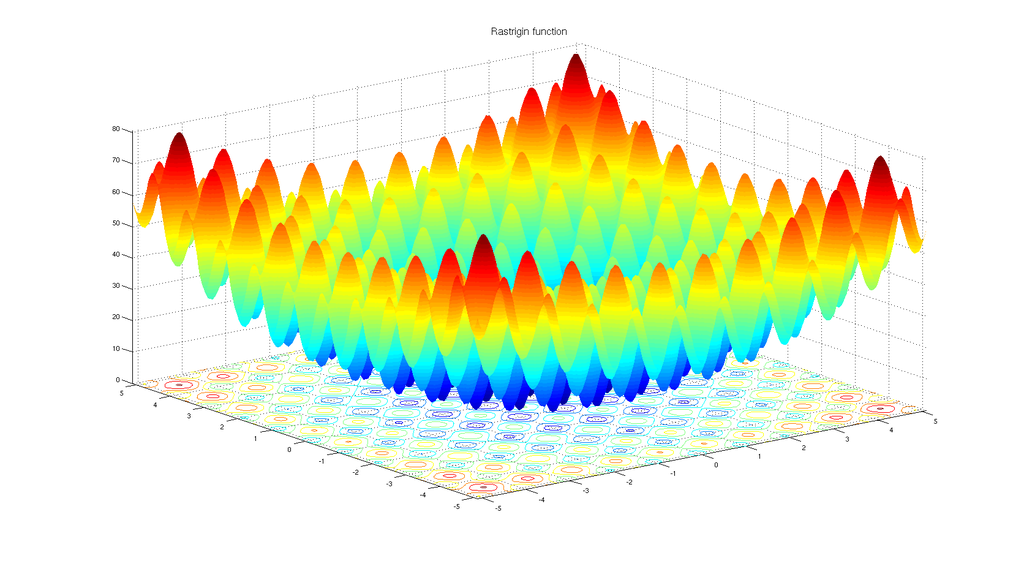
Podstawy Sztucznej Inteligencji

Sprawozdanie z projektu nr 3

Celem ćwiczenia jest poznanie budowy i działania wielowarstwowych sieci neuronowych poprzez uczenie kształtu wykresu funkcji matematycznej z użyciem algorytmu wstecznej propagacji błędu.

1. **Syntetyczny opis budowy wykorzystanego algorytmu:**

Funkcja Rastrigin jest funkcją nie [wypukłą,](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&prev=search&rurl=translate.google.pl&sl=en&sp=nmt4&u=https://en.wikipedia.org/wiki/Convex_function&usg=ALkJrhiksvrz5QJC_K5sUMOV7QAyj5W-Fw) wykorzystywaną jako problem z testowaniem wydajności dla [algorytmów optymalizacyjnych](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&prev=search&rurl=translate.google.pl&sl=en&sp=nmt4&u=https://en.wikipedia.org/wiki/Optimization_algorithm&usg=ALkJrhgh69o-aynw7e-k4inxNJK9NVIaLw). Jest to typowy przykład nieliniowej funkcji multimodalnej. Po raz pierwszy został zaproponowany przez Rastrigina  jako funkcja dwuwymiarowa i został uogólniony przez Mühlenbeina i in. Znalezienie minimalnej tej funkcji jest dość trudnym problemem z powodu dużej przestrzeni wyszukiwania i dużej liczby [lokalnych minimów](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&prev=search&rurl=translate.google.pl&sl=en&sp=nmt4&u=https://en.wikipedia.org/wiki/Local_minimum&usg=ALkJrhjne2BLUe42FtcTqSW2MEWbYjp7Ew).



Rys.1 Funckcja Rastrigin w 3D

Zakładam że podczas całego procesu trenowania sieci wykorzystane zostaną rekordy zorganizowane w kolumny: x1, x2, y.

Funkcja Rastrigin ma postać:

z = f(x, y) = 20 + x^2 + y^2 – 10 \* ( cos(2πx) + cos(2πy) )

Wartości poddałem procesowi normalizacji, według następującego wzóru:

Xnew = ( Xold - Xmin ) / ( Xmax - Xmin ) \* ( XnewMax - XnewMin ) + XnewMin

gdzie:

