

# **A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM**

**submitted by**

**PNT2022TMID37612**

<b>PRIYANKA S</b>	<b>-</b>	<b>210119205005</b>
<b>VARSHA K</b>	<b>-</b>	<b>210119205006</b>
<b>SWETHA S</b>	<b>-</b>	<b>210119104010</b>
<b>RESHMA S</b>	<b>-</b>	<b>210119104008</b>
<b>DEVI K</b>	<b>-</b>	<b>210119205003</b>
<b>ANU JULIET A</b>	<b>-</b>	<b>210119205001</b>

## **1. INTRODUCTION**

1.1 Project Overview

1.2 Purpose

## **2. LITERATURE SURVEY**

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

## **3. IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

## **4. REQUIREMENT ANALYSIS**

4.1 Functional requirement

4.2 Non-Functional requirements

## **5. PROJECT DESIGN**

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

## **6. PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

## **7. CODING & SOLUTIONING (Explain the features added in the project along with code)**

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

## **8. TESTING**

8.1 Test Cases

8.2 User Acceptance Testing

## **9. RESULTS**

9.1 Performance Metrics

## **10. ADVANTAGES & DISADVANTAGES**

## **11. CONCLUSION**

## **12. FUTURE SCOPE**

## **13. APPENDIX**

Source Code GitHub & Project Demo Link

# **1. INTRODUCTION**

## **1.1 PROJECT OVERVIEW**

Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many real-time applications. The MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI

## **1.2 PURPOSE**

Handwritten character recognition is one of the practically important issues in pattern recognition applications. The applications of digit recognition include postal mail sorting, bank check processing, form data entry, etc. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize handwritten digits and which is submitted by users by way of a scanner, tablet, and other digital devices.

# **2 LITERATURE SURVEY**

## **2.1 EXISTING PROBLEM**

The fundamental problem with handwritten digit recognition is that handwritten digits do not always have the same size, width, orientation, and margins since they vary from person to person. Additionally, there would be issues with identifying the numbers because of similarities

between numerals like 1 and 7, 5 and 6, 3 and 8, 2 and 5, 2 and 7, etc. Finally, the individuality and variation of each individual's handwriting influence the structure and appearance of the digits.

## **2.2 REFERENCES**

- [1] Assayony, O. Mohammed and S. A. Mahmoud, Recognition of Arabic Handwritten Words using Gabor-based Bag-of-Features Framework. International Journal of Computing and Digital Systems, 2018.
- [2] Assayony, O. Mohammed and S. A. Mahmoud, An Enhanced Bag-of-Features Framework for Arabic Handwritten Sub-words and Digits Recognition. Journal of Pattern Recognition and Intelligent Systems, 2016.
- [3] L. Rothacker and G. A. Fink, Robust Output Modeling in Bag-of-Features HMMs for Handwriting Recognition. International Conference on Frontiers in Handwriting Recognition, 2016.
- [4] P. Vincent, H. Larochelle, I. La-joie, Y Bengio and P.A.Manzagol, Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. Journal of Machine Learning Research 11, 2010.
- [5] H. E. Mubtaahij, A. Halli and K. Satori, Using Features of Local Densities, Statistics and HMM Toolkit (HTK) for Offline Arabic Handwriting Text Recognition. International Journal of Electrical Systems and Information Technology, 2017.
- [11] M. Rabi and M. Amrouch, Recognition of Cursive Arabic Handwritten Text Using Embedded Training Based on Hidden Markov Models. International Journal of Pattern

