**National University of Computer and Emerging Sciences**



**Lab Manual 08**

**CL461-Artificial Intelligence Lab**

Department of Computer Science

FAST-NU, Lahore, Pakistan

Table of Contents

* Objectives

After performing this lab, students shall be able to understand the following:

* Artificial Neural Network (ANN)
* Image classification using ANN with scikit-learn
* Task Distribution

| **Total Time** | **170 Minutes** |
| --- | --- |
| Artificial Neural Network (ANN) | 15 Minutes |
| Image classification using ANN with scikit-learn | 25 Minutes |
| Exercise | 120 Minutes |
| Online Submission | 10 Minutes |

* Artificial Neural Network (ANN)

An Artificial neural network is a computational network, based on biological neural networks that construct the structure of the human brain. Similar to how a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks. A typical single neuron (also known as perceptron) is represented below.



ANNs are designed to spot patterns in the data. This makes ANNs an optimal solution for classifying (sorting data into predetermined categories), clustering (finding like characteristics among data and pulling that data together into categories) and making predictions from data (such as helping determine infection rates for COVID, the next catastrophic weather event or box-office smash).

ANNs have an input layer and output layer. Between these two layers there are other hidden layers that perform the mathematical computations that help determine the decision or action the machine should take. Ultimately, these hidden layers are in place to transform the input data into something the output unit can use. A typical two layer NN is represented in the image below.



* **Back Propagation Algorithm**

It is the training or learning algorithm. It learns by example. If you submit to the algorithm the example of what you want the network to do, it changes the network’s weights so that it can produce desired output for a particular input on finishing the training.

The backprop algorithm cycles through two distinct passes, a forward pass followed by a backward pass through the layers of the network. The algorithm alternates between these passes several times as it scans the training data. Typically, the training data has to be scanned several times before the networks ”learns” to make good classifications.

**Forward Pass:** Computation of outputs of all the neurons in the network. The input is fed to the input layer, the neurons perform a linear transformation on this input using the weights and biases. Post that, an activation function is applied on this linear transformation to add non-linearity in the model in order to learn complex pattern from the data. Finally, the output from the activation function moves to the next hidden layer and the same process is repeated. This forward movement of information is known as the forward pass or propagation.



**Backward pass:** Propagation of error and adjustment of weights

The task is to make the output to the neural network as close to the actual (desired) output. Each of these neurons is contributing some error to the final output. How do you reduce the error? We try to minimize the value/weight of neurons that are contributing more to the error and this happens while traveling back to the neurons of the neural network and finding where the error lies. In order to minimize the error, the neural networks use a common algorithm known as “Gradient Descent”, which helps to optimize the task quickly and efficiently.

The one round of forwarding and backpropagation iteration is known as one training iteration aka “**Epoch**“. Most applications of feedforward networks and backprop require several epochs before errors are reasonably small.

* **Activation Functions**

Activation function is one of the building blocks of Neural Network. Popular types of activation functions are:

* Binary Step Function
* Linear Function
* Sigmoid
* Tanh
* Rectified Linear Unit (ReLU)
* Leaky ReLU
* Parameterised ReLU
* ELU
* Softmax

Refer to the following links to understand activation functions in details:

* <https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/>
* <https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0>
* <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>

Refer to the lecture for further details regarding ANNs.

* **Image Classification Using ANNs with Scikit-learn**

**Note:**

Find the complete scikit-learn user guide here: <https://scikit-learn.org/stable/user_guide.html>

Find the Multilayer Perceptron scikit-learn here: [https://scikit-learn.org/stable/modules/generated/sklearn.neural\_network.MLPClassifier.html#](https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html%23)

Refer to the Colab Notebook uploaded along this lab manual on Classroom.

Image classification is an amazing application of artificial intelligence. We can train a powerful algorithm to model a large image dataset. This model can then be used to classify a similar but unknown set of images.

We will build an Image classifier for the [Fashion-MNIST Dataset](https://github.com/zalandoresearch/fashion-mnist). The Fashion-MNIST dataset is a collection of [Zalando's](https://research.zalando.com/) article images. It contains 60,000 images for the training set and 10,000 images for the test set data (*we will discuss the test and training datasets along with the validation dataset later*). These are 28 x 28 grayscale images and belong to the labels of 10 different classes.

* **Labels**

Each training and test example is assigned to one of the following labels:

| **Label** | **Description** |
| --- | --- |
| 0 | T-shirt/top |
| 1 | Trouser |
| 2 | Pullover |
| 3 | Dress |
| 4 | Coat |
| 5 | Sandal |
| 6 | Shirt |
| 7 | Sneaker |
| 8 | Bag |
| 9 | Ankle boot |

* **Classification Evaluation Metric**

Learn more about Confusion matrix, Precision, Recall and Accuracy here: <https://medium.com/@kennymiyasato/classification-report-precision-recall-f1-score-accuracy-16a245a437a5>

* **Exercise (50 Marks)**
* Improve the accuracy of the Fashion-MNIST Neural Network model given in the Colab Notebook.
* Perform hyper parameter tuning and k-fold cross validation using GridSearchCV: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>
* **Submission Instructions**

Always read the submission instructions carefully.

* Rename your Jupyter notebook to your roll number and download the notebook as **.ipynb** extension.
* To download the required file, go to **File->Download .ipynb**
* Only submit the **.ipynb** file. DO NOT **zip** or **rar** your submission file.
* Submit this file on Google Classroom under the relevant assignment.
* Late submissions will not be accepted.