Xnew – nowa wartość x

Xold – stara wartość x

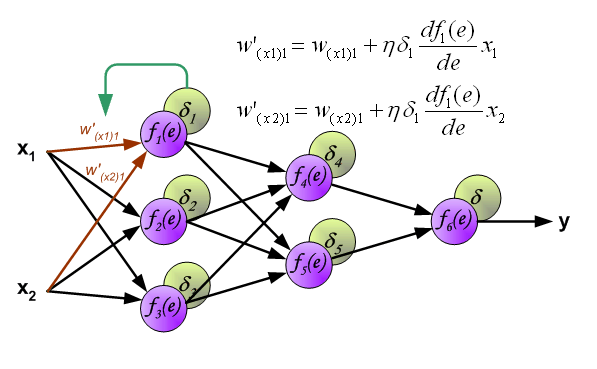
Xmin - minimalna wartość x

Xmax - maksymalna wartość x

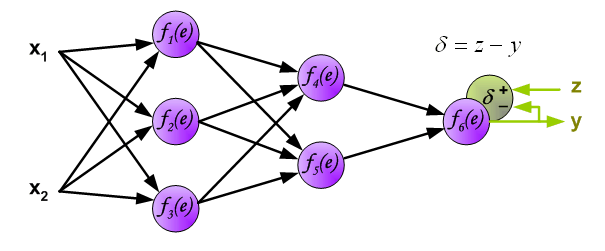
XnewMin - nowa minimalna wartość

XnewMax – nowa maksymalna wartość

**Propagacja wsteczna** – podstawowy algorytm uczenia nadzorowanego wielowarstwowych, jednokierunkowych sieci neuronowych. Podaje on przepis na zmianę wag dowolnych połączeń elementów przetwarzających rozmieszczonych w sąsiednich warstwach sieci. Oparty jest on na minimalizacji sumy kwadratów błędów (lub innej funkcji błędu) uczenia z wykorzystaniem optymalizacyjnej metody największego spadku. Dzięki zastosowaniu specyficznego sposobu propagowania błędów uczenia sieci powstałych na jej wyjściu, tj. przesyłania ich od warstwy wyjściowej do wejściowej, algorytm propagacji wstecznej stał się jednym z najskuteczniejszych algorytmów uczenia sieci.



Rys.3 Schemat ukazujący modyfikacje wag



Rys. 4 Schemat ukazujący obliczanie błędu

**Błąd średniokwadratowy** ( **MSE** ) mierzy średnią kwadratów błędów lub odchyleń - czyli różnicę między estymatorem i tym co jest szacowane. MSE jest funkcją ryzyka , odpowiadającą oczekiwanej wartości straty kwadratowej lub straty kwadratowej. Różnica wynika z przypadkowości lub dlatego, że estymator nie uwzględnia informacji, które mogłyby dać bardziej dokładny szacunek. [[1]](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&prev=search&rurl=translate.google.pl&sl=en&sp=nmt4&u=https://en.wikipedia.org/wiki/Mean_squared_error&usg=ALkJrhj42KBgyFC_uEx7Hj5NN0JOPx1bLg#cite_note-pointEstimation-1)

MSE jest miarą jakości estymatora - zawsze jest nieujemny, a wartości bliższe zeru są lepsze.

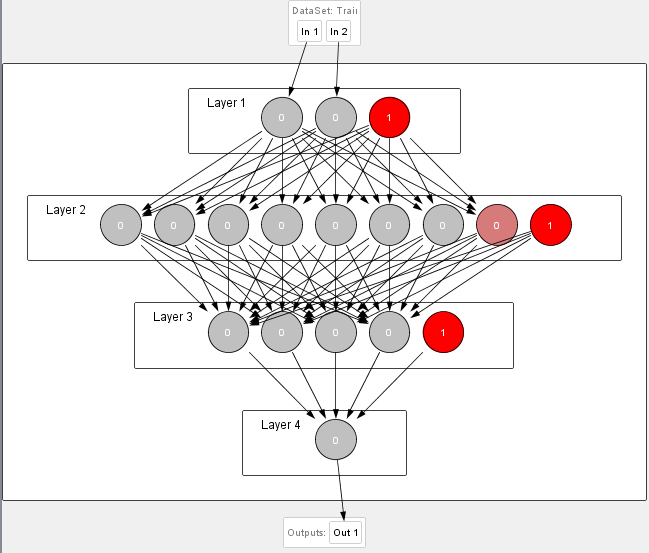


1. **Zestawienie otrzymanych wyników:**

Przeprowadziłem proces uczenia oraz testów dla 2 rodzajów sieci:

