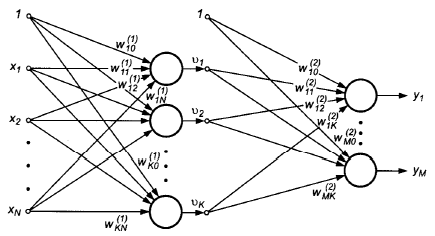
**Sprawozdanie -scenariusz 3**

**Temat ćwiczenia:** Budowa i działanie sieci wielowarstwowej.

**Cel ćwiczenia:** Celem ćwiczenia jest poznanie budowy i działania wielowarstwowych sieci neuronowychpoprzez uczenie z użyciem algorytmu wstecznej propagacji błędu rozpoznawaniakonkretnych liter alfabetu.

1. **Opisz budowy wykorzystanej sieci i algorytmu uczenia**

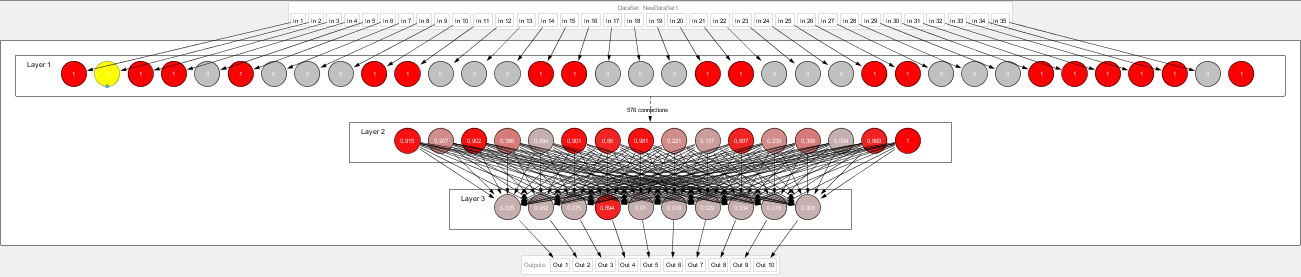
Siecią, która była wykorzystana w moim projekcie była sieć wielowarstwowa. Sieć taką tworzą neurony ułożone w wielu warstwach. Są to warstwy: wejściowa, wyjściowa oraz przynajmniej jedna warstwa ukryta. Poniżej został przedstawiony schemat takiej sieci.



W moim programie sieć składa się z trzech warstw: wejściowej(35 neuronów), wyjściowej(10 wyjściowej) i jednej ukrytej(15 neuronów). Wykorzystanym algorytmem był algorytm propagacji wstecznej. Określa on strategie doboru wag w sieci wielowarstwowej przy wykorzystaniu gradientowych metod optymalizacji.

Ogólny schemat procesu trenowania sieci wygląda następująco:

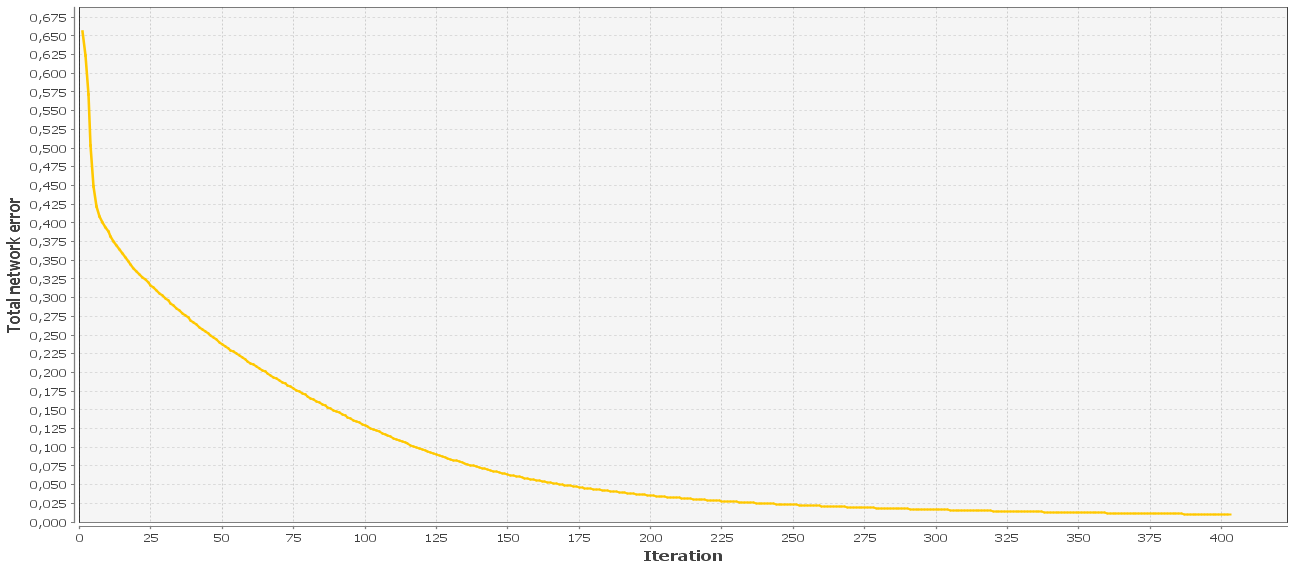
* Ustalamy topologię sieci, tzn. liczbę warstw, liczbę neuronów w warstwach.
* Inicjujemy wagi losowo (na małe wartości).
* Dla danego wektora uczącego obliczamy odpowiedź sieci (warstwa po warstwie).
* Każdy neuron wyjściowy oblicza swój błąd, oparty na różnicy pomiędzy obliczoną odpowiedzią *y* oraz poprawną odpowiedzią *t*.
* Błędy propagowane są do wcześniejszych warstw.
* Każdy neuron (również w warstwach ukrytych) modyfikuje wagi na podstawie wartości błędu i wielkości przetwarzanych w tym kroku sygnałów.
* Powtarzamy od punktu 3. dla kolejnych wektorów uczących. Gdy wszystkie wektory zostaną użyte, losowo zmieniamy ich kolejność i zaczynamy wykorzystywać powtórnie.
* Zatrzymujemy się, gdy średni błąd na danych treningowych przestanie maleć. Możemy też co jakiś czas testować sieć na specjalnej puli nieużywanych do treningu próbek testowych i kończyć trenowanie, gdy błąd przestanie maleć.



