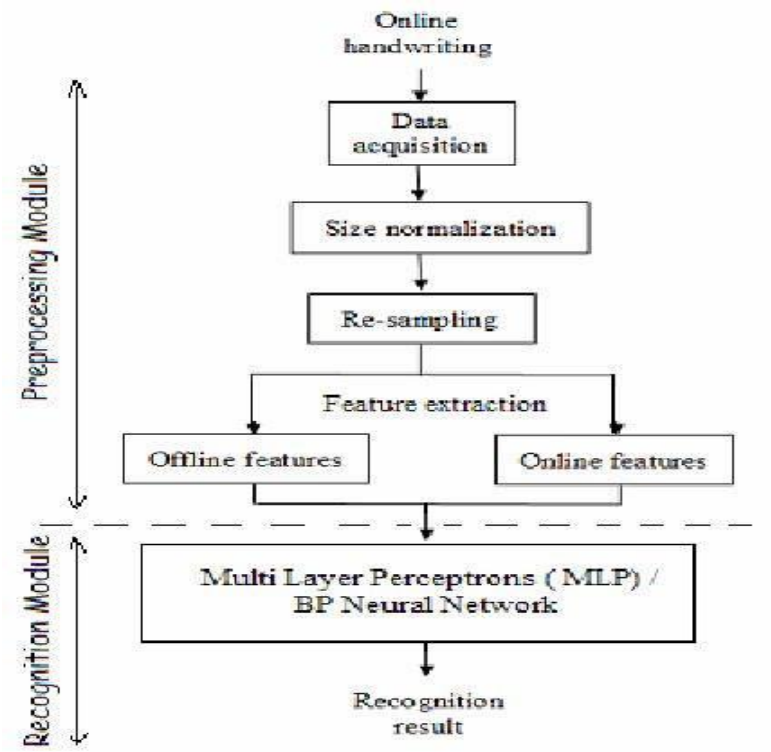
Project Design Phase-I
Solution Architecture

Date	**19 September 2022**
Team ID	**PNT2022TMIDxxxxxx**
Project Name Project	**A Novel Method for Handwritten Digit Recognition System**
Maximum Marks	**4 Marks**

Project Description :

Handwritten digit recognition is one of the important problems in computer vision these days. There is a great interest in this field because of many potential applications, most importantly where a large number of documents must be dealt such as post mail sorting, bank cheque analysis, handwritten form processing etc. So a system should be designed in such a way that it is capable of reading handwritten digits and providing appropriate results. We propose a solution on neural network approaches to recognize handwritten digits.

Technical Architecture :



Working :

Dataset used :

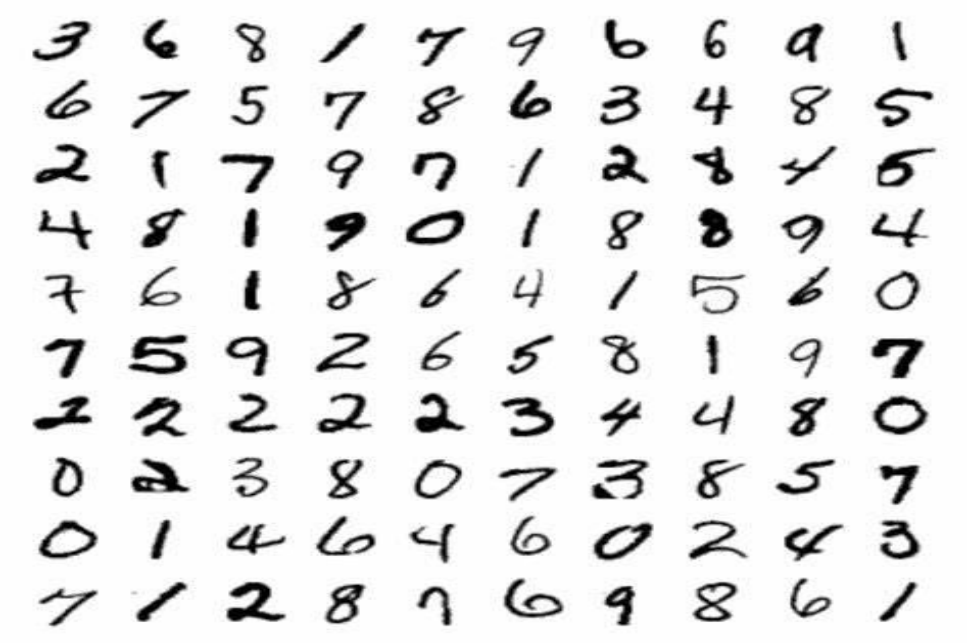
MNIST ("Modified National Institute of Standards and Technology") is the “Hello World” dataset of computer vision. Since its release in 1999, this classic dataset of handwritten images has served as the basis for benchmarking classification algorithms. As new machine learning techniques emerge, MNIST remains a reliable resource for researchers and learners alike.