* **Wariant 1:**

**Sieć składa się 8 neuronów w pierwszej warstwie ukrytej oraz 4 neuronów w drugiej warstwie ukrytej.**

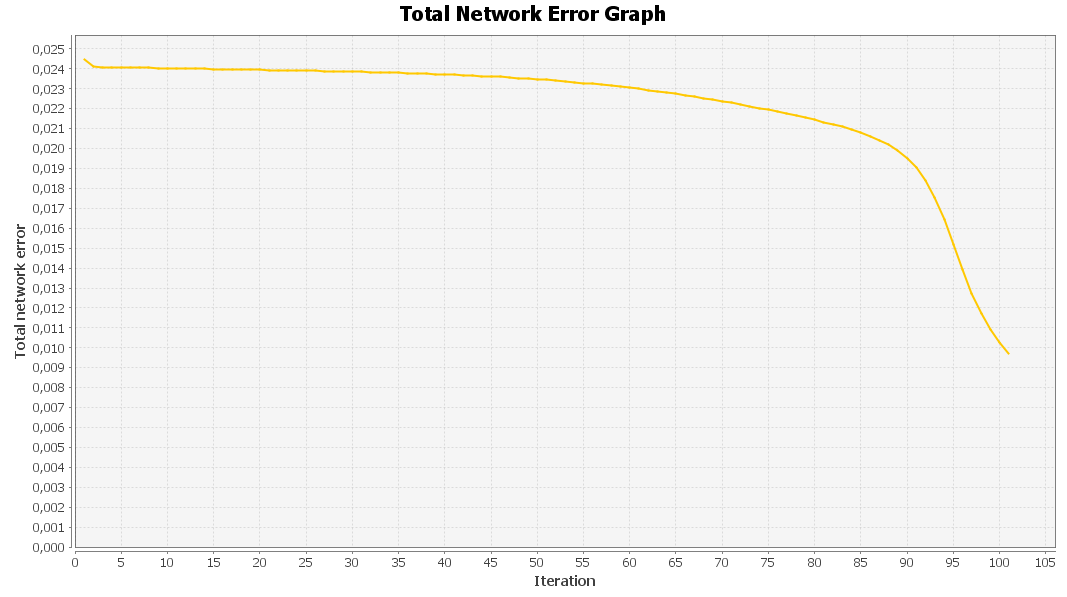


Rys . 5 Graficzne przedstawienie wykorzystanej sieci

Dla zastosowanej sieci uczenie zostało wykonane przy różnych współczynnikach uczenia. Wszystkie próby zostały przeprowadzane aż do uzyskania błędu maksymalnego 0.01.