## 2.3 PROBLEM STATEMENT DEFINITION

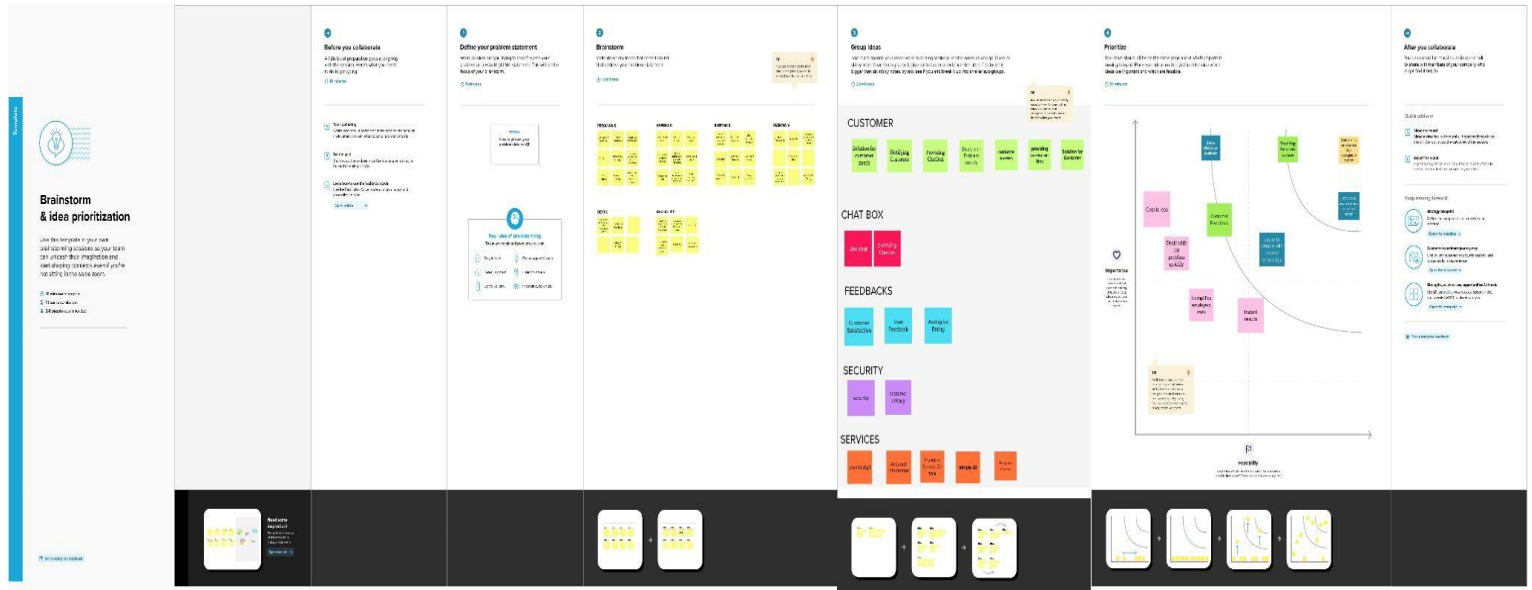
For years, the traffic department has been combating traffic law violators. These offenders endanger not only their own lives, but also the lives of other individuals. Punishing these offenders is critical to ensuring that others do not become like them. Identification of these offenders is next to impossible because it is impossible for the average individual to write down the license plate of a reckless driver. Therefore, the goal of this project is to help the traffic department identify these offenders and reduce traffic violations as a result.

## 3. IDEATION AND PROPOSED SOLUTION

### 3.1 EMPATHY MAP CANVAS



## 3.2 IDEATION & BRAINSTORMING



## 3.4 PROPOSED SOLUTION

Problem Statement	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Employee in post office	I am trying to sort the postal mail	I can't recognize what is written	The handwriting of the person	Like i am entering a wrong digit
PS-2	Employee in bank	I am trying to processing the cheques	I can't recognize what is written	The account number is not written clearly	Like i am entering a wrong account

					number
PS-3	Employee in XXX company	I am trying to enter the data	I can't recognize what is written	The digits are not mentioned clearly	Like i am entering an invalid data

