Handwritten Digit Recognition using Machine Learning

---***---

Abstract - *People write in as many different ways as there are stars in a galaxy. This leads to development of different patterns in writing. Costly manual labour is required to do a mundane and tedious job of converting the physical written data and information into digital form for storing it in a digital form. This project discusses the solution to a part of problem as we have limited the scope to only the hand written digits (0-9). We have trained a model using deep neural networks for digit recognition using Google's Machine Learning tool TensorFlow and Python Programming language. We have used the 'MNIST DATABASE' which consist of training and test set for hand written digits (0-9) of size (28x28) pixels i.e. 784 pixels. The data set consist of 60,000 training data and 10,000 test data. The limitation of this model will be if digits other than (0-9) are given then the model will not be able to recognize and classify it and the model will be able to predict numbers only in black and white images.*

Key Words: **Machine Learning, Computer Vision, K-Nearest Neighbour, Support Vector Machine, Neural Networks**

1. INTRODUCTION

The advancements in Computer Vision with Deep Learning has been constructed and perfected with time, primarily over one particular algorithm- a Convolutional Neural Network. It is a deep learning algorithm which can take in an input image, assign important (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The accuracies in these fields including handwritten digits recognition using Deep Convolutional Neural Networks (CNNs) have reached human level perfection. The CNN architecture is inspired by the mammalian visual system. Cells in the cat's visual cortex are sensitized to a tiny area of the receptive field. It was found by D. H. Hubel et al. in 1962. The neocognitron, the pattern recognition model inspired by the work of D. H. Hubel et al. was the first computer vision. It was introduced by Fukushima in 1980. In 1998, the framework of CNNs as designed by LeCun et al. which had seven layers of Convolutional neural networks. It was adept in handwritten digits classification direct from pixel values of images. Gradient descent and back propagation algorithm is used for training the model. In handwritten recognition digits, characters are given as input. The model can be recognized by the system. A simple artificial neural network (ANN) has an input layer, an output layer and some hidden layers between the input and output layer. CNN has a very similar architecture as ANN. There are several neurons in each layer in ANN. The weighted sum of all the neurons of a layer becomes the input of a neuron of the next layer adding a biased value. In CNN the layer has three dimensions. Here all the neurons are not fully connected. Instead, every neuron in the layer is connected to the local receptive field. A cost function generates in order to train the network. It compares the output of the network with the desired output. The signal propagates back to the system, again and again, to update the shared weights and biases in all the receptive fields to minimize the value of cost function which increases the network's performance. We have applied a different type of Convolutional Neural Network algorithm on Modified National Institute of Standards and Technology (MNIST) dataset using TensorFlow, a Neural Network library written in python.

2. LITERARY SURVEY

CNN is playing an important role in many sectors like image processing. It has powerful impact on many fields. Even, in nano-technologies like manufacturing semiconductors, CNN is used for fault detection and classification. Handwritten digit recognition has become an issue of interest among researchers. There are a large number of papers and articles being published about this topic. In research, it is shown that Deep Learning algorithms like multilayer CNN using Keras with Theano and TensorFlow gives the highest accuracy in comparison with the most widely used machine learning algorithms like SVM, KNN & RFC. Because of its highest accuracy, Convolutional Neural Network (CNN) is being used on a large scale in image classification, video analysis, etc. Many researchers are trying to make sentiment recognition in a sentence. CNN is being used in natural language processing and sentiment recognition by varying different parameters. It is pretty challenging to get a good performance as more parameters are needed for the largescale neural network. Many researchers are trying to increase the accuracy with less error in CNN. In another research, they have shown that deep nets perform better when they are trained by simple back-propagation. Their architecture results in the lowest error rate on MNIST

compare to NORB and CIFAR10. Researchers are working on this issue to reduce the error rate as much as possible in handwriting recognition. In one research, an error rate of 1.19% is achieved using 3-NN trained and tested on MNIST. Deep CNN can be adjustable with the input image noise. Coherence Recurrent Convolutional Network (CRCN) is a multimodal neural architecture. It is being used in recovering sentences in an image. Some researchers are trying to come up with new techniques to avoid drawbacks of traditional convolutional layer's. Ncfm (No combination of feature maps) is a technique which can be applied for better performance using MNIST datasets. Its accuracy is 99.81% and it can be applied for large-scale data. New applications of CNN are developing day by day with many kinds of research. Researchers are trying hard to minimize error rates. Using MNIST datasets and CIFAR, error rates are being observed. To clean blur images CNN is being used. For this purpose, a new model was proposed using MNIST dataset. This approach reaches an accuracy of 98% and loss range 0.1% to 8.5%. In Germany, a traffic sign recognition model of CNN is suggested. It proposed a faster performance with 99.65% accuracy. Loss function was designed, which is applicable for light-weighted 1D and 2D CNN. In this case, the accuracies were 93% and 91% respectively.

