HANDWRITTEN CHARACTER RECOGNITION USING DEEP LEARNING & MACHINE LEARNING

Maitreyi Pitale & Arish Dakhani
December 12, 2021

Abstract

We often see in offices, banks etc. that we still continue with manual entries. When we rely on humans, mistakes are bound to happen. One of the most significant downsides of manual entry is Human error. Handwritten transactions are less secure as compared to an Intelligent system. To manage this time-consuming process, we propose Handwritten Character Recognition using Deep Neural Networks and algorithms from Machine Learning. In today's world, it has become easier to train deep neural networks because of the availability of huge amounts of data and various algorithmic innovations which are taking place. Now-a-days the amount of computational power needed to train a neural network has increased due to the availability of GPU's and other cloud based services like Google Cloud platform and Amazon Web Services which provide resources to train a Neural network on the cloud. We have two types of datasets for training our Machine Learning and Convolution Networks Model. In terms of Machine Learning, we have used K-Nearest Neighbors and Multilayer Perceptron. In terms of Deep Learning, we have used Convolution Neural Networks. The results were then compared using graphs accordingly, the best performance was shown by CNN. We have developed this system using the python programming language.

1 INTRODUCTION

To make machines more intelligent, the developers are diving into machine learning and deep learning techniques. A human learns to perform a task by practicing and repeating it again and again so that it memorizes how to perform the tasks. Then the neurons in his brain automatically trigger and they can quickly perform the task they have learned. Deep learning is also very similar to this. It uses different types of neural network architectures for different types of problems. For example – object recognition, image and sound classification, object detection, image segmentation, etc.Handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image. We have used one of the most popular datasets This is probably one of the most popular datasets among machine learning and deep learning enthusiasts. The MNIST dataset contains 60,000 training images of handwritten digits from zero to nine and 10,000 images for testing. So, the MNIST dataset has 10 different classes. The handwritten digits images are represented as a 28×28 matrix where each cell contains grayscale pixel value. The second dataset is the text dataset containing values 0 1 which make a whole number. We converted the text dataset into png dataset for implementation of KNN and MLP.In the end, we are going to build a GUI in which you can upload the handwritten data and predict the digit. We will further discuss the specifications in the report.

2 PROBLEM STATEMENT

The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual's handwriting. The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

3 LITERATURE REVIEW

Being a well-studied classification problem, multiple machine learning techniques such as regression, KNN, MLP and Neural Networks have been investigated to recognize digits with maximum accuracy. To reach this conclusion, we have undergone extensive paper research whose literature review is stated in this section. Immense research is going on in the field of handwritten character recognition.

Many people have developed systems for handwritten character recognition. We have studied some of the systems: A character recognition system has been designed using fuzzy logic. The system developed by them can be created on a VLSI structure. Their character recognition system is immune to distortion and variations in shift. They have made use of hamming neural networks in their system. An innovative method for recognition of handwritten Tamil characters using Neural Networks has been developed. They have

made use of Kohonen Self Organizing Map(SOM) which is an unsupervised neural network. The system developed by them can be used for recognition of tamil characters as well Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies IEEE 772 Authorized licensed use limited to: Middlesex University. Downloaded on September 01,2020 from IEEE Xplore. Restrictions do apply as for the recognition of other Indic languages. Their system produces near accurate results but sometimes produces errors if the handwritten characters are not properly segmented. One of the Authors has presented a unique method for authenticating a person based on their handwriting[3]. The author has used the Multi layer feed forward neural network in their system. The author has proposed in this paper that the height and width of a handwritten alphabet is unique for each and every person. The author has presented a method for recognition and identification of a person from their handwriting. A novel method for handwritten character recognition has been designed which does not use feature extraction.. They have implemented their system in Matlab. Their system uses a feed forward neural network with backpropagation. One of the authors has proposed a unique method for handwriting recognition. Their system uses a Self Organizing Map for feature extraction. They have used a Recurrent neural network for learning. They conducted their experiment on recognition of Japanese characters.

