Cairo University
Faculty of Computers & Artificial Intelligence
Department of Computer Science
Soft Computing Course - 2021

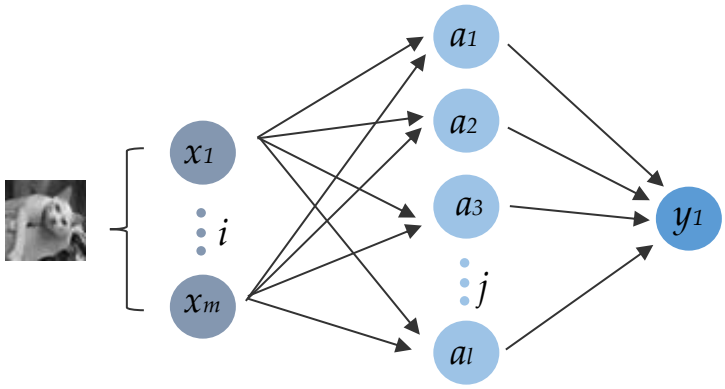
Assignment 4 – Image Classification Using a Neural Network

About the problem:

Image classification is a supervised learning problem in which a set of target classes (objects to identify in images) are defined, and a model is trained to recognize them using labeled example photos.

What you are required to do:

Implement a feedforward neural network (from scratch) that classifies animals in images into 2 classes: cats and dogs. This FFNN should have 3 layers: input, hidden, and output.



You will be given the following:

- **"Cats & Dogs Sample Dataset" folder** which contains 20 images of cats and 20 images of dogs. Each image is grayscale and already scaled to 40x40 pixels.
- **"ImageHandler.java" file** which contains a class implemented to manipulate an image. It contains a function that opens an image as a 1D array of pixels and a function that displays an image in a frame.
- **"Main.java" file** which contains the class "ImageData" to store the pixels array and the label of a single image. It also contains the "Main" class which in turn contains "main" method that runs the whole program as shown below.

"Main" class:

```
public class Main {
    public static void main(String[] args) throws IOException {
        //1) Load Data
        File[] images= new File("Cats & Dogs Sample Dataset").listFiles();
        ImageData[] data = new ImageData[images.length];
        for (int i = 0; i < images.length; i++) {
            data[i].setPixels(ImageHandler.ImageToIntArray(images[i]));
            data[i].setLabel(images[i].getName().contains("cat")? 0 : 1);
        }

        //2) Shuffle
        List<ImageData> tempData = Arrays.asList(data);
        Collections.shuffle(tempData);
        tempData.toArray(data);
        tempData.clear();

        //3) Split the data into training (75%) and testing (25%) sets
        int[][] trainingSetFeatures, testingSetFeatures;
        int[] trainingSetLabels, testingSetLabels;
        /*
            ...
         */

        //4.1) Create the NN
        NeuralNetwork nn = new NeuralNetwork();
        //4.2) Set the NN architecture
        /*
            ...
         */

        //4.3) Train the NN
        nn.train(trainingSetFeatures, trainingSetLabels);

        //4.4) Test the model
        int[] predictedLabels = nn.predict(testingSetFeatures);
        double accuracy = nn.calculateAccuracy(predictedLabels,
                                               testingSetLabels);

        //4.5) Save the model (final weights)
        nn.save("model.txt");

        //5.1) Load the model and use it on an image
        NeuralNetwork nn2 = NeuralNetwork.load("model.txt");
        int[] sampleImgFeatures = ImageHandler.ImageToIntArray(
                                                    new File("sample.jpg"));
        int samplePrediction = nn2.predict(sampleImgFeatures);
        ImageHandler.showImage("sample.jpg");
        //5.2) Print "Cat" or "Dog"
        /*
            ...
         */
    }
}
```

As you can see, the first part of the "main" function loads the data using the "ImageHandler" and stores it as an array of "ImageData". The second part shuffles the data array to allow diversity when splitting the data. The remaining parts of the "main" method are not complete, so you are required to complete them as follows:

1. In the third part, you will need to **split the data** into training (75% of the data) and testing (25% of the data) sets. You will also need to separate the pixels (features) from the labels. This is because the neural network only uses the features as its input in the forward propagation step, while the labels are used as the actual outputs in the backpropagation.
2. For the fourth part, you will need to **implement the class "NeuralNetwork".** This class can have any needed attributes and it must have methods to:
    a. **set the architecture and hyperparameters of the NN** such as number of neurons in each layer, number of epochs, etc. (Use these methods in "main" part 4.2)
    b. **train the NN** for a number of epochs by performing forward and backward propagation on each training example.
    c. **predict the label** (0 or 1) of images by performing forward propagation using the final weights. (*Note:* This method could take a single training example or multiple training examples)
    d. **calculate the accuracy** of the NN model by comparing the predicted labels with the actual labels.
    e. **Save the weights to a file and load them from the file.**
3. In the fifth and final part (5.2), convert the predicted label to the actual class name and print it to the user.

Note: You can implement any classes you want (Layer, Neuron, etc.) to be used by the "NeuralNetwork" class. However, don't change the "Main" class.

Assignment submission notes:

- The **maximum** number of students in a team is **4** and the **minimum** is **3**.
- The **deadline will be announced, and no late submission** is allowed.
- Please submit **one compressed folder**. The folder name should follow this structure: ID1_ID2_ID3_GROUP
- **Cheating** students will take **negative grades** and no excuses will be accepted. If you have any problems during the submission, contact your TA but don't, under any circumstances, give your code to or take the code from your friends.

Grading Criteria: (7 marks)

Splitting the Data	0.75
"NeuralNetwork" Structure	0.5
Training	3
Getting Predictions	1.5
Accuracy	0.5
Reading & Writing the Weights	0.75

Good luck