

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi-590010



**MINI PROJECT REPORT
ON**

“MNIST HANDWRITTEN DIGIT RECOGNITION WITH DEEP LEARNING”

Submitted in partial fulfillment for the requirements of the sixth semester curriculum

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING**

For the Academic year 2019-2020

Submitted by:

**PRAKRUTHI M
SARAH TARANUM K
SOMYA
SRISHTI NEMA
PRATILIP AICH**

**1MV17CS079
1MV17CS099
1MV17CS106
1MV17CS110
1MV17CS128**

Project carried out at:

Sir M. Visvesvaraya Institute of Technology
Bengaluru-562157

Under the guidance of:

Mrs. Mayuri K P

Associate Professor, Department of CSE
Sir M. Visvesvaraya Institute of Technology, Bengaluru



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SIR. M VISVESVARAYA INSTITUTE OF TECHNOLOGY
HUNASAMARANAHALLI, BENGALURU-562157**

**SIR M. VISVESVARAYA INSTITUTE OF TECHNOLOGY
BENGALURU -562157**

**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING**



CERTIFICATE

It is certified that the project work entitled “**MNIST HANDWRITTEN DIGIT RECOGNITION WITH DEEP LEARNING**” is a bona fide work carried out by Prakruthi M (1MV17CS079), Sarah Taranum K (1MV17CS099), Somya (1MV17CS106), Srifhti Nema (1MV17CS110) and Pratilipi Aich (1MV17VS128) in partial fulfillment for the requirements of mini project for the VI semester curriculum, Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi during the academic year 2019-2020. It is certified that all corrections and suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the course of Bachelor of Engineering.

Name & Signature of Guide

Mrs. Mayuri K P

Associate Prof. & Internal Guide

Dept. Of CSE, Sir MVIT

Bengaluru -562157

Name & Signature of HOD

Dr. Bhanu Prakash G C

HOD, Dept of CSE

Sir MVIT Bengaluru -562157

ACKNOWLEDGMENT

It gives us immense pleasure to express our sincere gratitude to the management of Sir M. Visvesvaraya Institute of Technology, Bangalore for providing the opportunity and the resources to accomplish our project work in their premises.

On the path of learning, the presence of an experienced guide is indispensable and we would like to thank our guide **Mrs. Mayuri K P**, Associate Professor, Dept. of CSE, for her invaluable help and guidance.

We would also like to convey our regards and sincere thanks to **Dr. G. C. Bhanu Prakash**, HOD, Dept. of CSE for his suggestions, constant support and encouragement, Heartfelt and sincere thanks to **Dr. V.R. Manjunath**, Principal, Sir. MVIT for providing us with the infrastructure and facilities needed to develop our project.

We would also like to thank the staff of Department of Computer Science and Engineering and lab-in-charges for their co-operation and suggestions. Finally, we would like to thank all our friends for their help and suggestions without which completing this project would not have been possible.

- Prakruthi M (1MV17CS079)
- Sarah Taranum K (1MV17CS099)
- Somya (1MV17CS106)
- Srishti Nema (1MV17CS110)
- Pratilipi Aich (1MV17CS128)

DECLARATION

We hereby declare that the entire mini project work embodied in this dissertation has been carried out by us and no part has been submitted for any degree or diploma of any institution previously.

Place: Bengaluru

Date:

Signature of Students:

PRAKRUTHI M (1MV17CS079)

SARAH TARANUM K (1MV17CS099)

SOMYA (1MV17CS106)

SRISHTI NEMA (1MV17CS110)

PRATILIP AICH (1MV17CS128)

ABSTRACT

This project aims at building a Handwritten Digit Recognizer using neural system by training the MNIST dataset that takes digits (0-9) from a sample generator with random distortion and passes it to a network of CNN structures. The best result is chosen based on the recognition rate. It is a simple multi-layered neural network by alternating dense layers and [dropout layers](#). Drop-out layer helps in generalization of the model by dropping specified percentage of connections between the two dense layers.

Initially a small network with only a few convolutional layers and one hidden layer was built. Once the model was validated and its accuracy was checked, the size of the network was increased. Additional features were added at every step, validated and checked for accuracy rate. A model with the MNIST dataset is built with an accuracy of 99.75%.

Further research led towards training the EMNIST dataset which is an extended version of the MNIST dataset and includes characters (upper and lower case) along with digits (0-9). Input is augmented and data is validated to increase the efficiency. Augmentation means creating additional images from the already existing by rotating, mirroring and shearing the original ones. This makes the EMNIST dataset unique from the other available datasets.