3. EXISTING MODELS

In this section we are going to discuss few algorithms which are used so far for hand written digit recognition. Mentioned below are the algorithms:
a. KNN (K nearest Neighbors)
b. SVM (Support Vector Machine)
c. NN (Neural Networks)

3.1 KNN (K nearest Neighbors)

KNN is the non-parametric method or classifier used for classification as well as regression problems. This is the lazy or late learning classification algorithm where all of the computations are derived until the last stage of classification, as well as this, is the instance-based learning algorithms where the approximation takes place locally. Being simplest and easiest to implement there is no explicit training phase earlier and the algorithm does not perform any generalization of training data. KNN explains categorical value using majority votes of K nearest neighbours where the value for K can differ, so on changing the value of K, the value of votes can also vary.

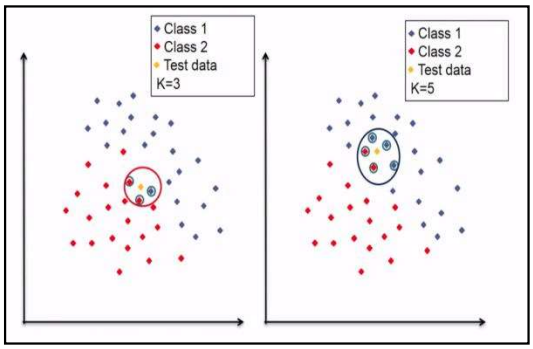

Fig- 1: KNN Example

3.2 SVM (Support Vector Machine)

SVM falls into the category of supervised learning, and with the bonus of classification as well as regression problems. Generally, SVM draws an optimal hyper plane which classifies into different categories. In two dimensional spaces, To start with we plot the data points of the independent variable corresponding to the dependent variables. Then, begin the classification process from looking the hyper plane or any linear or nonlinear plane differentiated the two classes at its best.

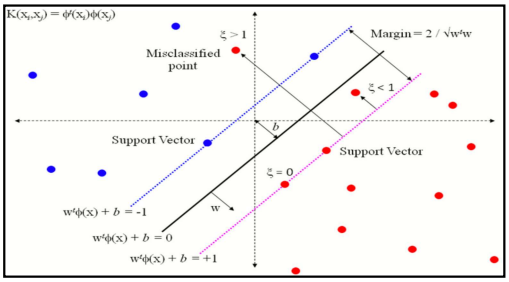

Fig- 2: SVM

3.3 NN (Neural Networks)

Neural Networks mimics the working of how our brain works. They have emerged a lot in the era of advancements in computational power. Deep learning is the acronym for Neural Networks, the network connected with multilayer. The layers are composited form nodes. A node is just a perception which takes an input performs some computation and then passed through a node's activation function, to show that up to what context signal progress proceeds through the network to perform classification.

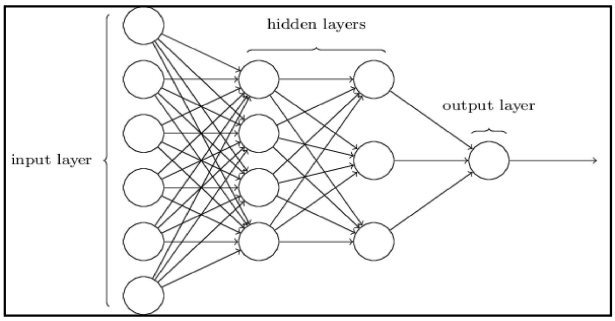

Fig- 3: Neural Network

Table -1: Overall comparison results:

Parameters	KNN	SVM	NN
Accuracy (Test images)	96.67	97.91	3.15 (Error)
Accuracy (Training Images)	97.88	99.91	2.17 (Error)
Time (Training) approx.	25 min.	19 min.	35 min
Time (Testing) approx.	15 min.	11 min.	20 min

4. METHODOLOGY ADOPTED

In this project we will be using the TensorFlow library to build, train and test our models and the data which we will use is MNIST database. The algorithm used will be Convolutional Neural Network (CNN).