3.4 PROBLEM SOLUTION FIT





## 4. REQUIREMENT ANALYSIS

### 4.1 FUNCTIONAL REQUIREMENT

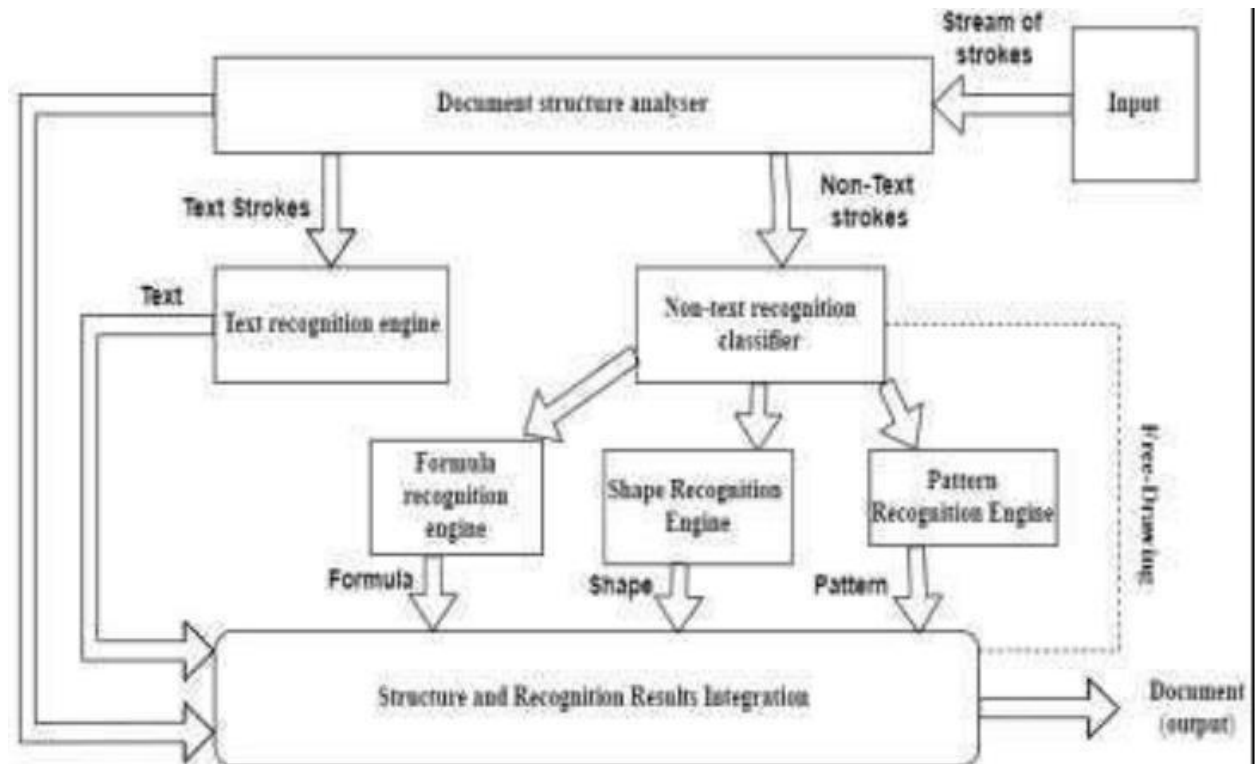
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Image Data	Handwritten digit recognition refers to a computer's capacity to identify human handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categorize them into ten established classifications (0-9). In the realm of deep learning, this has been the subject of countless studies
FR-4	Website	Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you've ever visited. The type of hosting determines how much space is allotted to a website on a server. Shared, dedicated, VPS, and reseller hosting are the four basic varieties
FR-5	Digit Classifier Model	To train a convolutional network to predict the digit from an image, use the MNIST database of handwritten digits. get the training and validation data first
FR-6	Cloud	The cloud offers a range of IT services, including virtual storage, networking, servers, databases, and applications. In plain English, cloud computing is described as a virtual platform that enables unlimited storage and access to your data over the internet