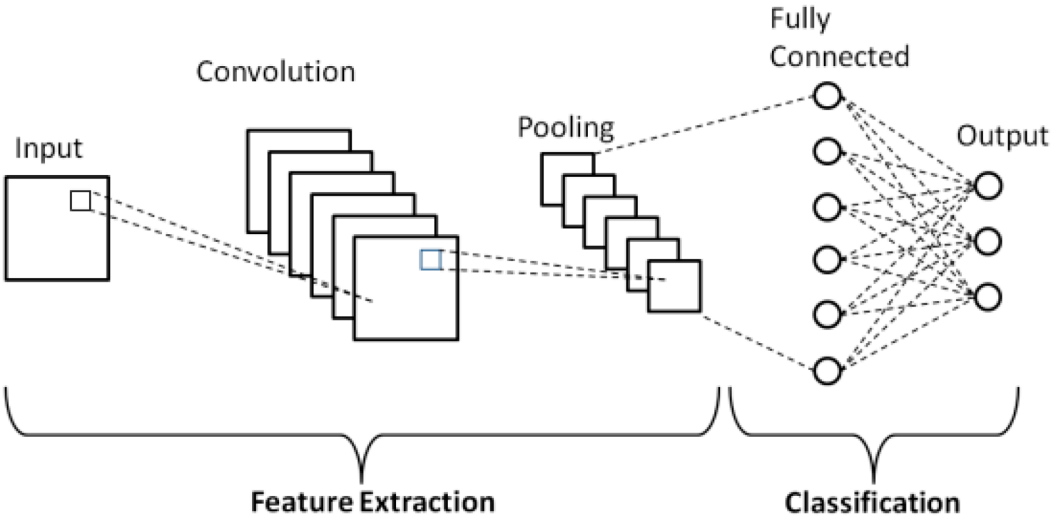
In the Springer published paper 'Data Classification with Deep Learning using Tensorflow' by Fatih Erham & Galip Aydin they have studied classification using the Tensorflow library. Tensorflow is an open-source software library developed by Google for numerical computation, which is now widely used by many large companies. Tensorflow provides an interface for expressing machine learning algorithms and an application for executing these algorithms.. A calculation expressed using TensorFlow can be carried out with little or no modification in a wide range of heterogeneous systems, from mobile devices such as phones and tablets, to large scale distributed systems of hundreds of machines, and to various computing devices such as GPU cards. In this paper , they have used a unique library for classification of dataset which is the Softmax classifier.Softmax is a classifier and function that is often used in deep learning applications . With Softmax, the data can be categorized by direct classifier.

6 METHODS & TECHNIQUES

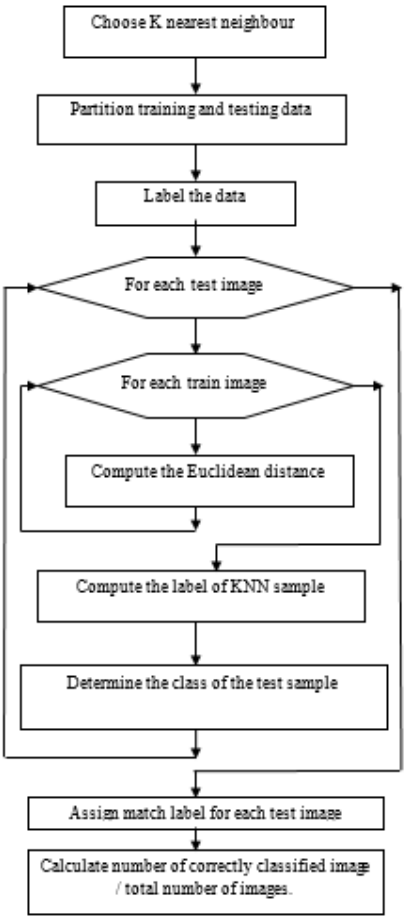
The first and foremost part for developing an intelligent system is the choice of libraries and dataset. We imported all the modules needed to train the model. The Keras library already contains some dataset and MNIST is one them so we could directly import it in.The mnist.load_data() returns us the training data and its labels. The next part of importing libraries is for developing the CNN model. The libraries imported for CNN are Keras, Keras.model, Dense, Dropout,Fallen, Cov2D, Maxpooling2D and test_train_split.