TABLE OF CONTENTS

CERTIFICATE	ii
ACKNOWLEDGEMENT	iii
DECLARATION	iv
ABSTRACT	v
TABLE OF CONTENTS	vi
LIST OF FIGURES	vii
Chapter 1 INTRODUCTION	1
1.1 Introduction of the project.....	1
1.2 Need for the project.....	1
Chapter 2 RELATED WORK	2-4
2.1 Existing Systems	2
2.2 Objective of the project.....	4
Chapter 3 SPECIFICATIONS	5
3.1 Hardware Requirements.....	5
3.2 Software Requirements.....	5
Chapter 4 IMPLEMENTATION	6-11
4.1 Approach.....	6
4.2 System Design.....	6
4.3 Code snippets.....	8

Chapter 5 RESULTS AND SNAPSHOTS	12-19
5.1 Results.....	12
5.2 Code snippets.....	12
5.3 Snapshots.....	18
 CONCLUSION	 20
 FUTURE ENHANCEMENT	 21
 REFERENCES	 22-23
Papers.....	22
Links.....	23

LIST OF FIGURES

Figure 4.1 The multi-supervised training	13
Figure 5.1 Result of MNIST	14

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION OF THE PROJECT

Our project is to make a minimalistic neural implementation of a handwritten digit recognition system using MNIST dataset. In data science, the recognition of handwritten digits is done using the MNIST dataset. The MNIST dataset is one of the most common datasets used for image classification and accessible from many different sources. In fact, even Tensorflow and Keras allow us to import and download the MNIST dataset directly from their API. Among the deep learning models, the convolutional neural networks (CNN) and the MNIST dataset Model is the most popular one especially for image recognition. This is because CNN is very suitable to represent the image structure.

1.2 NEED FOR THE PROJECT

Handwriting Detection is a technique or ability of a Computer to receive and interpret intelligible handwritten input from source such as paper documents, touch screen, photographs etc. Handwritten Character recognition is one of the area of pattern recognition. The purpose of pattern recognition is to categorizing or classification data or object of one of the classes or categories. Handwriting character recognition is defined as the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. The Handwritten character system is commonly used system in various applications, and it is a technology that is a mandatory need in this world as of now. In traditional systems it is needed to update, and labour is required in order to maintain proper organization of the data. Since for long time we have encountered a severe loss of data because of the traditional method of storing data. Modern day technology is boon, and this technology is making people to store the data over machines, where the storage, organization and accessing of data is easier. Through decades of research, the traditional method has reached its limit while the emergence of deep learning provides a new way to break this limit. Adoption of the Handwritten Character Recognition software is a practical idea and, it is easier to store and access data that was traditionally stored. Furthermore, it provides more security to the data.

CHAPTER 2

RELATED WORK

2.1 EXISTING SYSTEMS

Since a very long time, human used to write their thoughts in the form of letter, transcripts etc. in order to convey them to others. But since the development of computer technology the format of handwritten text changed rapidly to computer generated digital text and so people feel a need of such method that can transform the handwritten text to digital text as it makes the processing of such data very fast and easy.

1) The Handwritten English Dight recognition system based on the Hidden Markov Model (HMM) developed by Rajib.

This method made use of two different feature extractions namely global and local feature extraction. Global feature includes many features like gradient features, projection features and curvature features in the numbers of four, six and four respectively. Whereas local features are calculated by dividing the sample image into nine equal blocks. Gradient feature of each block is calculated using four feature vector. Then, these features are fed into HMM model in order to train it. Data post processing is also utilized by this method in order to decrease the cross classification of different classes. This method takes a lot of time in training and feature extraction. Moreover, it performs poor in case of such inputs, when many characters are combined in a single image.

2) Handwritten Digit Recognition method based on the multi scale neural network training developed and implemented by Velappa Ganapathy.

In order to improve the accuracy, this method used selective threshold, which is calculated based on minimum distance technique. This method also involves the development of GUI, which can find out the character throughout the scanned image. This method provides an accuracy of 85% with moderate level of training. This method used large resolution images (20×28 pixels) for training with lesser training time.

3) Anshul Mehta proposed their work based on the Heuristic segmentation algorithm for Handwritten Digit Recognition System.

Their system performs identification of valid segmentation points between handwritten letters quite well. Fourier descriptors are used in this approach for feature extraction. After a successful segmentation, Discrete Fourier Coefficients are calculated ($a[k]$ and $b[k]$) for the input image. It also provides a comparative analysis of different classification methods. The method also incorporates post processing in order to reduce the error rate but it suffers with a low accuracy and high cross classification rate because of the non-optimal choice of features.