## 4.2 NON - FUNCTIONAL REQUIREMENT

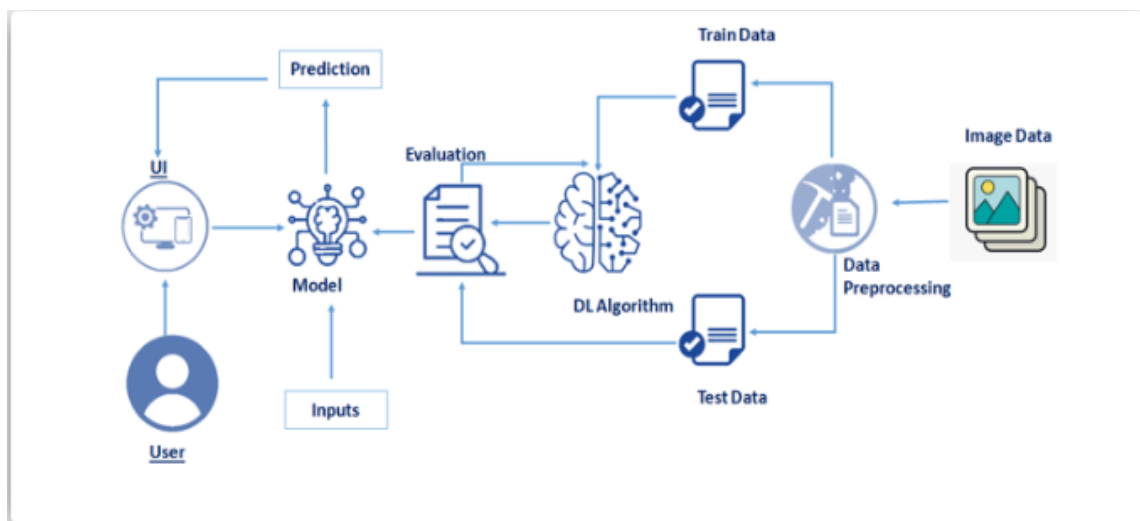
FR No.	Non -Functional Requirement (Epic)	Description
FR-1	Usability	One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail
FR-2	Security	1) The system generates a thorough description of the instantiation parameters, which might reveal information like the writing style, in addition to a categorization of the digit. 2) The generative models are capable of segmentation driven by recognition. 3) The procedure uses a relatively.
FR-3	Reliability	The samples are used by the neural network to automatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantity of training instances. Numerous techniques and algorithms, such as Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc., can be used to recognise handwritten numbers
FR-4	Performance	Information is restricted to each users limited access
FR-5	Availability	Applications for digit recognition include filling out forms, processing bank checks, and sorting mail
FR-6	Scalability	The system should be able to handle 1000 users accessing the website at the same time

## 5. PROJECT DESIGN

### 5.1 DATA FLOW DIAGRAM



## TECHNICAL ARCHITECTURE



### 5.3 USER STORIES

User Type	Functional Requirement	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming	I can access my account / dashboard	High	Sprint-1
		USN-2	my password. As a user, I will receive confirmation email	I can receive confirmation	High	Sprint-1
		USN-3	once I have registered for the application As a user, I can register for the application through Facebook	email & click confirm I can register & access the dashboard with Facebook	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	LoginI can register the application with Gmail	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login to the application	High	Sprint-1
	Dashboard	USN-6	As a user, I can view the application's home page where I can read the instructions to use this application	I can read instructions also and the home page is user-friendly	Low	Sprint-1

	Upload Image	USN-7	As a user, I can able to input the images of digital documents to the application	As a user, I can able to input the images of digital documents to the	High	Sprint-3
	Predict	USN-8	As a user I can able to get the recognised digit as output from the images of digital documents or images	application I can access the recognized digits from digital document or images	High	Sprint-3
		USN-9	As a user, I will train and test the input to get the maximum accuracy of output.	I can able to train and test the application until it gets maximum accuracy	Medium	Sprint-4

Customer (Web user)	Accessibility	USN-10	As a user, I can use the web application virtually anywhere.	I can use the application in any device with a browse	Medium	Sprint-4
---------------------	---------------	--------	--	---	--------	----------

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwriting.	10	Low	Anu Juliet A Swetha S
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	Medium	Varsha K Reshma S Devi K Priyanka S
Sprint-2	Model Building	USN-3	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	High	Devi K Varsha K Reshma S
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Swetha S Priyanka S
Sprint-2	Train & test the model	USN-6	As a user, let us train our model with our image dataset.	6	Medium	Reshma S Priyanka S Varsha K
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an android application or web application in order to predict something.	2	Low	Anu Juliet A Devi K
Sprint-3	Building UI Application	USN-8	As a user, I will upload the handwritten digit image to the application by clicking a upload button.	5	High	Anu Juliet A Devi K Reshma S
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low	Priyanka S Swetha S Varsha k

Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium	Varsha K Reshma S Anu Juliet A
Sprint-4	Train the model on IBM	USN-11	As a user, I train the model on IBM and integrate flask/Django with scoring end point.	10	High	Swetha S Varsha K Priyanka S
Sprint-4	Cloud Deployment	USN-12	As a user, I can access the web application and make the use of the product from anywhere.	10	High	Reshma S Priyanka S Varsha K