1) Pre-processing: This is the first step performed in image processing. In this step the noise from the image is removed by using median filtering. Median filtering is one of the most widely used noise reduction technique. This is because in median filtering the edges in image are preserved while the noise is still removed. 2) Conversion to Gray-Scale: The original image is black and white but the value ranges from [0-255]. 0 being the absolute black and 255 being absolute white. But for training, we have to convert this into [0-1].CNN works if we input a 4D array.So input data has a shape of (batch_size, height, width, depth), where the first dimension represents the batch size of the image and the other three dimensions represent dimensions of the image which are height, width, and depth. Depth of the image is the number of color channels. For example, an RGB image has a depth of 3, but a grayscale image has a depth of 1. We may add a depth of 1 to the greyscale photos. The picture is transformed to grayscale after the preprocessing stage. Conversion to grayscale is required because various authors use different colored pens with variable intensities. Working with grayscale photos also minimizes the system's overall complexity. For example, an RGB image has a depth of 3, but a grayscale image has a depth of 1. We may add a depth of 1 to the greyscale photos. The picture is transformed to grayscale after the preprocessing stage. Conversion to grayscale is required because various authors use different colored pens with variable intensities. Working with grayscale photos also minimizes the system's overall complexity.

4) Image Segmentation: A user can write text in the form of lines. Thus the thresholded image is first segmented into individual lines. Then each individual line is segmented into individual words. Finally each word is segmented into individual characters.Segmentation of image into lines is carried out using Horizontal projection method. First the thresholded image is inverted so that background becomes foreground and vice-versa. Now the image is scanned from top to bottom. While scanning , the sum of pixels in each row of image is calculated. The sum of pixels will be zero if all the pixels in one particular row are black. The sum will be non-zero if some white pixels are present in a row. After this a horizontal histogram is plotted in which the X-axis represents the Y-coordinate of image(Starting from Top to Bottom) and the Y-axis represents the sum of pixels in the row corresponding to the Y-coordinate. The horizontal histogram is plotted using MatPlotLib..

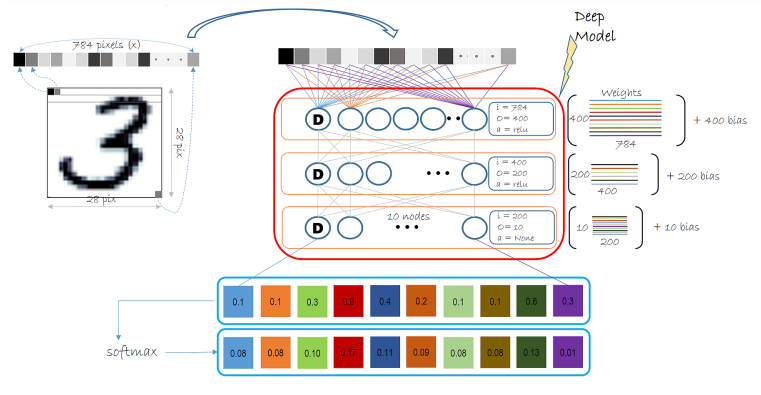


In accordance with the figure above creating a Deep Neural Network includes following steps:
Sequential - A feedforward neural network
Dense - A typical layer in our model
Dropout - Is used to make the neural network more robust, by reducing overfitting
Flatten - It is used to flatten the data for use in the dense layer
Conv2d - We will be using a 2-Dimensional CNN
MaxPooling2D- Pooling mainly helps in extracting sharp and smooth features. It is also done to reduce variance and computations. Max-pooling helps in extracting low level features like edges, points, etc. While Avg-pooling goes for smooth features. We have also used KNN and Multilayer Perceptron.The flowchart describes the series of our implementation using text dataset which we convert into png.