4) T. Som used fuzzy membership function to improve the accuracy of Handwritten text recognition system.

In this method, text images are normalized to 20×10 pixels and then fuzzy approach is the used to each class. Bonding box is created around the character in order to determine the vertical and horizontal projection of the text. Once the image is cropped to a bounding box, it is re-scaled to the size of 10×10 pixels. Then, cropped images are thinned by the help of thinning operation. In order to create the test matrix, all these pre-processed images are placed into a single matrix; one after another. When new (test) images are presented by the user, it is tested for the matching against the test matrix. The method was fast but it provides a low accuracy.

5) Serrano proposed a novel interactive approach for Handwritten Digit recognition. The system requires human suggestion for only those inputs for which the system gets confuse. Although It keep the accuracy to high level, it increases the human lead. The only problem was that the system was not fully automatic and requires human intervention for operation.

The existing systems all had their own drawbacks such as low conversion speed, low accuracy, higher false detection rate and poor performance with noisy input etc. Thus, recognition studies of handwritten character image samples still remain relevant because of their enormous application potentials.

2.2 OBJECTIVE OF THE PROJECT

The purpose of this project is the development of a neural system for Handwritten character recognition by training the EMNIST dataset that takes characters from a sample generator with random distortion and passing it to a network of CNN structures and the best result is chosen based on the recognition rate. It is a simple multi-layered neural network by alternating dense layers and dropout layers. Drop-out layer helps in generalization of the model by dropping specified percentage of connections between the two dense layers.

CHAPTER 3

SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- PROCESSOR : Intel Core i7
- RAM : 8 GB(64-bit)
- HARD DISK : 1TB

3.2 SOFTWARE REQUIREMENTS

1. Python version 3, PyCharm- another ide which supports Python and makes the task completion easy, Numpy.
2. Libraries tensorflow, tensorflowjs, keras.
3. “MNIST-byclass” dataset for recognition of characters.
4. Adam Optimizer as metrics to fit the model.

CHAPTER 4

IMPLEMENTATION

4.1 APPROACH

The purpose of this project is the development of a neural system for Handwritten character recognition by training the EMNIST dataset that takes characters from a sample generator with random distortion and passing it to a network of CNN structures and the best result is chosen based on the recognition rate. It is a simple multi-layered neural network by alternating dense layers and dropout layers. Drop-out layer helps in generalization of the model by dropping specified percentage of connections between the two dense layers.

We will initially begin by building a small network with only a few convolutional layers and one hidden layer. Once the model is validated and its accuracy is checked, we can increase the size of the network. We add extra features at every step, validate and check the accuracy rate. Augmenting the input and validation of data are done to increase the efficiency. Augmentation means creating additional images from the already existing by rotating, mirroring and shearing the original ones. This makes the EMNIST dataset unique from the other available datasets.

4.2 SYSTEM DESIGN

In the proposed framework, two different types of distortion are used:

- 1)The local distortion
- 2)The global distortion

These two distortions can simulate different situations when the character image is captured and generate nearly real samples. For a input training sample, both of the distortions are applied and then the distorted sample is used for training. In this schema classes are digits [0–9], lowercase letters [a–z] and uppercase letters [A–Z]. Thus, there are 62 different classes.

In the proposed framework, we used different CNN models. The CNN for MNIST dataset is an 8-layer network is used (including pooling layers), the “In”, “Conv”, “MaxP”, “Full” and “Out” indicates the input layer, convolutional layer, max pooling layer, fully-connected layer

and softmax output layer, respectively. The training of deep learning models is very important for the final performance, thus many regularization methods for training have been proposed.

Although these methods are lack of theoretical explanation (called training “tricks”), they are able to improve the deep learning classifiers significantly. the CNN model for CASIA dataset employed the multi-supervised training since there are as many as 15 layers. Multiple output layers of different depth are used to enhance the gradients propagated to the lower layers.

Besides the multi-supervised training, the dropout is also employed for both CNN models of MNIST and CASIA datasets. For a certain layer, a number of neurons are selected randomly and set to 0. As a result, all the connections to those selected neurons are disabled. In other words, those selected neurons are removed from the network temporarily. Therefore, in the training, only a sub-structure of the network is trained. After this iteration is finished, we will change the selected neurons(while the number remains the same)and repeat the same process. Through dropout, the different parts of the network can be fully trained, which is important to explore the potential of the network for classification task.