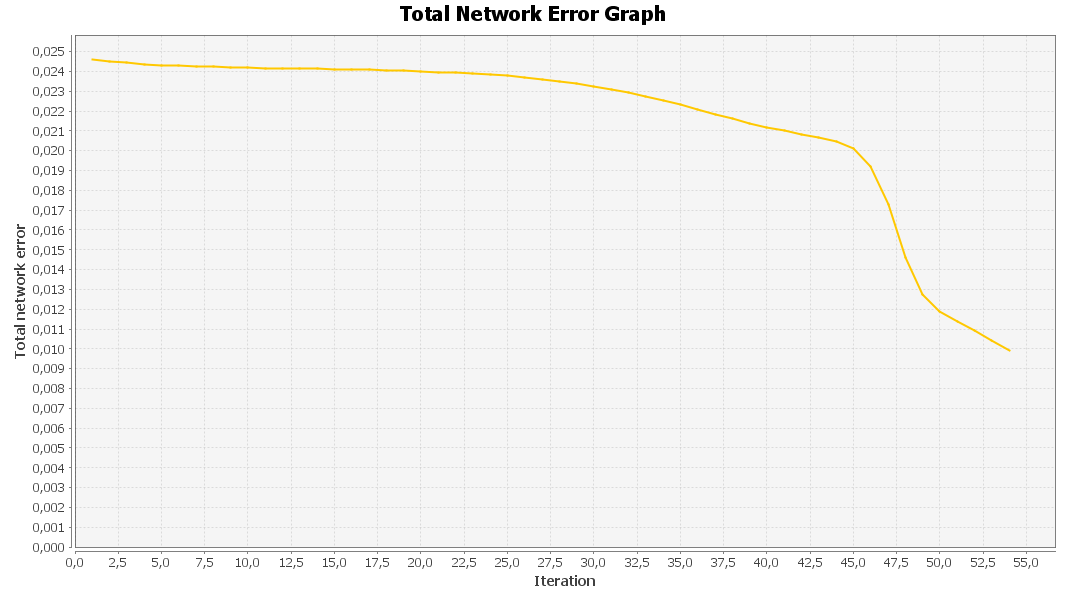
**Przypadek 1:**

**Współczynnik uczenia : 0.2**

****

Wykres .1 Wykres przedstawiający popełniony błąd w poszczególnych epokach

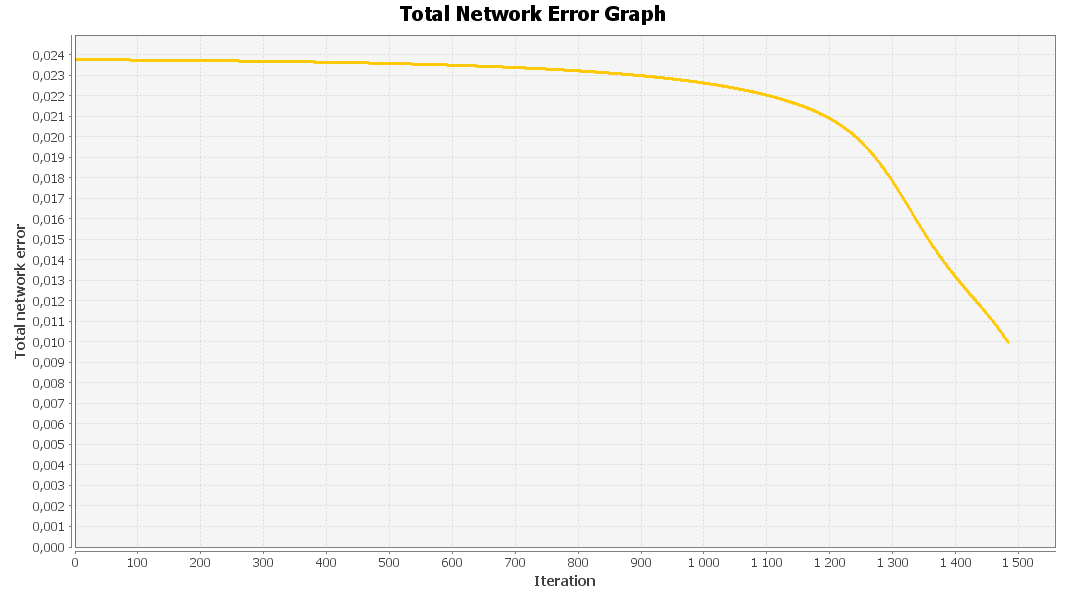
**Przypadek 2:**

**Współczynnik uczenia : 0.5**

Wykres .2 Wykres przedstawiający popełniony błąd w poszczególnych epokach

**Przypadek 3:**

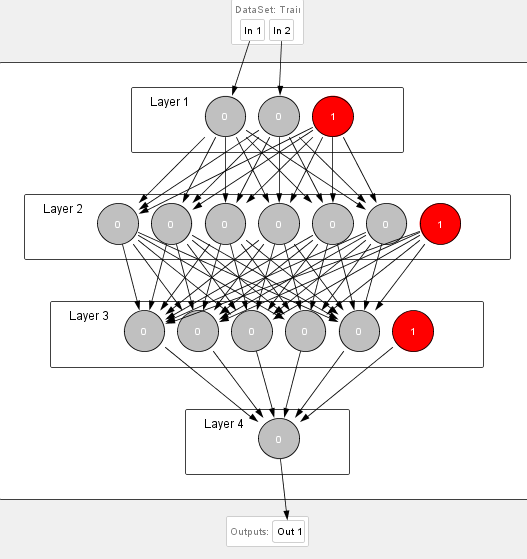