4.1 TensorFlow

TensorFlow is a free and open-source library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and also used in machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by Google Brain team for internal Google use. It was released under the Apache License 2.0 on November 9, 2015.

TensorFlow offers multiple levels of abstraction so we can choose the right one for our needs. We can build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

If we need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, we can use the Distribution Strategy API for distributed training on different hardware configurations without changing the model definition.

4.2 MNIST Dataset

The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning.

The MNIST database contains 60,000 training images and 10,000 testing images. Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset. There have been a number of scientific papers on attempts to achieve the lowest error rate.

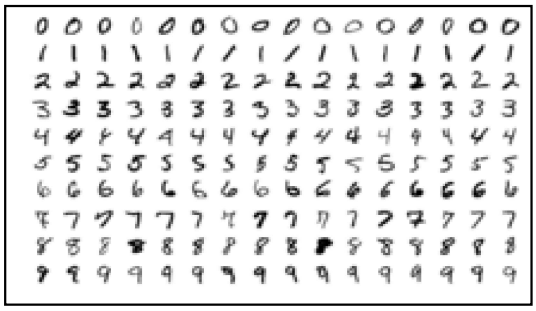

Fig- 4: Sample images from MNIST test dataset

4.3 The Algorithm: CNN

To recognize the handwritten digits, a seven-layered convolutional neural network with one input layer followed by five hidden layers and one output layer is designed.

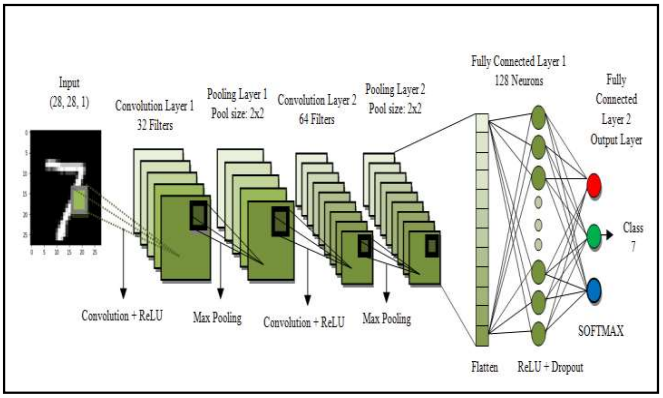

Fig- 5: A seven-layered convolutional neural network for digit recognition

The input layer consists of 28 by 28 pixel images which mean that the network contains 784 neurons as input data. The input pixels are greyscale with a value 0 for a white pixel and 1 for a black pixel. The proposed model has five hidden layers. The first hidden layer is the

convolution layer 1 which is responsible for feature extraction from an input data. This layer performs convolution operation to small localized areas by convolving a filter with the previous layer. In addition, it consists of multiple feature maps with learnable kernels and rectified linear units (ReLU).

The kernel size determines the locality of the filters. ReLU is used as an activation function at the end of each convolution layer as well as a fully connected layer to enhance the performance of the model. The next hidden layer is the pooling layer 1. It reduces the output information from the convolution layer and reduces the number of parameters and computational complexity of the model. Here, max pooling is used to subsample the dimension of each feature map. Convolution layer 2 and pooling layer 2 which has the same function as convolution layer 1 and pooling layer 1 and operates in the same way except for their feature maps and kernel size varies. A Flatten layer is used after the pooling layer which converts the 2D featured map matrix to a 1D feature vector and allows the output to get handled by the fully connected layers. A fully connected layer is another hidden layer also known as the dense layer. It is similar to the hidden layer of Artificial Neural Networks (ANNs) but here it is fully connected and connects every neuron from the previous layer to the next layer. In order to reduce overfitting, dropout regularization method is used at fully connected layer 1. It randomly switches off some neurons during training to improve the performance of the network by making it more robust. This causes the network to become capable of better generalization and less compelling to overfit the training data. The output layer of the network consists of ten neurons and determines the digits numbered from 0 to 9. Since the output layer uses an activation function such as softmax, which is used to enhance the performance of the model, classifies the output digit from 0 through 9 which has the highest activation value.