In this competition, we aim to correctly identify digits from a dataset of tens of thousands of handwritten images. Kaggle has curated a set of tutorial-style kernels which cover everything from regression to neural networks. They hope to encourage us to experiment with different algorithms to learn first-hand what works well and how techniques compare.

Dataset description :

For this competition, we will be using Keras (with TensorFlow as our backend) as the main package to create a simple neural network to predict, as accurately as we can, digits from handwritten images. In particular, we will be calling the Functional Model API of Keras, and creating a 4-layered and 5-layered neural network.

The MNIST Handwritten Digit Recognition Dataset contains 60,000 training and 10,000 testing labeled handwritten digit pictures. Each picture is 28 pixels in height and 28 pixels wide, for a total of 784 (28×28)pixels. Each pixel has a single pixel value associated with it. It indicates how bright or dark that pixel is (larger numbers indicate darker pixel). This pixel value is an integer ranging from 0 to 255.



Procedure :

- Install the latest TensorFlow library.
- Prepare the dataset for the model.

- Develop Single Layer Perceptron model for classifying the handwritten digits. Plot the change in accuracy per epochs.
- Evaluate the model on the testing data.
- Analyze the model summary.
- Add a hidden layer to the model to make it a Multi-Layer Perceptron.
- Add Dropout to prevent overfitting and check its effect on accuracy.
- Increasing the number of Hidden Layer neurons and checking its effect on accuracy.
- Use different optimizers and check its effect on accuracy.
- Increase the hidden layers and check its effect on accuracy.
- Manipulate the batch size and epochs and check its effect on accuracy.

Handwritten digit recognition using MNIST dataset is a major project made with the help of Neural Network. It basically detects the scanned images of handwritten digits. We have taken this a step further where our handwritten digit recognition system not only detects scanned images of handwritten digits but also allows writing digits on the screen with the help of an integrated GUI for recognition.

Approach :

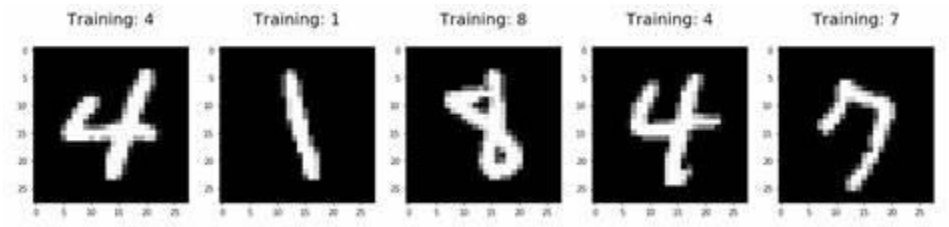
For this competition, we will be using Keras (with TensorFlow as our backend) as the main package to create a simple neural network to predict, as accurately as we can, digits from handwritten images. In particular, we will be calling the Functional Model API of Keras, and creating a 4-layered and 5-layered neural network.

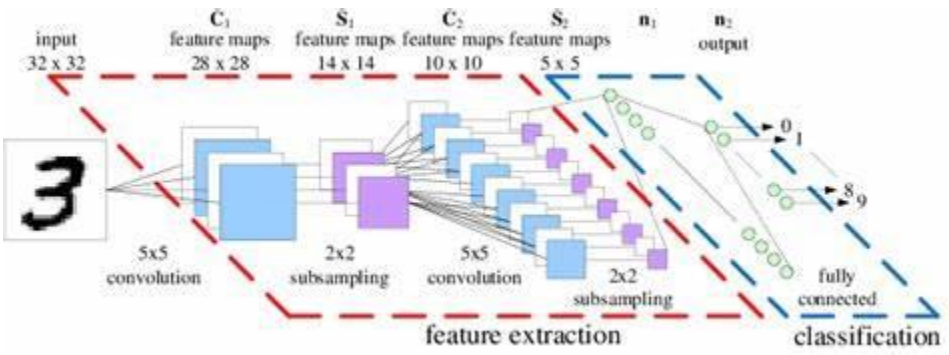
Also, we will be experimenting with various optimizers: the plain vanilla Stochastic Gradient Descent optimizer and the Adam optimizer. However, there are many other parameters, such as training epochs which we will not be experimenting with.