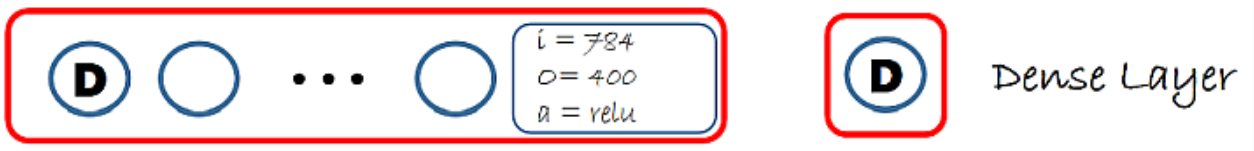


Model Creation of Multilayer Perceptron:

Our multi-layer perceptron will be relatively simple with 2 hidden layers . The number of nodes in the hidden layer being a parameter specified by Hidden Layers Dist. The figure below illustrates the entire model in the context of MNIST data.



Each Dense layer shows the input dimensions, output dimensions and activation function that layer uses. Specifically, the layer below shows: input dimension = 784 (1 dimension for each input pixel), output dimension = 400 (number of hidden nodes, a parameter specified by the user) and activation function being relu.



In this model we have 2 dense layers called the hidden layers each with an activation function of relu and one output layer with no activation.The output dimension (a.k.a. number of hidden nodes) in the 2 hidden layers is set to 400 and 200 in the illustration above. In the code below we keep both layers to have the same number of hidden nodes (set to 400). The number of hidden layers is 2. Fill in the following values: - num_hidden_layers - hidden_layers_dim. The final output layer emits a vector of 10 values. Since we will be using softmax to normalize the output of the model we do not use an activation function in this layer. The softmax operation comes bundled with the loss function

7 DISCUSSION AND RESULTS

(i) DATASETS & STATISTICS

We have used Mnist Dataset for training our Convolution Neural Networks model. The MNIST dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset. The dataset is provided in two file formats. Both versions of the dataset contain identical information, and are provided entirely for the sake of convenience. The second version of the dataset is provided in the same binary format as the original MNIST dataset. The EMNIST Balanced dataset contains a set of characters with an equal number of samples per class. The EMNIST Letters dataset merges a balanced set of the uppercase and lowercase letters into a single 26-class task. The MNIST Digits dataset provides balanced handwritten digit data sets directly compatible with the original MNIST dataset. The MNIST Digits dataset provides balanced handwritten digit data sets directly compatible with the original MNIST dataset.

The data is stored in a very simple file format designed for storing vectors and multidimensional matrices. General info on this format is given at the end of this page, but you don't need to read that to use the data files. All the integers in the files are stored in the MSB first (high endian) format used by most non-Intel processors. Users of Intel processors and other low-endian machines must flip the bytes of the header.

There are 4 files:

train-images-idx3-ubyte: training set images

train-labels-idx1-ubyte: training set labels

t10k-images-idx3-ubyte: test set images

t10k-labels-idx1-ubyte: test set labels

The training set contains 60000 examples, and the test set 10000 examples.The first 5000 examples of the test set are taken from the original NIST training set. The last 5000 are taken from the original NIST test set. The first 5000 are cleaner and easier than the last 5000. This dataset is a huge dataset but contains a lot of noisy data and redundancies which need to be removed to get the important features from the data.To remove the redundancies , we need to preprocess the dataset in such a way that it benefits the CNN.

The second dataset is the text dataset which we got from Kaggle Datasets. It consists of a 2000 text dataset of numbers ranging from 0-9.We encoded text or plain text files to a grayscale image to be easily shared. Each pixel represents a single character's decimal value. When decoding an image you can decode text straight to the console or to a plain text file.Images use a PNG file extension. The image size will automatically be set depending on the text length to be encoded. Text should use a character encoding scheme using 8 bits or less until additional functionality is added to support UTF-8 (16 bit). If a character's decimal value is greater than the limit then it will be divided by the limit and the new value used. When the character value equals the limit value the new value will be 1.The limit value passed using either command line argument -l(–limit) specifies the decimal value limit for pixel values starting from 1. The default is 256 allowing for numbers from 0 to 255 (i.e. 8 bit pixels). If the limit value is greater than 8 bits then the value will still be wrapped around since the output image is grayscale

(ii) EVALUATION METRICS

To evaluate the best accuracy , we have traced different algorithms from both the Machine Learning and Deep Learning domain. The three algorithms used are K-Nearest Neighbors, Multilayer Perceptron and Convolution Neural Networks. To evaluate the results we have used the following evaluation metrics: Mean Absolute Error, Mean Absolute Percentage Error, & Accuracy.