## 6.2 SPRINT DELIVERY SCHEDULE

SPRINT	TOTAL STORY POINT	DURATION	SPRINT START DATE	SPRINT END DATE (PLANNED)	STORY POINTS COMPLETED (AS ON PLANNED DATE)	SPRINT RELEASE DATE (ACTUAL)
Sprint - I	11	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint - II	9	6 Days	31 Oct 2022	05 Nov 2022	9	05 Nov 2022
Sprint - III	10	6 Days	07 Oct 2022	12 Nov 2022	10	12 Nov 2022
Sprint - IV	9	6 Days	14 Nov 2022	19 Nov 2022	9	19 Nov 2022

## 7. CODING & SOLUTIONING

```
app.py X
C:\Users\lenovo\Desktop\project_1> Application Building > app.py
1 import numpy as np
2 import os
3 from PIL import Image
4 from flask import Flask, request, render_template, url_for
5 from werkzeug.utils import secure_filename, redirect
6 #from event.pywsgi import WSGIServer
7 from keras.models import load_model
8 from keras.preprocessing import image
9 from flask import send_from_directory
10
11 UPLOAD_FOLDER = 'F:\IBM\IBM-Project-50222-1660900453-main\Application Building\data'
12
13
14 app = Flask(__name__)
15 app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
16
17 model = load_model("./models/mnistCNN.h5")
18
19
20 @app.route('/')
21 def index():
22     return render_template('index.html')
23
24
25 @app.route('/predict', methods=['GET', 'POST'])
26 def upload():
27     if request.method == "POST":
28         f = request.files["image"]
29         filepath = secure_filename(f.filename)
30         f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))
31
```

```
app.py X
C:\Users\lenovo\Desktop\project_1> Application Building > app.py
18
19
20 @app.route('/')
21 def index():
22     return render_template('index.html')
23
24
25 @app.route('/predict', methods=['GET', 'POST'])
26 def upload():
27     if request.method == "POST":
28         f = request.files["image"]
29         filepath = secure_filename(f.filename)
30         f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))
31
32         upload_img = os.path.join(UPLOAD_FOLDER, filepath)
33         img = image.open(upload_img).convert("L") # convert image to monochrome
34         img = img.resize((28, 28)) # resizing of input image
35
36         im2arr = np.array(img) # converting to image
37         im2arr = im2arr.reshape(1, 28, 28, 1) # reshaping according to our requirement
38
39         pred = model.predict(im2arr)
40
41         num = np.argmax(pred, axis=1) # printing our labels
42
43         return render_template('predict.html', num=str(num[0]))
44
45
46 if __name__ == '__main__':
47     app.run(debug=True, threaded=False)
```

## 8. TESTING

### 8.1 TEST CASES

Test case ID	Feature Type	Component	Expected Result	Actual Result	Status
--------------	--------------	-----------	-----------------	---------------	--------



HP_TC_001	UI	Home Page	The Home page must be displayed properly	Working as expected	PASS
HP_TC_002	UI	Home Page	The Home page must be displayed properly in all sizes	The UI is not displayed properly in screen size 2560 x 1801 and 768 x 630	FAIL
HP_TC_003	Functional	Home Page	The input image should be uploaded to the application successfully	Working as expected	PASS
HP_TC_004	Functional	Home Page	The application should not allow user to select a non image file	User is able to upload any file	FAIL
HP_TC_005	Functional	Home Page	The page should redirect to the results page	Working as expected	PASS
BE_TC_001	Functional	Backend	All the routes should properly work	Working as expected	PASS
M_TC_001	Functional	Model	The model should rescale the image and predict the results	Working as expected	PASS
M_TC_002	Functional	Model	The model should predict the number	Working as expected	PASS
M_TC_003	Functional	Model	The model should predict the number in the complex image	The model fails to identify the digit since the model is not built to handle such data	FAIL
RP_TC_001	UI	Result Page	The Result page must be displayed properly	Working as expected	PASS
RP_TC_002	UI	Result Page	The input image should be displayed properly	The size of the input image exceeds the display container	FAIL
RP_TC_003	UI	Result Page	The result should be displayed properly	Working as expected	PASS

