Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Кафедра автоматизації проектування енергетичних процесів і систем

Лабораторна робота №2

з дисципліни «Методології розробки інтелектуальних комп'ютерних програм»

«**Розпізнавання образів за допомогою штучних нейронних мереж**»

Виконала: студентка 3 курсу

групи ТІ-01

Круть Катерина

Перевірив: д.т.н. Мусієнко А. П.

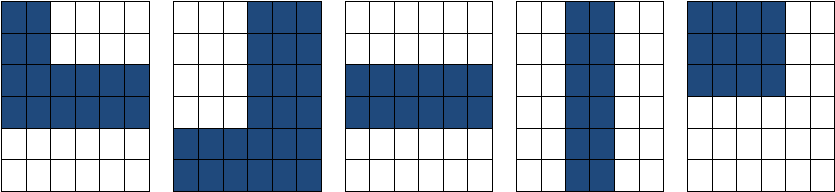
**КИЇВ 2023**

**ПОСТАНОВКА МЕТИ ДОСЛІДЖЕННЯ**

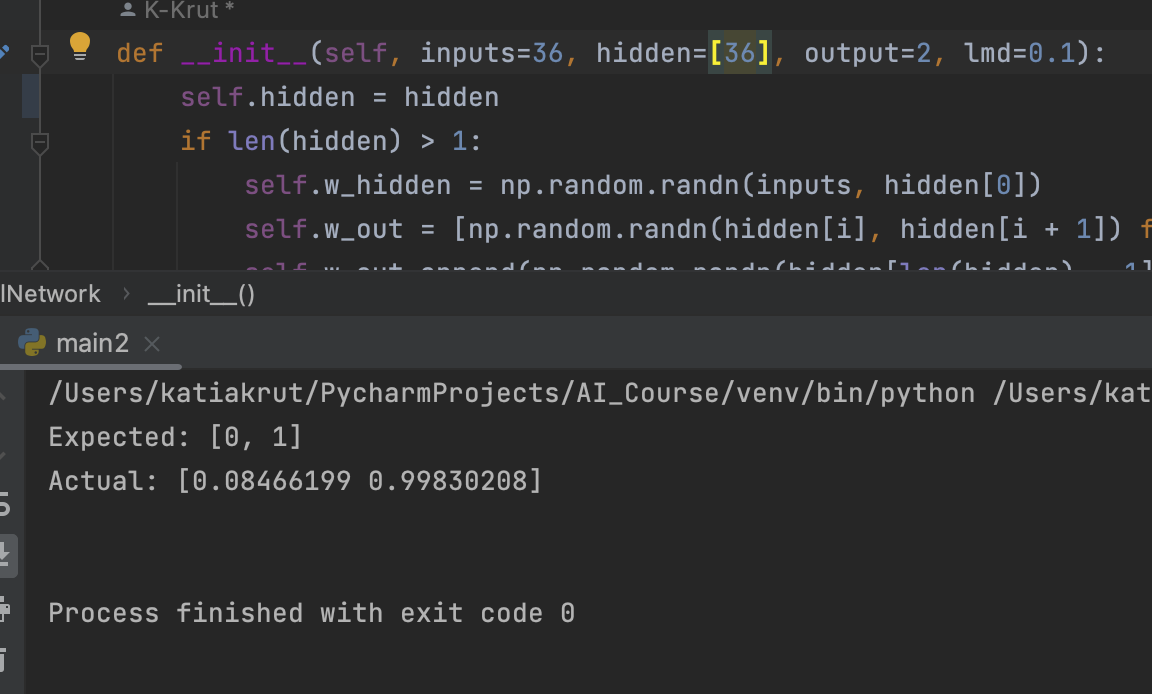
**Мета дослідження:** Ознайомитися з методами розпізнавання образів за допомогою штучних нейронних мереж, побудувати, навчити та протестувати нейронну мережу для розпізнавання букв.