In the proposed framework, 5 CNN models were used to vote for the final recognition result. The voting strategy is as follows. First, in the output of the models, if some class has more votes than other classes, then this class is the final result. Second, if the top classes share the same votes, then the confidences of those classes are added up and the class with the highest confidence is the final result. The multi-model voting is very efficient in improving the performance.

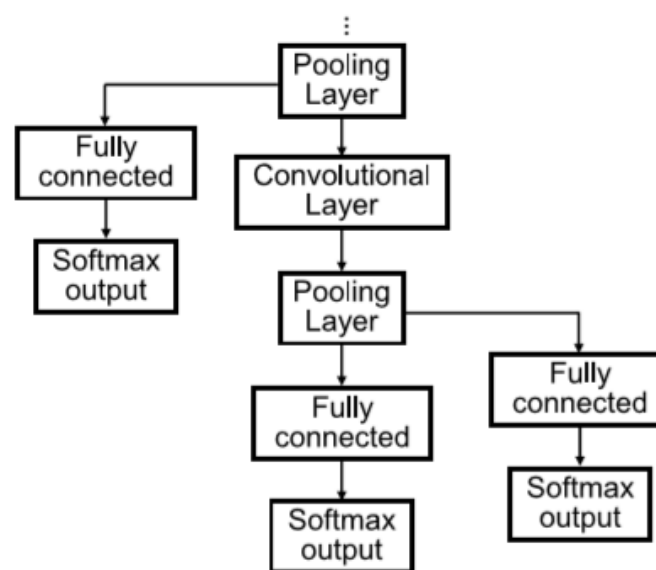


Fig 4.1: The multi-supervised training

4.3 CODE SNIPPETS

MNIST DATASET

#LOAD THE DATA

```
(train_img,train_label),(test_img,test_label) = keras.datasets.mnist.load_data()
```

```
train_img = train_img.reshape([-1, 28, 28, 1])
```

```
test_img = test_img.reshape([-1, 28, 28, 1])
```

```
train_img = train_img/255.0
```

```
test_img = test_img/255.0
```

```
train_label = keras.utils.to_categorical(train_label)
```

```
test_label = keras.utils.to_categorical(test_label)
```

DEFINE THE MODEL ARCHITECTURE

```
model = keras.Sequential([
```

```
    keras.layers.Conv2D(32, (5, 5), padding="same", input_shape=[28, 28, 1]),
```

```
    keras.layers.MaxPool2D((2,2)),
```

```
    keras.layers.Conv2D(64, (5, 5), padding="same"),
```

```
    keras.layers.MaxPool2D((2,2)),
```

```
    keras.layers.Flatten(),
```

```
    keras.layers.Dense(1024, activation='relu'),
```

```
    keras.layers.Dropout(0.2),
```

```
    keras.layers.Dense(10, activation='softmax')
```

```
])
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```



```
# TRAIN THE MODEL
```

```
model.fit(train_img,train_label, validation_data=(test_img,test_label), epochs=10)
```

```
test_loss,test_acc = model.evaluate(test_img, test_label)
```

```
print("Test accuracy:", test_acc)
```

```
# SAVE MODEL AS TFJS FORMAT
```

```
tfjs.converters.save_keras_model(model, 'models')
```

HTML-FRONT END

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
    <meta charset='utf-8'>
```

```
    <meta http-equiv='X-UA-Compatible' content='IE=edge'>
```

```
    <meta name='viewport' content='width=device-width, initial-scale=1'>
```

```
    <title>Hand Written Digit Recognition using TensorFlow.js</title>
```

```
    <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
```

```
    <script src="js/chart.min.js"></script>
```

```
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>
```

```
    <link                                                                    rel="stylesheet"
```

```
    href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" >
```

```
    <link rel="stylesheet" type="text/css" href="style/digit.css">
```

```
</head>
```

```
<body>
```

```

<main>

  <div class="container mt-1">

    <div class="digit-demo-container">

      <h3>DIGIT RECOGNITION</h3>

      <div class="flex-two">

        <div id="canvas_box_wrapper" class="canvas-
        box-wrapper">

          <div id="canvas_box" class="canvas-
          box"></div>

          <div class="col-12">

            <button id="clear-button"
            class="btn btn-
            dark">CLEAR</button>

          </div>

          <div class="col-12">

            <button id="predict-button"
            class="btn btn-
            dark">PREDICT</button>

          </div>

        </div>

      </div>

      <div class="col-6">

        <div id="result_box" class="col-12 col-
        md-7">

          <canvas id="chart_box"
          width="100"
          height="100"></canvas>

        </div>

```

```
<div class="col-12 d-block mt-2 mt-md-0 text-  
md-left prediction-text"></div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</main>
```

```
<script src="js/recogniser.js"></script>
```

```
</body>
```