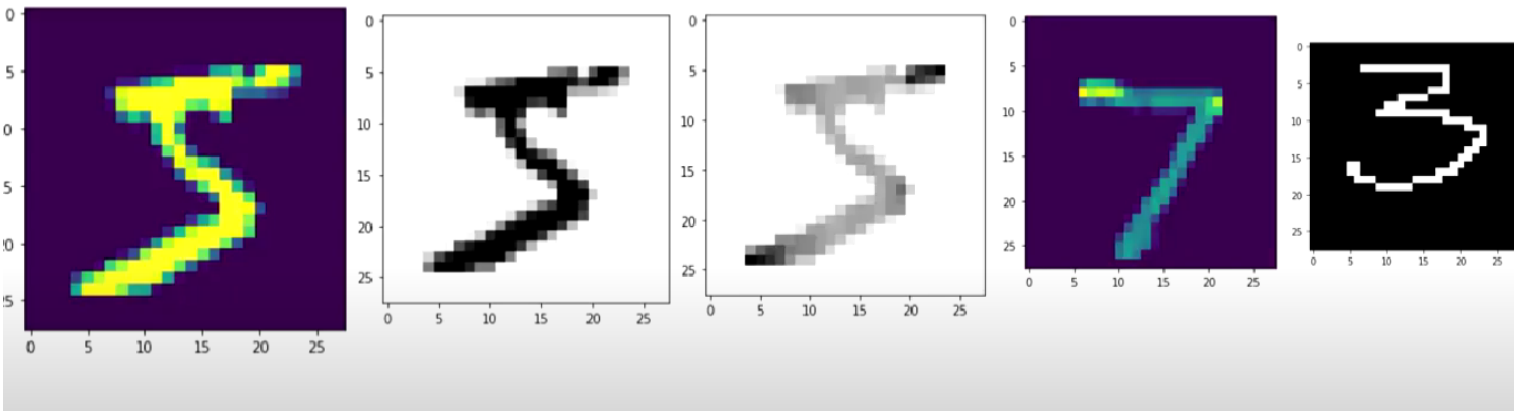
Metrics:MAE is a model evaluation metric often used with regression models. Given any test data-set, Mean Absolute Error of your model refers to the mean of the absolute values of each prediction error on all instances of the test data-set. Prediction error is the difference between the actual value and the predicted value for that instance. *Statistically, Mean Absolute Error* (MAE) refers to a the results of measuring the difference between two continuous variables.