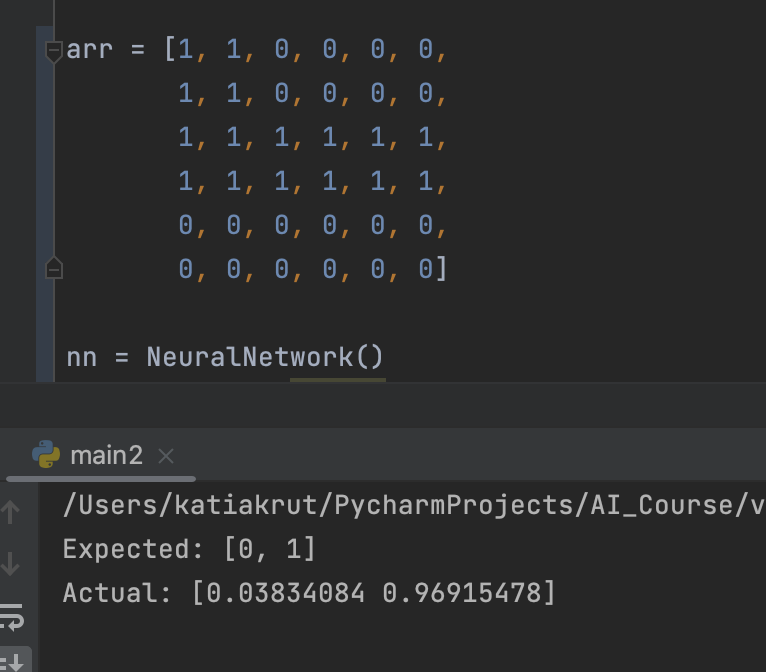
**Набір даних:**

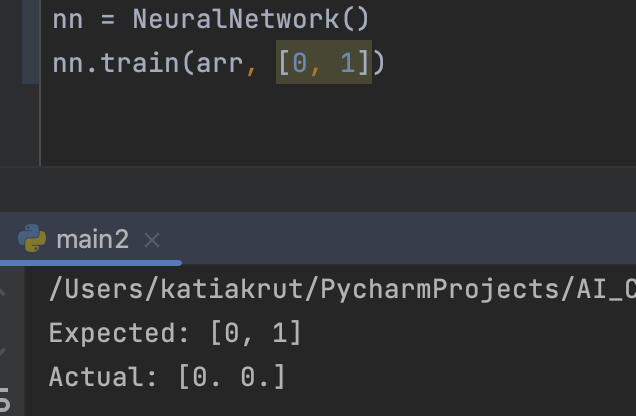
**Варіант 12**

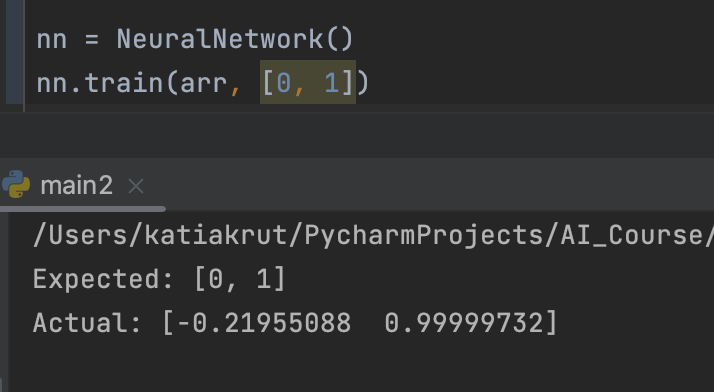
****

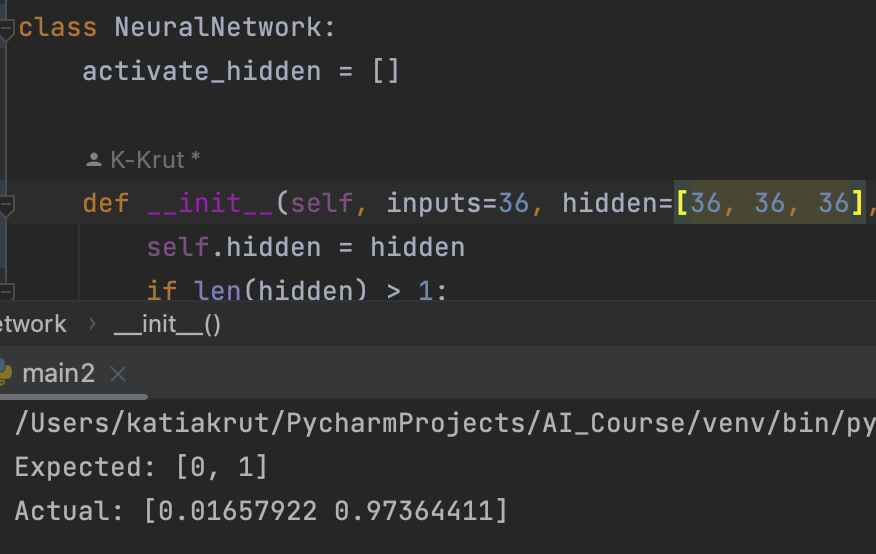
**ХІД РОБОТИ**

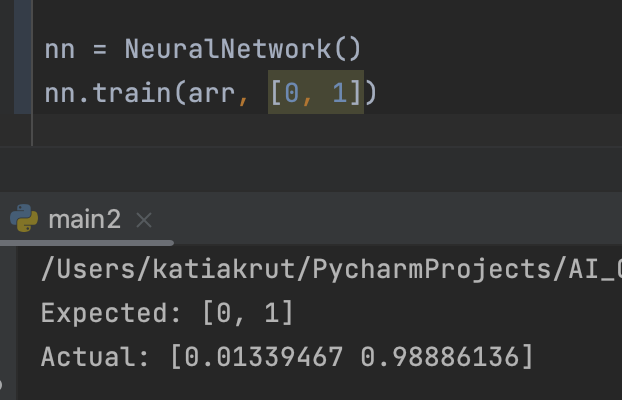
1. Один прихований шар з 36 нейронівз функцією активації SIGMOID, 100 епох:

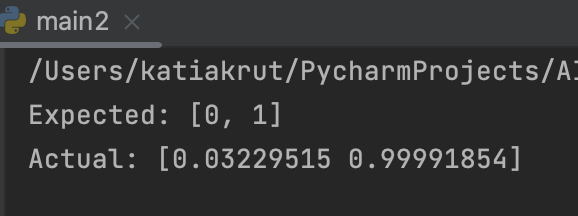
2. Два прихованих шари кожен з який складається з 36 нейронів та активатором SIGMOID, 100 епох:

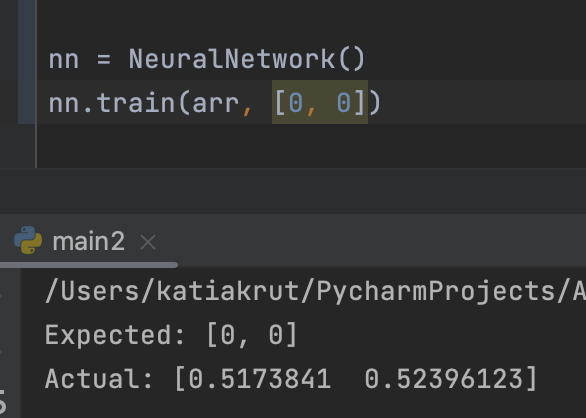