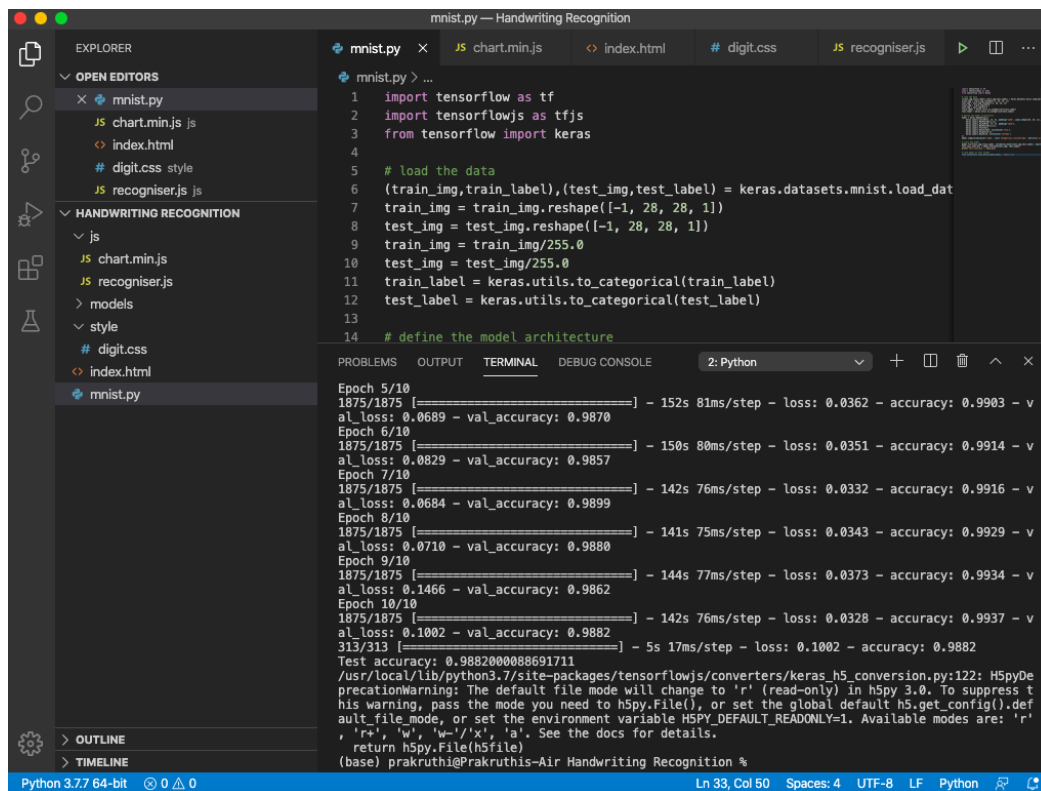
```
</html>
```

CHAPTER 5

RESULTS AND SNAPSHOTS

5.1 TEST RESULTS

When we trained the model with `train_img`, `train_label`, `validation_data`, epochs layers we achieved 98% accuracy.



```
mnist.py — Handwriting Recognition
mnist.py x JS chart.min.js index.html digit.css JS recogniser.js
mnist.py > ...
1 import tensorflow as tf
2 import tensorflowjs as tfjs
3 from tensorflow import keras
4
5 # load the data
6 (train_img, train_label), (test_img, test_label) = keras.datasets.mnist.load_data
7 train_img = train_img.reshape([-1, 28, 28, 1])
8 test_img = test_img.reshape([-1, 28, 28, 1])
9 train_img = train_img/255.0
10 test_img = test_img/255.0
11 train_label = keras.utils.to_categorical(train_label)
12 test_label = keras.utils.to_categorical(test_label)
13
14 # define the model architecture

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 2: Python
Epoch 5/10
1875/1875 [=====] - 152s 81ms/step - loss: 0.0362 - accuracy: 0.9903 - v
al_loss: 0.0689 - val_accuracy: 0.9870
Epoch 6/10
1875/1875 [=====] - 150s 80ms/step - loss: 0.0351 - accuracy: 0.9914 - v
al_loss: 0.0829 - val_accuracy: 0.9857
Epoch 7/10
1875/1875 [=====] - 142s 76ms/step - loss: 0.0332 - accuracy: 0.9916 - v
al_loss: 0.0684 - val_accuracy: 0.9899
Epoch 8/10
1875/1875 [=====] - 141s 75ms/step - loss: 0.0343 - accuracy: 0.9929 - v
al_loss: 0.0710 - val_accuracy: 0.9880
Epoch 9/10
1875/1875 [=====] - 144s 77ms/step - loss: 0.0373 - accuracy: 0.9934 - v
al_loss: 0.1466 - val_accuracy: 0.9862
Epoch 10/10
1875/1875 [=====] - 142s 76ms/step - loss: 0.0328 - accuracy: 0.9937 - v
al_loss: 0.1002 - val_accuracy: 0.9882
313/313 [=====] - 5s 17ms/step - loss: 0.1002 - accuracy: 0.9882
Test accuracy: 0.9882000088691711
Warning: The default file mode will change to 'r' (read-only) in h5py 3.0. To suppress t
his warning, pass the mode you need to h5py.File(), or set the global default h5.get_conf
ig().default_file_mode, or set the environment variable H5PY_DEFAULT_READONLY=1. Availab
le modes are: 'r', 'r+', 'w', 'w-', 'x', 'a'. See the docs for details.
return h5py.File(h5file)
(base) prakruthi@Prakruthi-Air Handwriting Recognition %
```