5. EXPERIMENTAL SETUP

In this project we will use Google Colab (also known as Colaboratory) which is a free Jupyter notebook environment that runs in the cloud and stores its notebooks on Google Drive. Colaboratory started as a part of Project Jupyter, but the development was eventually taken over by Google.

We will use Python 3 as the runtime type and GPU as the hardware accelerator in Colab. TensorFlow 2.0 will be used throughout to build, train and test models.

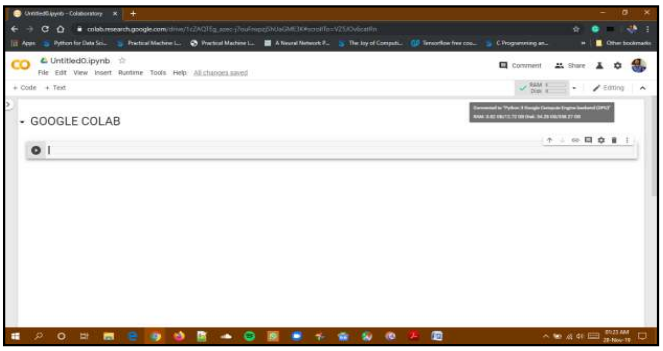

Fig- 6: Google Colab

6. RESULTS AND DISCUSSIONS

In the above section CNN has been applied on the MNIST dataset in order to observe the variation of accuracies for handwritten digits. The accuracies are obtained using TensorFlow in python. Training and validation accuracy for 15 different epochs were observed by taking the batch size 100.

For 5 hidden layers, with the batch size of 100 and 15 epochs, the minimum training accuracy was found to be 93.43% and the minimum validation accuracy was found to be 97.89%, both at epoch 1. For the same setup, the maximum training was found to be 99.77% at epoch 13 and the maximum validation accuracy was found to be 99.12% at epoch 15.

During testing, the model performed with the accuracy of 99.16% over the 10000 test images.

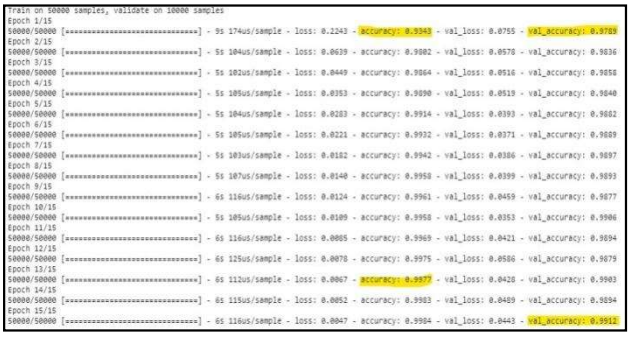

Fig- 7: Training and Validation Accuracies

7. CONCLUSIONS AND FUTURE WORKS

In this project we build a convolutional neural network with a total of 7 layers to classify the handwritten digits. The layers were in the order of conv -> max_pool-> conv -> max_pool -> dense -> dense. We use MNIST dataset to train the model. We found out that our model outperformed the previously existing models (ANN, KNN and SVM) as we got an accuracy of 99.12% while testing.

In future we plan to observe the variation in the overall classification accuracy by varying the number of hidden layers and batch size.

REFERENCES

[1]. Hand Written Digit Recognition Using TensorFlow and Python; Shekhar Shiroor Department of Computer Science College of Engineering and Computer Science California State University-Sacramento Sacramento, CA 95819-6021, USA

[2]. Analogizing Time Complexity of KNN and CNN in Recognizing Handwritten Digits; Tanya Makkar & Yogesh Kumar Department of ECE, Amity School of Engg. and Technology, Amity University Uttar Pradesh, Noida, India; Ashwani Kr Dubey Department of ECE, Amity School of Engg. and Technology, Amity University Uttar Pradesh, Noida, India; Álvaro Rocha Department of Informatics Engineering, University of Coimbra Portugal; Ayush Goyal Electronics & Communication Engineering, Texas A&M University – Kingsville

[3]. Handwritten Digits Recognition with Artificial Neural Network; Kh Tohidul Islam, Ghulam Mujtaba, Dr. Ram Gopal Raj, Henry Friday Nweke

[5]. https://www.tensorflow.org/