**Współczynnik uczenia : 0.01**



Wykres .3 Wykres przedstawiający popełniony błąd w poszczególnych epokach

* **Wariant 2:**

**Sieć składa się 6 neuronów w pierwszej warstwie ukrytej oraz 5 neuronów w drugiej warstwie ukrytej.**

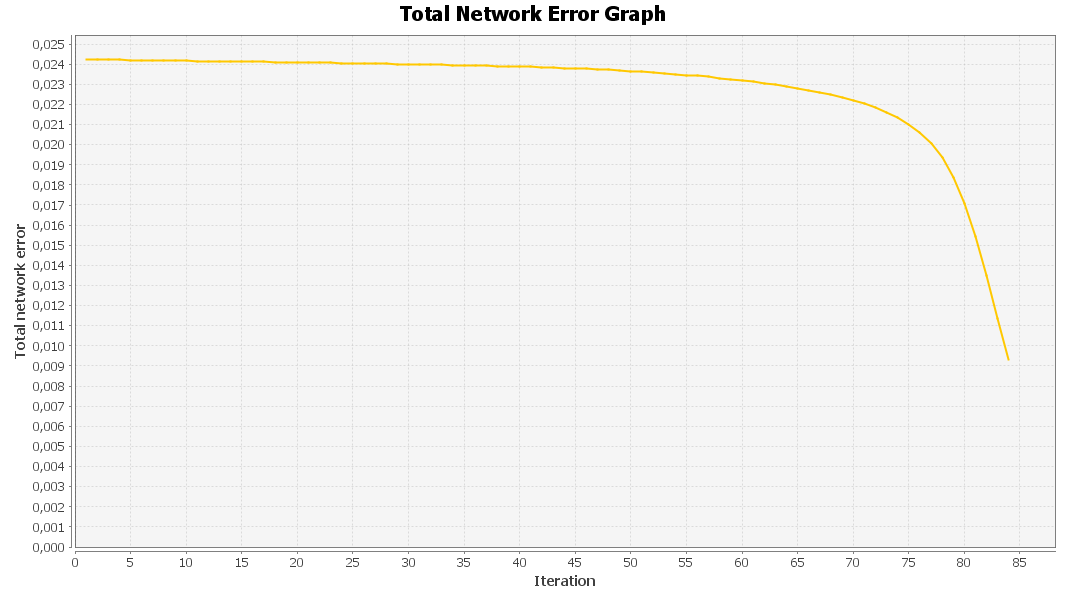


Rys . 6 Graficzne przedstawienie wykorzystanej sieci

Dla zastosowanej sieci uczenie zostało wykonane przy różnych współczynnikach uczenia. Wszystkie próby zostały przeprowadzane aż do uzyskania błędu maksymalnego 0.01.