In addition, the choice of hidden layer units are completely arbitrary and may not be optimal. This is yet another parameter which we will not attempt to tinker with. Lastly, we introduce dropout, a form of regularization, in our neural networks to prevent overfitting.

Methodology :

A neural network with one hidden layer and 100 activation units has been put into practise (excluding bias units). The features (X) and labels (Y) were retrieved after the data was loaded from a.mat file. To prevent overflow during computation, features are then scaled into a range of [0,1] by dividing by 255. 10,000 testing cases and 60,000 training examples make up the data. With the training set, feedforward is used to calculate the hypothesis, and backpropagation is then used to lower the error between the layers. To combat overfitting, the regularization parameter lambda is set to 0.1. To identify the model that fits the situation the optimizer runs for 70 times.





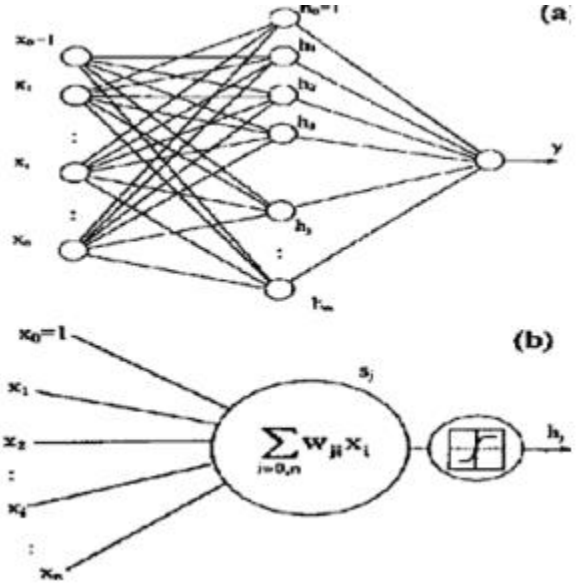
Algorithm used :

Multi-Layer Feed-forward Neural Networks.

Multi-Layer Feed-forward neural networks (FFNN) have high performances in input and output function approximation. In a three-layer FFNN, the first layer

connects the input variables. This layer is called the input layer. The last layer connects the output variables. This layer is called the output layer. Layers between the input and output layers are called hidden layers. In a system, there can be more than one hidden layer. The processing unit elements are called nodes. Each of these nodes is connected to the nodes of neighboring layers.

The parameters associated with node connections are called weights. All connections are feed forward; therefore they allow information transfer from the previous layer to the next consecutive layers only. For example, the node j receives incoming signals from node i in the previous layer. Each incoming signal is a weight. The effective incoming signal to node j is the weighted sum of all incoming signals. The following figures are an example of a usual FFNN and nodes.



Working :

Neural Networks receive an input and transform it through a series of hidden layers. Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer. Neurons in a single layer function completely independently. The last fully connected layer is called the "output layer“.

Tensor flow :

TensorFlow is an open-source machine learning library for research and production. TensorFlow offers APIs for beginners and experts to develop for desktop, mobile, web, and cloud. See the sections below to get started. By scanning the numerical digit and converting it into png format using the python3 command in the terminal we can get text output and sound output.

Feature extraction :

All neurons in a feature share the same weights .In this way all neurons detect the same feature at different positions in the input image. Reduce the number of free parameters.

Classification :

Convolutional neural network that is very popular for computer vision tasks like image classification, object detection, image segmentation and a lot more. Image classification is one of the most needed techniques in today's era, it is used in various domains like healthcare, business, and a lot more.

Result :

We do not consider our results to be flawless, as with any study or project undertaken in the field of machine learning and image processing. There is always opportunity for improvement in your methods because machine learning is a topic that is continually developing; there will always be a fresh new idea that solves a given problem more effectively. Three models were used to test the application: Multi-Layer Perceptron (MLP), Convolution NeuralNetwork, and (CNN). We obtain a different classifier accuracy with each model, indicating which is superior.