Fig 5.1: Result of MNIST

5.2 CODE SNIPPETS

RECOGNIZER CODE

```
// -----
```

Create canvas

```
//-----
```

```
var canvasBox = document.getElementById('canvas_box');
```

```

var canvas = document.createElement("canvas");

canvas.setAttribute("width", canvasWidth);

canvas.setAttribute("height", canvasHeight);

canvas.setAttribute("id", canvasId);

canvas.style.backgroundColor = canvasBackgroundColor;

canvasBox.appendChild(canvas);

if(typeof G_vmlCanvasManager != 'undefined') {

    canvas = G_vmlCanvasManager.initElement(canvas);

}

ctx = canvas.getContext("2d");

//-----

// MOUSE DOWN function

//-----

$("#canvas").mousedown(function(e) {

    var rect = canvas.getBoundingClientRect();

    var mouseX = e.clientX- rect.left;;

    var mouseY = e.clientY- rect.top;

    drawing = true;

    addUserGesture(mouseX, mouseY);

    drawOnCanvas();

});

//-----

// TOUCH START function

//-----

```

```

canvas.addEventListener("touchstart", function (e) {

    if (e.target == canvas) {

        e.preventDefault();

    }

    var rect = canvas.getBoundingClientRect();

    var touch = e.touches[0];

    var mouseX = touch.clientX - rect.left;

    var mouseY = touch.clientY - rect.top;


    drawing = true;

    addUserGesture(mouseX, mouseY);

    drawOnCanvas();


}, false);

//-----

// MOUSE MOVE function

/-----

$("#canvas").mousemove(function(e) {

    if(drawing) {

        var rect = canvas.getBoundingClientRect();

        var mouseX = e.clientX- rect.left;;

        var mouseY = e.clientY- rect.top;

        addUserGesture(mouseX, mouseY, true);

```

```

        drawOnCanvas();

    }

});

//-----

// TOUCH MOVE function

//-----

canvas.addEventListener("touchmove", function (e) {

    if (e.target == canvas) {

        e.preventDefault();

    }

    if(drawing) {

        var rect = canvas.getBoundingClientRect();

        var touch = e.touches[0];

        var mouseX = touch.clientX - rect.left;

        var mouseY = touch.clientY - rect.top;

        addUserGesture(mouseX, mouseY, true);

        drawOnCanvas();

    }

}, false);

//-----

// MOUSE UP function