**Przypadek 1:**

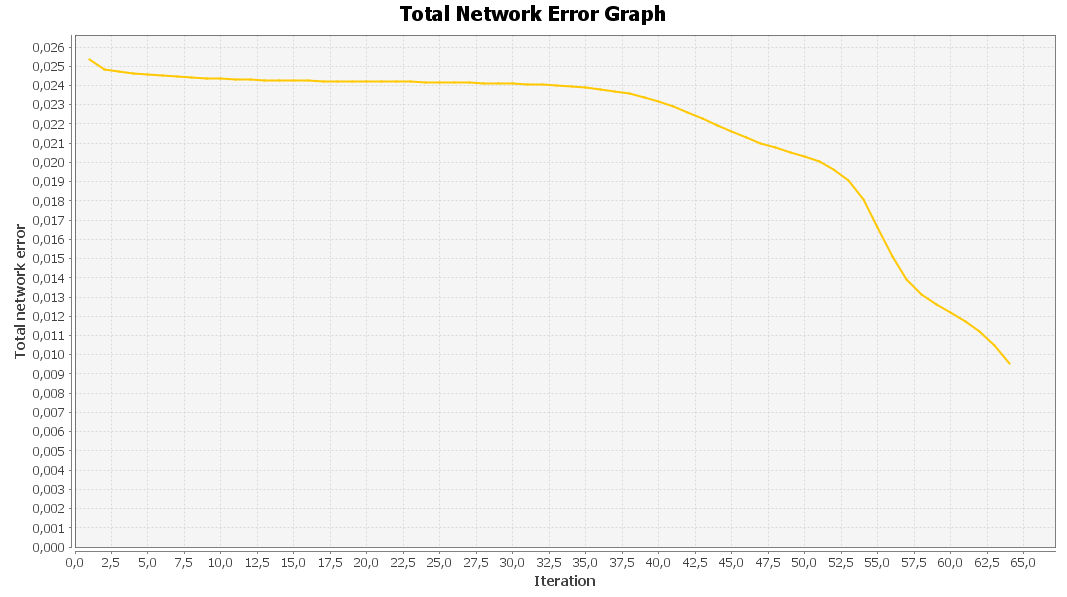
**Współczynnik uczenia : 0.2**



Wykres .4 Wykres przedstawiający popełniony błąd w poszczególnych epokach

**Przypadek 2:**

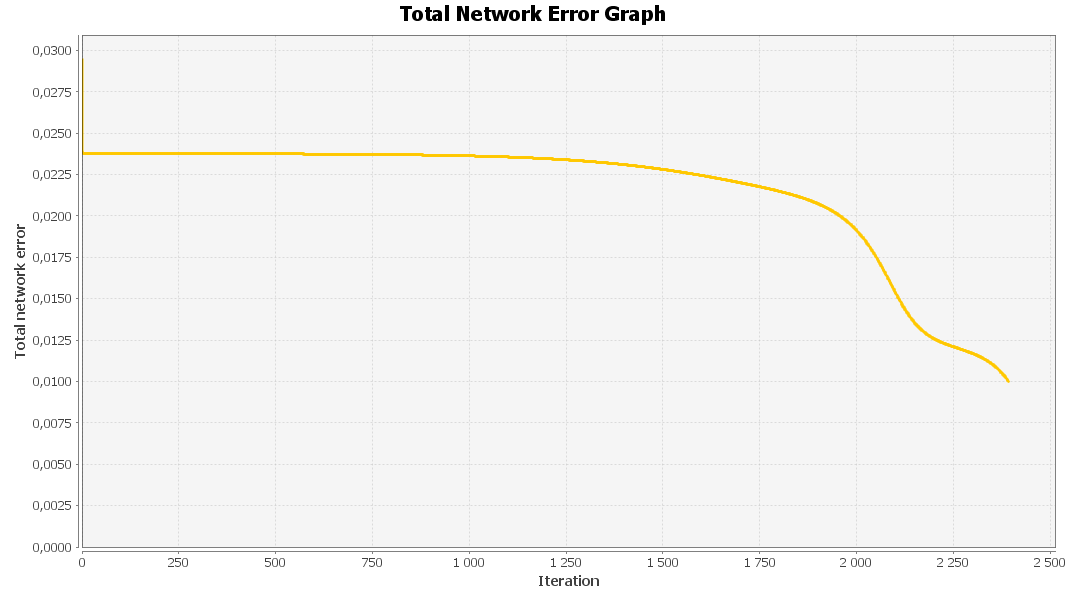
**Współczynnik uczenia : 0.5**

****

Wykres .5 Wykres przedstawiający popełniony błąd w poszczególnych epokach

**Przypadek 3:**

**Współczynnik uczenia : 0.01**



Wykres .6 Wykres przedstawiający popełniony błąd w poszczególnych epokach

1. **Testowanie przygotowanych sieci**

Po przeprowadzeniu uczenia, 30% z przygotowanych danych wykorzystałem do przetestowania sieci.

Początkowe zestawienie wyników obejmuje porównanie całkowitego błędu MSE (Total Mean Square Error) dla obu sieci przy tych samych współczynnikach uczenia. Następnie stworzyłem wykres zestawiający wszystkie otrzymane wyniki błędu.

**Przypadek 1:**

**Współczynnik uczenia = 0.2**

Wykres .7 Zestawienie wartości popełnionego błędu przez obie sieci przy wspólnym współczynniku uczenia

**Przypadek 2:**

**Współczynnik uczenia = 0.5**

Wykres .8 Zestawienie wartości popełnionego błędu przez obie sieci przy wspólnym współczynniku uczenia

**Przypadek 3:**

**Współczynnik uczenia = 0.01**

Wykres .9 Zestawienie wartości popełnionego błędu przez obie sieci przy wspólnym współczynniku uczenia

**Przypadek 4:**

**Współczynniki uczenia odpowiednio 0.2 (niebieski) , 0.5 (czerwony) , 0.01 (zielony)**

Wykres .10 Zestawienie wartości popełnionego błędu przez obie sieci

1. **Analiza i dyskusja błędów uczenia i testowania opracowanej sieci w zależności od wartości współczynnika uczenia oraz ilości warstw i neuronów**

Tabela.1 Zestawienie błędu MSE poszczególnych prób testujących

|  |  |  |
| --- | --- | --- |
| Rodzaj sieci | Współczynnik uczenia | Błąd MSE |
| Sieć 2,8,4,1 | 0.2 | 0,0162162918110996 |
| 0.5 | 0,0185562065953516 |
| 0.01 | 0,0205623538660648 |
| Sieć 2,6,5,1 | 0.2 | 0,0195827517362608 |
| 0.5 | 0,0167252692249414 |
| 0.01 | 0,0193649470234874 |