RP_TC_004	UI	Result Page	The other predictions should be displayed properly	Working as expected	PASS
-----------	----	-------------	--	---------------------	------

## 8.2 USER ACCEPTANCE TESTING

### DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Totals	6	1	4	3	14

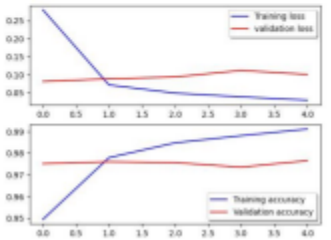
### TEST CASE ANALYSIS

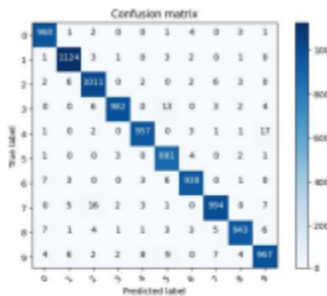
Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1

--

Performance	3	0	0	3
Exception Reporting	2	0	0	2

9. PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Model Summary		
2.	Accuracy	Training Accuracy - 99% Validation Accuracy - 97%	

3.	Confusion Matrix																																																																							
4.	Classification Report	<table><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr><tr><td>0</td><td>0.98</td><td>0.99</td><td>0.99</td><td>988</td></tr><tr><td>1</td><td>0.98</td><td>0.99</td><td>0.99</td><td>1115</td></tr><tr><td>2</td><td>0.97</td><td>0.98</td><td>0.97</td><td>1012</td></tr><tr><td>3</td><td>0.99</td><td>0.97</td><td>0.98</td><td>1018</td></tr><tr><td>4</td><td>0.98</td><td>0.97</td><td>0.98</td><td>102</td></tr><tr><td>5</td><td>0.96</td><td>0.99</td><td>0.97</td><td>852</td></tr><tr><td>6</td><td>0.98</td><td>0.98</td><td>0.98</td><td>958</td></tr><tr><td>7</td><td>0.98</td><td>0.97</td><td>0.97</td><td>1028</td></tr><tr><td>8</td><td>0.98</td><td>0.97</td><td>0.98</td><td>974</td></tr><tr><td>9</td><td>0.96</td><td>0.96</td><td>0.96</td><td>1009</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.98</td><td>10000</td></tr><tr><td>macro avg</td><td>0.98</td><td>0.98</td><td>0.98</td><td>10000</td></tr><tr><td>weighted avg</td><td>0.98</td><td>0.98</td><td>0.98</td><td>10000</td></tr></table>		precision	recall	f1-score	support	0	0.98	0.99	0.99	988	1	0.98	0.99	0.99	1115	2	0.97	0.98	0.97	1012	3	0.99	0.97	0.98	1018	4	0.98	0.97	0.98	102	5	0.96	0.99	0.97	852	6	0.98	0.98	0.98	958	7	0.98	0.97	0.97	1028	8	0.98	0.97	0.98	974	9	0.96	0.96	0.96	1009	accuracy			0.98	10000	macro avg	0.98	0.98	0.98	10000	weighted avg	0.98	0.98	0.98	10000
	precision	recall	f1-score	support																																																																				
0	0.98	0.99	0.99	988																																																																				
1	0.98	0.99	0.99	1115																																																																				
2	0.97	0.98	0.97	1012																																																																				
3	0.99	0.97	0.98	1018																																																																				
4	0.98	0.97	0.98	102																																																																				
5	0.96	0.99	0.97	852																																																																				
6	0.98	0.98	0.98	958																																																																				
7	0.98	0.97	0.97	1028																																																																				
8	0.98	0.97	0.98	974																																																																				
9	0.96	0.96	0.96	1009																																																																				
accuracy			0.98	10000																																																																				
macro avg	0.98	0.98	0.98	10000																																																																				
weighted avg	0.98	0.98	0.98	10000																																																																				

## 10. ADVANTAGES & DISADVANTAGES

### ADVANTAGES

- Reduces manual work
- More accurate than average human
- Capable of handling a lot of data
- Can be used anywhere from any device