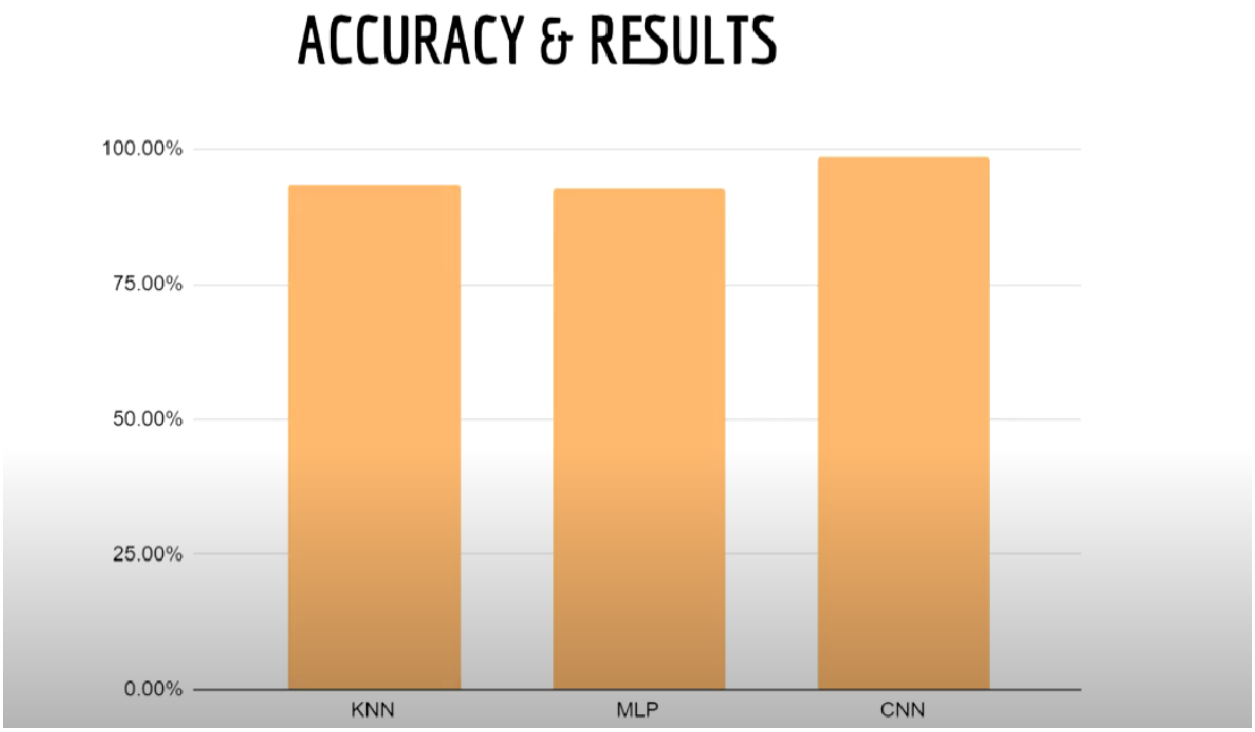
Prediction Error → Actual Value - Predicted Value

This prediction error is taken for each record after which we convert all error to positive. This is achieved by taking Absolute value for each error as below; Absolute Error → |Prediction Error| Finally we calculate the mean for all recorded absolute errors (Average sum of all absolute errors). MAE = Average of All absolute errors. **Mean Absolute Percentage Error (MAPE)** is a statistical measure to define the accuracy of a machine learning algorithm on a particular dataset.

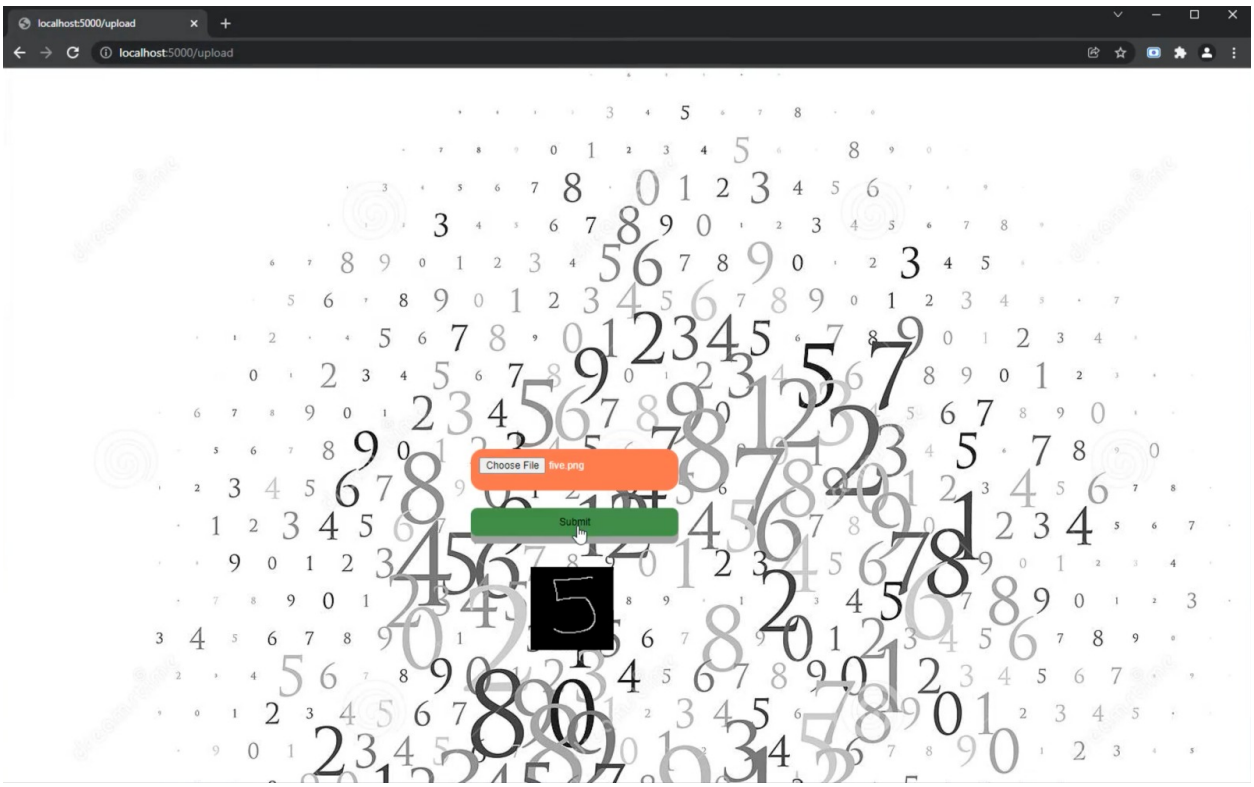
ALGORITHMS	ACCURACY
Convolution Neural Networks	93.52 %
K-Nearest Neighbors	92.68 %
Multilayer Perceptron	98.62 %