```

```

//-----

$("#canvas").mouseup(function(e) {

    drawing = false;

});

//-----

// TOUCH END function

//-----

canvas.addEventListener("touchend", function (e) {

    if (e.target == canvas) {

        e.preventDefault();

    }

    drawing = false;

}, false);

//-----

// MOUSE LEAVE function

//-----

$("#canvas").mouseleave(function(e) {

    drawing = false;

});

//-----

// TOUCH LEAVE function

//-----

canvas.addEventListener("touchleave", function (e) {

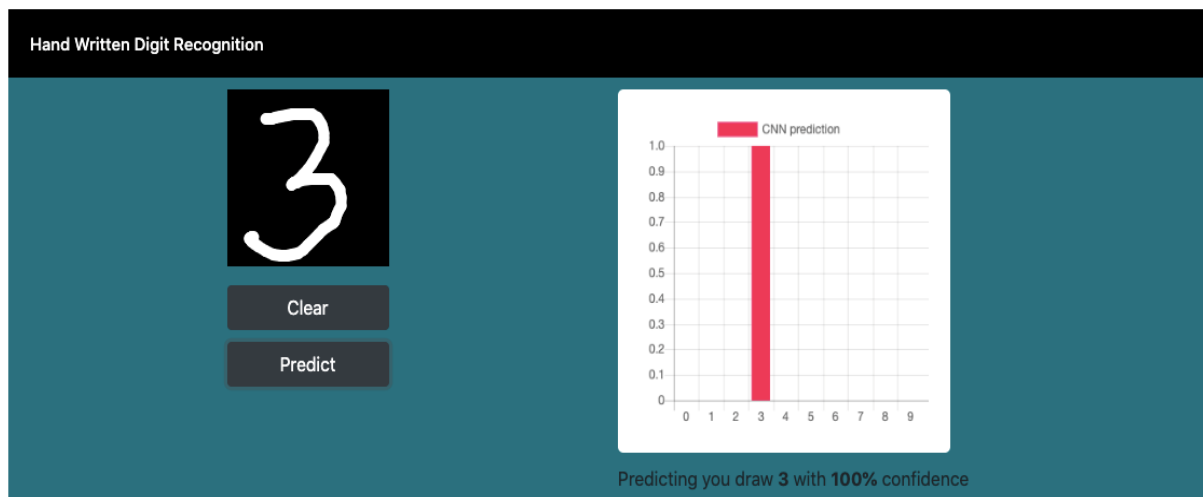
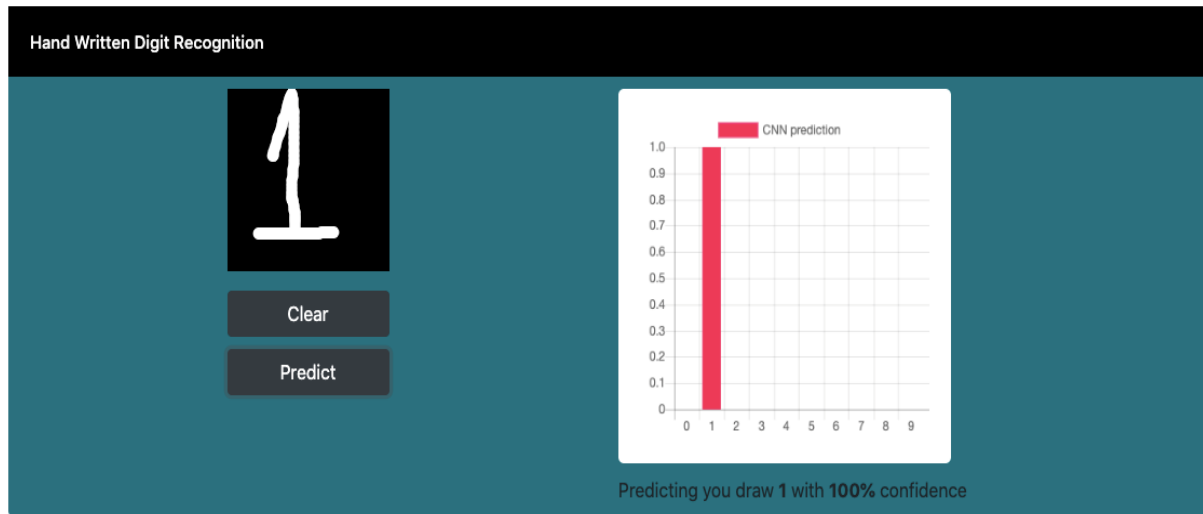
    if (e.target == canvas) {

```



```
e.preventDefault();  
  
}  
  
drawing = false;  
}, false);
```

5.3 SNAPSHOTS

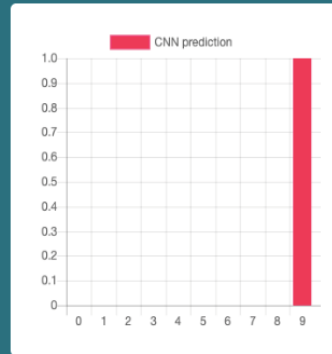


Hand Written Digit Recognition



Clear

Predict



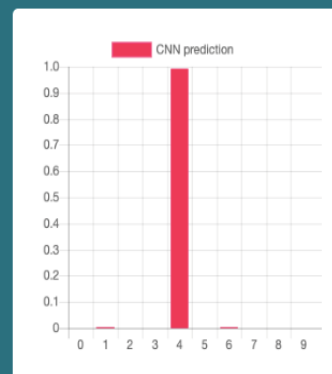
Predicting you draw 9 with 100% confidence

Hand Written Digit Recognition



Clear

Predict



Predicting you draw 4 with 99% confidence

CONCLUSION

In this paper, a CNN-based framework is proposed for handwritten digit recognition. Although CNN is already very powerful for classification tasks, it still need to work under proper framework to achieve the state-of-the-art performance on handwritten character recognition task. In the proposed framework, we have designed proper strategies according to the handwritten character characteristic, such as sample generation, training tricks and multimodel voting. As a result, the proposed framework achieved beyond human performance on MNIST datasets.