### DISADVANTAGES

- Cannot handle complex data
- All the data must be in digital format
- Requires a high performance server for faster predictions
- Prone to occasional errors

## **11. CONCLUSION**

This project demonstrated a web application that uses machine learning to recognise handwritten numbers. Flask, HTML, CSS, JavaScript, and a few other technologies were used to create this project. The model predicts the handwritten digit using a CNN network. During testing, the model achieved a 99.61% recognition rate. The proposed project is scalable and can easily handle a huge number of users. Since it is a web application, it is compatible with any device that can run a browser. This project is extremely useful in real-world scenarios such as recognizing number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on. There is so much room for improvement, which can be implemented in subsequent versions.

## **12. FUTURE SCOPE**

This project is far from complete and there is a lot of room for improvement. Some of the improvements that can be made to this project are as follows:

- Add support to detect from digits multiple images and save the results
- Add support to detect multiple digits
- Improve model to detect digits from complex images
- Add support to different languages to help users from all over the world

This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency

## SOURCE CODE

## MODEL CREATION

## Importing the required libraries

```
import numpy
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D
from keras.optimizers import Adam
from keras.utils import np_utils
```

## Python

## Loading the data

```
(X_train,y_train),(X_test,y_test)=mnist.load_data()
```

## Python

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [=====] - 0s 0us/step
```

## Analysing the data

```
X_train[0]
```

## Python

[illegible]

```
0, 0]], dtype=uint8)

y_train[0]

5

import matplotlib.pyplot as plt
plt.imshow(X_train[0])

<matplotlib.image.AxesImage at 0x7fbc944b9150>

0
5
10
15
20
25
0 5 10 15 20 25
```

```
Reshaping the data

X_train=X_train.reshape(60000,28,28,1).astype('float32')
X_test=X_test.reshape(10000,28,28,1).astype('float32')

Applying one hot encoding

number_of_classes=10
y_train=np_utils.to_categorical(y_train,number_of_classes)
y_test=np_utils.to_categorical(y_test,number_of_classes)

y_train[0]

array([[0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

```
Adding CNN layers

model=Sequential()
model.add(Conv2D(64,(3,3),input_shape=(28,28,1),activation='relu'))
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(Flatten())
model.add(Dense(number_of_classes,activation='softmax'))

Compiling the model

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

Train the model

model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=5,batch_size=32)

Epoch 1/5
1875/1875 [=====] - 195s 103ms/step - loss: 0.2126 - accuracy: 0.9543 - val_loss: 0.0867 - val_accuracy: 0.9730
```

### Observing the metrics

```
metrics=model.evaluate(X_test,y_test,verbose=0)
print("Metrics(Test loss & Test Accuracy):")
print(metrics)
```

Python

```
Metrics(Test loss & Test Accuracy):
[0.11479371041059494, 0.9763000011444092]
```

### Test the model

```
prediction=model.predict(X_test[:4])
print(prediction)
```

Python

```
1/1 [=====] - 0s 109ms/step
[[[4.21976367e-19 2.76640418e-21 1.07767846e-13 4.71178166e-11
  2.10180035e-22 1.09333558e-23 3.61171152e-26 1.00000000e+00
  6.55367672e-18 5.30832922e-15]
 [1.43194623e-10 1.67714037e-11 9.9998331e-01 1.29229049e-12
  2.77214423e-15 6.86406363e-20 1.72249804e-06 5.78033124e-16
  6.75309364e-10 1.71233192e-21]
```

### Test the model

```
prediction=model.predict(X_test[:4])
print(prediction)
```

Python

```
1/1 [=====] - 0s 109ms/step
[[[4.21976367e-19 2.76640418e-21 1.07767846e-13 4.71178166e-11
  2.10180035e-22 1.09333558e-23 3.61171152e-26 1.00000000e+00
  6.55367672e-18 5.30832922e-15]
 [1.43194623e-10 1.67714037e-11 9.9998331e-01 1.29229049e-12
  2.77214423e-15 6.86406363e-20 1.72249804e-06 5.78033124e-16
  6.75309364e-10 1.71233192e-21]
 [1.41170897e-09 9.9998927e-01 9.63970237e-09 3.00579769e-11
  4.58667927e-07 1.44993351e-08 1.61555591e-09 6.39673292e-09
  6.12086751e-07 4.38722626e-12]
 [1.00000000e+00 1.69338509e-15 1.90748825e-13 5.08622956e-16
  4.55195029e-16 3.16392081e-14 1.27083393e-08 5.42493669e-14
  6.74359917e-12 2.46589225e-11]]]
```

```
import numpy as np
print(np.argmax(prediction,axis=1))
print(y_test[:4])
```