EXPERIMENTAL RESULTS: 1) AFTER PREPROCESSING





UI FOR THE SYSTEM :



6 CONCLUSION

According to the results, there is still a lot of potential for development. The data employed for this model, on the other hand, was quite sparse, to begin with. Furthermore, because they were inactive accounts or accounts with only one anime watched, a large amount of data had to be destroyed. We have learned more about Neural networks and Ml algorithms to incorporate in our system. We have evaluated the results using evaluation metrics like MAE, MAPE. It successfully identifies the digits with 98.62 % accuracy.

7 DIRECTIONS FOR FUTURE WORK

Upon working on this project, we learned that for future work, we would start with building an accurate, organized, and lossless dataset first. The dataset we use is a bit overtrained and therefore sometimes identifies the digit differently. Also , we have used digits as our characters but we can also add alphabets, words and double digits. We can do this by using image segmentation of pixels of the image dataset. We could also work on making this system real-time as now we need to upload the image and then it predicts.
Making a real-time system will increase the weightage of this system a lot. These are some future directions for our future work.

8 REFERENCES

1. Rohan Vaidya , Darshan Trivedi , Sagar Satra & Mrunalini Pimpale. “Handwritten Character Recognition Using Deep-Learning” in 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT).
   Handwritten Character Recognition Using Deep-Learning | Request PDF (researchgate.net)
2. Wei Lu, Zhijian Li,Bingxue Shi . ”Handwritten Digits Recognition with Neural Networks and Fuzzy Logic” in IEEE International Conference on Neural Networks, 1995. Proceedings.
   Handwritten digits recognition with neural networks and fuzzy logic | IEEE Conference Publication | IEEE Xplore

3. P. Bhanumathi , Dr. G. M. Nasira .”Handwritten Tamil Character Recognition using Artificial Neural Networks” in 2011 International Conference on Process Automation, Control and Computing.
   Dr G.M.Nasira - Google Scholar

4. B. V. S. Murthy. ”Handwriting Recognition Using Supervised Neural Networks” in International Joint Conference on Neural Networks, 1999. IJCNN ’99. [4] J.Pradeep, E.Srinivasan, S.Hymavathi. ”Neural Network based Handwritten Character Recognition system without feature extraction” in International Conference on Computer, Communication and Electrical Technology ICCCET 2011.
   IRJET-V8I4375.pdf
5. Shun Nishide, Hiroshi G. Okuno, Tetsuya Ogata, Jun Tani. ”Handwriting Prediction Based Character Recognition using Recurrent Neural Network” in 2011 IEEE International Conference on Systems, Man, and Cybernetics.
   Handwriting prediction based character recognition using recurrent neural network | Request PDF (researchgate.net)