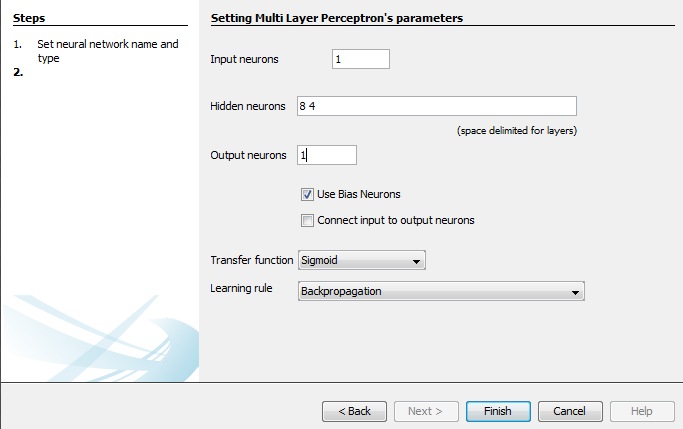
W przypadku używanych przeze mnie sieci znaczna zmiana współczynnik uczenia bardzo wydłużyła proces nauki sieci. Dla wartości 0.2 i 0.5 obie sieci uzyskiwały wartość wyznaczoną jako maksymalny błąd uczenia tj. 0.01 w mniej niż stu epokach, jednak zmiana wartości na 0.01 spowodowało skok czasu nauki do ponad tysiąca epok. Analizując wyniki testów sieci 2,8,4,1 najniższy błąd uczenia wystąpił dla współczynnika 0.2 , w przypadku drugiej sieci najlepszy wynik uzyskuje współczynnik 0.5.

1. **Sformułowanie wniosków**

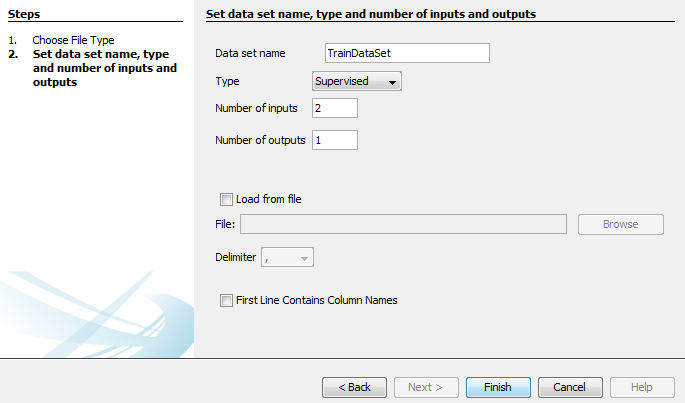
Na podstawie przeprowadzonego ćwiczenia mogę określić ze w moim przypadku budowa sieci tj. różna ilość neuronów w warstwach ukrytych nie miały znaczącego wpływu na otrzymane wyniki.. Obie sieci dla odpowiednich wag uczyły się w podobnej ilości epok , wyjątkiem jest proces uczenia dla współczynnika uczenia 0.01 ponieważ w tym przypadku sieć 2,8,4,1 wykonał się znacznie szybciej niż dla sieci 2,6,5,1. Niezależnie od budowy sieci możemy zaobserwować ze zmniejszanie współczynnika uczenia powoduje wzrostu czasu uczenia. Należy jednak uwagę ze w zależności od zadanego problemu konstrukcja sieci może w równym stopniu co współczynnik wpływać na działanie programu.

1. **Przedstawienie użytego narzędzia**

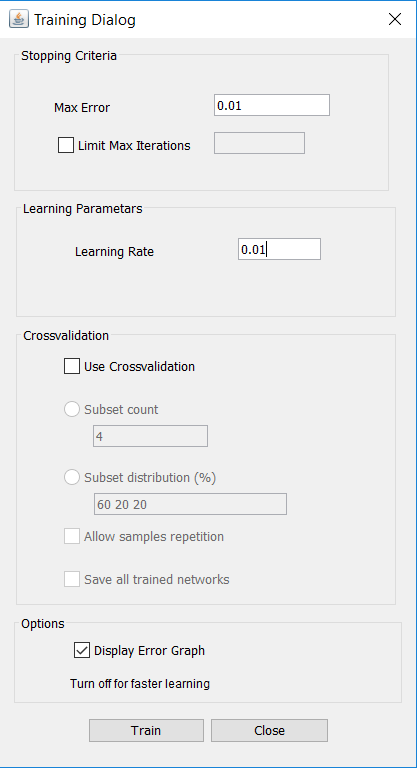
Do wykonania ćwiczenia użyłem „NeurophStudio” przy pomocy którego wykonałem przedstawione wyżej sieci oraz testy. Poniżej przedstawiam widoki okien konfiguracji programu w których dostosowywałem program do założonych przeze mnie danych.

