**LABOLATORIUM NR 3  
Michał Ligęza  
Sieć wielowarstwowa – nauka liter**

1. **Opis sieci**  
   W trakcie przeprowadzania laboratorium wykorzystano gotową bibliotekę Neuroph przygotowaną dla języka Java. Biblioteka posiada gotowe potrzebne rozwiązania, dzięki czemu więcej czasu można było poświęcić wielowarstwowości i analizie działania bardziej skomplikowanych sieci. Sieć posiadała 144 wejścia - odpowiada to literom zapisanym na bazie 8x18. Wewnętrzna warstwa posiadała 10 neuronów, natomiast ostatnia, wyjściowa 20 – tyle ile liter sieć miała za zadanie się nauczyć. Sieci zadano błąd 0.001, wyniki to ilość iteracji potrzebnych do nauczenia się całości zestawu danych.
2. **Wyniki**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Współczynnik  nauki | Wyniki | | | | | | | | | | Średnia |
| 0.1 | 4385 | 5025 | 4233 | 4585 | 12473 | 8981 | 4339 | 8502 | 5337 | 5977 | 6383 |
| 0.2 | 3038 | 5951 | 3857 | 3738 | 3909 | 2263 | 6024 | 3214 | 2611 | 2226 | 3683 |
| 0.3 | 14571 | 1850 | 3150 | 3806 | 1818 | 6805 | 1666 | 2520 | 4326 | 1636 | 4241 |
| 0.4 | 1134 | 2336 | 1678 | 1836 | 1766 | 2141 | 2478 | 4369 | 1853 | 1837 | 2142 |
| 0.5 | 4000 | 4652 | 2524 | 8033 | 1382 | 1336 | 1456 | 18134 | 1574 | 3458 | 4654 |
| 0.6 | 1620 | 2748 | 2882 | 2089 | 1632 | 5261 | 1087 | 5957 | 7107 | 6103 | 3648 |

1. **Analiza**Jak można zauważyć po przedstawionym wykresie, tak jak i po tablicy wyników, wyniki były bardzo rozbieżne, rozmyte i trudno było jednoznacznie określić szybkość wykonania dla różnych współczynników nauki. Choć większość wyników malała z wzrastającym współczynnikiem, to pojawiające się pojedyncze wyniki będące bardzo odległe od przeciętych, silnie zmieniały średnią przez co na podstawie wykresu nie można jednoznacznie określić najlepszego parametru sieci
2. **Wnioski**Otrzymane wyniki, które trudno uznać za bardzo dobre lub dobre, prawdopodobnie są wynikiem zastosowanego rozwiązania oraz układem sieci – powiększając warstwę ukrytą lub dodając kolejną istnieje duża szansa że można byłoby poprawić wyniki. Takie rozbieżności mogły zostać spowodowane także przez zadany maksymalny błąd – przez to sieć nawet będąc blisko rozwiązania, musiałaby wielokrotnie poprawiać naukę.
3. **LISTING KODU**

public class Net {  
  
 public static void main(String[] args) {  
 DataSet data = new DataSet(144, 20);  
 data.addRow(*getData*((char) "A".toCharArray()[0])); - zestawy danych  
  
  
MultiLayerPerceptron multiLayerPerceptron = new MultiLayerPerceptron(TransferFunctionType.*SIGMOID*, 144, 10, 20);   
multiLayerPerceptron.setLearningRule(backPropagation); - rodzaj nauki  
multiLayerPerceptron.learn(data); -ustawienie zestawu danych  
multiLayerPerceptron.save("my.nnet");-zapisanie do pliku

BackPropagation backPropagation = new BackPropagation();  
backPropagation.setMaxError(0.001);-wymagany maksymalny błąd  
backPropagation.setLearningRate(0.5); -współczynnik nauki

System.*out*.println("Ilosc iteracji: " + backPropagation.getCurrentIteration());-wyświetlenie wyniku

NeuralNetwork neuralNetwork = NeuralNetwork.*createFromFile*("my.nnet");-wczytanie sieci  
  
neuralNetwork.setInput(data.getRows().get(19).getInput()); -wybranie wyjścia  
System.*out*.println("Sprawdzam U:");   
neuralNetwork.calculate(); -zbadanie wyniku  
DecimalFormat df = new DecimalFormat("0.00000000");  
for (double n : neuralNetwork.getOutput())  
 System.*out*.println(df.format(n)); -wyswietlenie wyniku