Since people's expectation on machine learning is endless, human's performance is not the end of our research. Simple strategy requires much more computational resource. The more advanced way is to find better sample generation methods, training scheme and network structure of CNN.

The handwritten digit recognition is very easy for human but extremely difficult for machines. In the past decades, many researchers have been trying to improve the handwritten digit recognition methods. However, the performance of those methods was still not comparable with human. No one even believe that someday the machine learning methods can surpass human's performance. Now, based on the powerful CNN classifier, the proposed framework achieved a goal beyond human performance on handwritten character recognition.

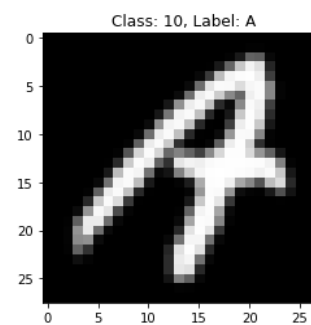
FUTURE ENHANCEMENT

We planned to further improve our project by extending the classification model to English alphabets (upper and lower case) and have been able to train a CNN model using the EMNIST-byclass dataset. We have acquired the accuracy of %.

Model: "sequential"		
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 12, 12, 12)	312
dropout (Dropout)	(None, 12, 12, 12)	0
conv2d_1 (Conv2D)	(None, 5, 5, 18)	1962
dropout_1 (Dropout)	(None, 5, 5, 18)	0
conv2d_2 (Conv2D)	(None, 4, 4, 24)	1752
flatten (Flatten)	(None, 384)	0
dense (Dense)	(None, 150)	57750
dense_1 (Dense)	(None, 47)	7097
Total params: 68,873		
Trainable params: 68,873		
Non-trainable params: 0		

```
Prediction: 27 , Char: R
Label: 27
Prediction: 15 , Char: F
Label: 40
Prediction: 9 , Char: 9
Label: 16
Prediction: 8 , Char: 8
Label: 44
Prediction: 23 , Char: N
Label: 23
Prediction: 28 , Char: S
Label: 28
Prediction: 9 , Char: 9
Label: 9
Prediction: 27 , Char: R
Label: 27
Prediction: 41 , Char: g
Label: 41
```

```
In [48]: show_img(train_data, 149)
```



We are planning to improve the model:

1. Achieve a better accuracy percentage.
2. Develop an easy to use UI for the same.

REFERENCES

PAPERS

[1] Matthew Y.W. Teow, Artificial Intelligence Lab, Faculty of Engineering and Computing “Understanding Convolutional Neural Networks Using A Minimal Model for Handwritten Digit Recognition” in 2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS 2017), 21 October 2017, Kota Kinabalu, Sabah, Malaysia

[2] Kh Tohidul Islam, Ghulam Mujtaba, Dr. Ram Gopal Raj, Henry Friday Nweke “Handwritten Digits Recognition with Artificial Neural Network” in 2017 International Conference on Engineering Technology and Technopreneurship (ICE2T)

[3] Mahmoud M. Abu Ghosh, Ashraf Y. Maghari “A Comparative Study on Handwriting Digit Recognition Using Neural Networks” in 2017 International Conference on Promising Electronic Technologies

[4] Li Chen, Song Wang, Wei Fan, Jun Sun, Satoshi Naoi “Beyond Human Recognition: A CNN-Based Framework for Handwritten Character Recognition” in 2015 3rd IAPR Asian Conference on Pattern Recognition

LINKS

1. <https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d>
2. <https://machinelearningmastery.com/handwritten-digit-recognition-using-convolutional-neural-networks-python-keras/>
3. <https://towardsdatascience.com/handwritten-digit-recognition-with-tensorflow-js-6ddb22ae195>
4. <https://www.digitalocean.com/community/tutorials/how-to-build-a-neural-network-to-recognize-handwritten-digits-withtensorflow#:~:text=Using%20TensorFlow%2C%20an%20open%2Dsource,label%20for%20the%20digit%20displayed.>
5. <https://medium.com/@arthurflor23/handwritten-text-recognition-using-tensorflow-2-0-f4352b7afe16>