****Rys. 7 Widok dostępnych parametrów sieci

W powyższym widoku miałem możliwość ustawienia ilości neuronów wejściowych , wyjściowych oraz warstw ukrytych wraz z ilością neuronów w tych warstwach. Następnie wybrałem opcje funkcji sigmoidalnej. Jako warunek uczenia przyjąłem opisywaną we wstępie metodę wstecznej propagacji.

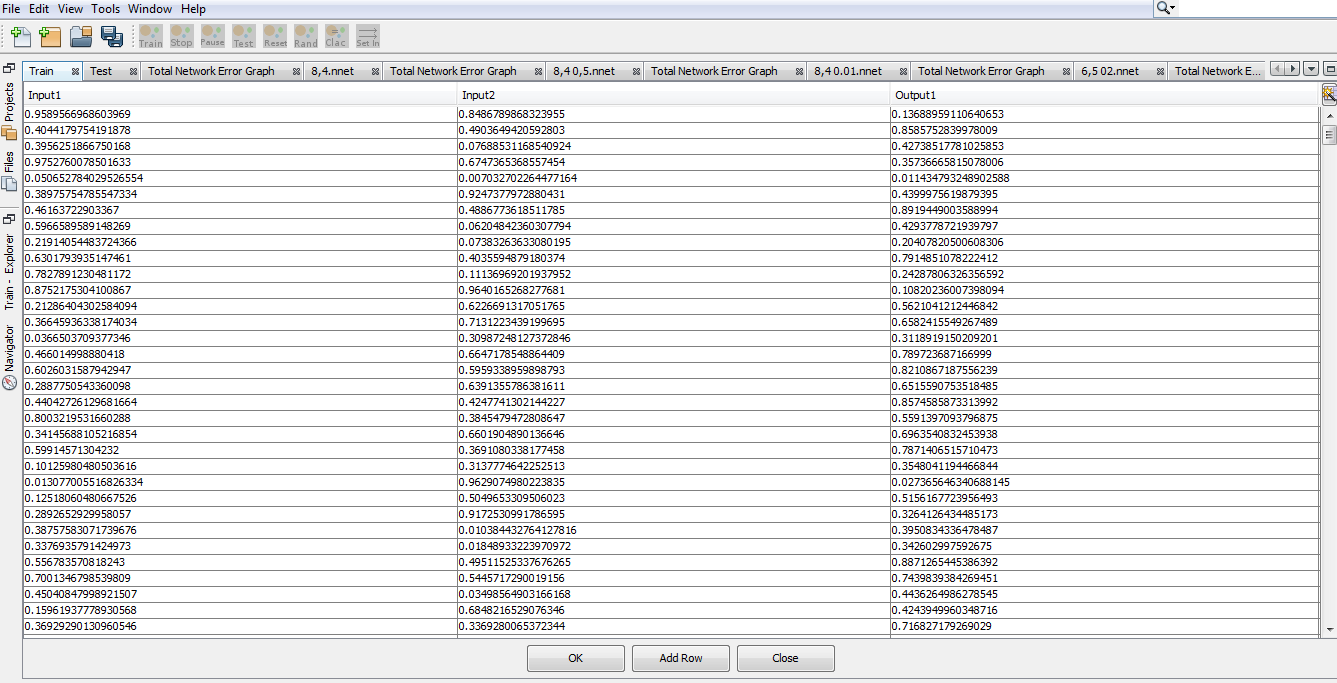
Rys. 8 Widok dostępnych parametrów wczytywania danych

Przedstawione okno ukazuje możliwość dostosowania danych wejściowych(uczących) oraz testujących. Informacje możemy wczytać z pliku lub utworzyć w wygenerowanym formularzu



Rys. Okno ustawień danych dotyczących uczenia

Ten ekran umożliwia ustawienie „Max Error” który odpowiada za przerwanie procesu uczenia i uznanie go za poprawnie wykonany po osiągnięciu podanej wartości. Dodatkowo określany współczynnik uczenia który ma bardzo duży wpływ proces uczenia.



Rys. 5 Przykładowy widok wczytanych danych wejściowych oraz testujących

Po wczytaniu danych do programu są one przedstawiane w powyższy sposób

Bibliografia:

<http://edu.pjwstk.edu.pl/wyklady/nai/scb/wyklad3/w3.htm>

<https://translate.google.pl/translate?hl=pl&sl=en&u=https://en.wikipedia.org/wiki/Mean_squared_error&prev=search>

<http://galaxy.agh.edu.pl/~vlsi/AI/backp_t/backprop.html>