5. Два прихованих шари кожен з який складається з 36 нейронів та активатором RELU, 100 епох:

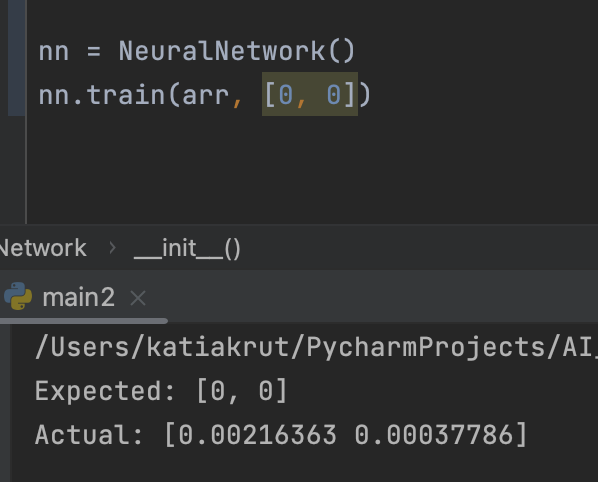
6. Два прихованих шари кожен з який складається з 36 нейронів та активатором TANH, 100 епох:

7. Три прихованих шари з 36 нейронів та активаторами SIGMOID, 100 епох:

8. Три прихованих шари з 72 нейронів та активатором SIGMOID, 1000 епох:

9. Три прихованих шари з 72 нейронів та активатором SIGMOID, 100 епох:

15. Три прихованих шари (36, 36, 36) та активатором TANH, 1000 епох:

17. Два прихованих шари з 72 нейронів з активатором SIGMOID, 5000 епох:

**КОПІЯ ВИКОНАНОЇ ПРОГРАМИ**

import org.datavec.api.records.reader.RecordReader;  
import org.datavec.api.records.reader.impl.csv.CSVRecordReader;  
import org.datavec.api.split.FileSplit;  
import org.deeplearning4j.datasets.datavec.RecordReaderDataSetIterator;  
import org.nd4j.common.io.ClassPathResource;  
import org.nd4j.linalg.dataset.DataSet;  
import org.nd4j.linalg.dataset.api.iterator.DataSetIterator;  
  
public class DataSetReader {  
 private static final int *FEATURES\_COUNT* = 36;  
  
 public static DataSet readDataFromFile(String fileName) {  
 DataSet dataSet = null;  
 try (RecordReader recordReader = new CSVRecordReader(0, ' ')) {  
 recordReader.initialize(new FileSplit(  
 new ClassPathResource(fileName).getFile()));  
 DataSetIterator iterator = new RecordReaderDataSetIterator(  
 recordReader, 5, *FEATURES\_COUNT*, 5);  
 dataSet = iterator.next();  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 }  
 return dataSet;  
 }  
}

import org.deeplearning4j.nn.conf.MultiLayerConfiguration;  
import org.deeplearning4j.nn.conf.NeuralNetConfiguration;  
import org.deeplearning4j.nn.conf.layers.DenseLayer;  
import org.deeplearning4j.nn.conf.layers.OutputLayer;  
import org.deeplearning4j.nn.multilayer.MultiLayerNetwork;  
import org.deeplearning4j.nn.weights.WeightInit;  
import org.nd4j.linalg.activations.Activation;  
import org.nd4j.linalg.api.ndarray.INDArray;  
import org.nd4j.linalg.dataset.DataSet;  
import org.nd4j.linalg.lossfunctions.LossFunctions;  
  
public class NetworkConfig {  
  
 private MultiLayerConfiguration configuration;  
 private MultiLayerNetwork network;  
  
 private INDArray output;  
  
 public NetworkConfig createConfig() {  
 this.configuration = new NeuralNetConfiguration.Builder()  
 .weightInit(WeightInit.*UNIFORM*)  
 .list()  
 .layer(0, new DenseLayer.Builder()  
 .activation(Activation.*SIGMOID*)  
 .nIn(36)  
 .nOut(72)  
 .build())  
 .layer(1, new OutputLayer.Builder()  
 .activation(Activation.*SOFTMAX*)  
 .nIn(72)  
 .nOut(5)  
 .build())  
 .build();  
 return this;  
 }  
  
 public NetworkConfig initializeNetwork() {  
 network = new MultiLayerNetwork(configuration);  
 network.init();  
 return this;  
 }  
 public NetworkConfig trainNetwork(DataSet dataSet, double learningRate, int epochs) {  
 this.network.setLearningRate(learningRate);  
 for (int i = 0; i < epochs; i++) {  
 network.fit(dataSet);  
 }  
 return this;  
 }  
  
 public NetworkConfig createOutput(DataSet dataset) {  
 this.output = network.output(dataset.getFeatures());  
 return this;  
 }  
  
 public INDArray getOutput() {  
 return output;  
 }  
}

import org.nd4j.linalg.dataset.DataSet;  
  
public class Main {  
  
 public static void main(String[] args) {  
 NetworkConfig networkConfig = new NetworkConfig();  
 DataSet dataSet = DataSetReader.*readDataFromFile*("train.txt");  
 DataSet testDataSet = DataSetReader.*readDataFromFile*("test.txt");  
 networkConfig.createConfig()  
 .initializeNetwork()  
 .trainNetwork(dataSet, 0.1, 1\_000)  
 .createOutput(testDataSet);  
 System.*out*.println(networkConfig.getOutput());  
  
 }  
}

**Висновки:** В ході виконання даної лабораторної роботи було отримано відомості про можливість розпізнавання певних образів за допомогою штучних нейронних мереж. Для виконання поставленої задачі знову було використано алгоритм зворотного поширення похибки. На основі набутих знань було розроблено нейронну мережу, яка з указаними умовами здатна розпізнавати образи, які частково відрізняються від початкових. Також було проведено порівняння роботи нейронної мережі залежно від різних функцій активацій, кількості прихованих шарів, кількості нейронів у прихованих шарах тощо. Можна впевнено стверджувати, що найгіршою комбінацією є прихований шар з функцією активації SIGMOID та зовнішній шар з функцією активації SOFTMAX. Загалом же найкращі результати були отримані завдяки використанню функції активації RELU в різних комбінаціях.