\*na rysunku powyżej widać schemat wykorzystanej sieci

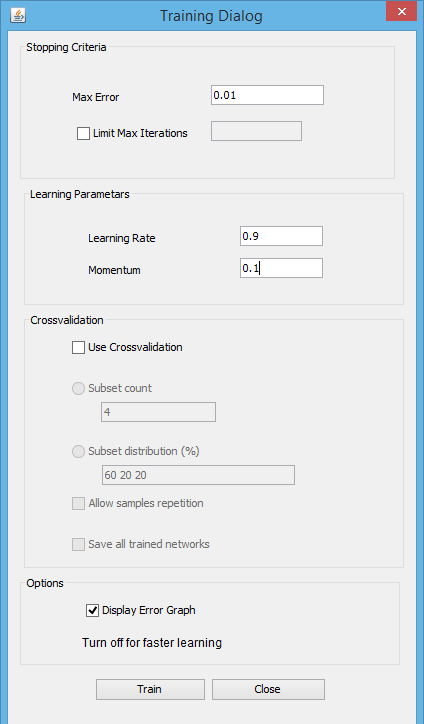
1. **Zestawnie otrzymanych wyników**

Wykres błędu sieci dla współczynnika uczenia sieci = 0,2 oraz bezwładności = 0,7



|  |  |  |
| --- | --- | --- |
| współczynnik uczenia | bezwładność | Ilość iteracji |
| 0,2 | 0,7 | 405 |
| 0,9 | 0,1 | 93 |
| 0,1 | 0,9 | 780 |
| 0,2 | 0,1 | 410 |
| 0,2 | 0,9 | 420 |

Przykładowe ustawienie współczynnika uczenia i bezwładności:



Wyniki rozpoznawania liter np. A,D oraz C spośród liter od A do J

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I | J |
| A | **0,922** | 0,07 | 0,021 | 0,04 | 0,022 | 0,018 | 0,012 | 0,019 | 0,019 | 0,008 |
| D | 0,026 | 0,067 | 0,064 | **0,907** | 0,014 | 0,023 | 0,016 | 0,019 | 0,016 | 0,018 |
| C | 0,024 | 0,038 | **0,93** | 0,058 | 0,026 | 0,019 | 0,032 | 0,022 | 0,02 | 0,027 |

1. **Wnioski**

Wykorzystanie sieci wielowarstwowej i algorytmu propagacji wstecznej umożliwiło poprawne uczenie sieci i rozpoznawanie zadanych liter. Z przeprowadzonych testów wynika, że współczynnik uczenia wpływa znacząco na szybkość uczenia sieci. Im większy współczynnik uczenia, tym mniej iteracji potrzeba do uczenia sieci. Bezwładność zaś nie wpływa znacząco na szybkość uczenia. Przy kilkukrotnym uruchamianiu programu dla tych samych wartości współczynników, liczba potrzebnych iteracji do nauczenia nie różniła się znacząco(np. 3 iteracje).

1. **Wykorzystane źródła:**

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

<http://www.ai.c-labtech.net/sn/pod_prakt.html>

<http://neuroph.sourceforge.net/index.html>