### Test with saved model

```
from keras.datasets import mnist
from matplotlib import pyplot
(X_train,y_train),(X_test,y_test)=mnist.load_data()
print("X_train:" +str(X_train.shape))
print("y_train:" +str(y_train.shape))
print("X_test:" +str(X_test.shape))
print("y_test:" +str(y_test.shape))
from matplotlib import pyplot
for i in range(9):
    pyplot.subplot(330+1+i)
    pyplot.imshow(X_train[i],cmap=pyplot.get_cmap('gray'))
    pyplot.show()
```

Python

```
X_train:(60000, 28, 28)
y_train:(60000,)
X_test:(10000, 28, 28)
y_test:(10000,)
```





# FLASK APP

```
import numpy as np
import os
from PIL import Image
from flask import Flask, request, render_template, url_for
from werkzeug.utils import secure_filename, redirect
# from gevent.pywsgi import WSGIServer
from keras.models import load_model
from keras.preprocessing import image
from flask import send_from_directory

UPLOAD_FOLDER = 'F:\ibm\IBM-Project-50222-1660900453-main\Application Building\data'

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

model = load_model("./models/mnistCNN.h5")

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == "POST":
        f = request.files["image"]
        filepath = secure_filename(f.filename)
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))
```

```
@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == "POST":
        f = request.files["image"]
        filepath = secure_filename(f.filename)
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))

        upload_img = os.path.join(UPLOAD_FOLDER, filepath)
        img = Image.open(upload_img).convert("L") # convert image to monochrome
        img = img.resize((28, 28)) # resizing of input image

        im2arr = np.array(img) # converting to image
        im2arr = im2arr.reshape(1, 28, 28, 1) # reshaping according to our requirement

        pred = model.predict(im2arr)

        num = np.argmax(pred, axis=1) # printing our Labels

        return render_template('predict.html', num=str(num[0]))

if __name__ == '__main__':
    app.run(debug=True, threaded=False)
```

## HOME PAGE (HTML)

```
<html>

<head>
  <title>HDR</title>

  <meta name="viewport" content="width=device-width">

  <link href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Varela+Round&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Source+Code+Pro:wght@500&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Calistoga|Josefin+Sans:400,700|Pacifico&display=swap" rel="stylesheet">

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR01XCbMQv3Xipma34M"
  <link rel="stylesheet" type="text/css" href= "{{ url_for('static',filename='css/style.css')}}">

  <script src="https://kit.fontawesome.com/b3aed9cb07.js" crossorigin="anonymous"></script>

  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6j"
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-U02eT0CpHqdS3Q6hJty5KVphtPhzWj9W01clH"
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy60RQ6VrjIEaFf"
  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">
  <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>

</head>
<style>
  body{
    background-image: url('static/images/bc1.jpg');
    background-repeat: no-repeat;
```

```
<script>
  function preview() {
    frame.src=URL.createObjectURL(event.target.files[0]);
  }

  $(document).ready(function() {
    $('#clear_button').on('click', function() {
      $('#image').val('');
      $('#frame').attr('src','');
    });
  });
</script>

<body>
<div class="container p-7 my-7 bg-dark text-light">
  <h1>HandWritten Digit Recognition System</h1>
</div>

  <section id="content">

    <div class="leftside">
      <form action="/predict" method="POST" enctype="multipart/form-data">
        <label><h3>Select a image:</h3></label>
        <input id="image" type="file" name="image" accept="image/png, image/jpeg" onchange="preview()"><br><br>
        <img id="frame" width="100px" height="100px"/>
        <div class="buttons_div">
          <button type="submit" class="btn btn-dark">Predict</button>
          <button type="button" class="btn btn-dark">&nbsp; Clear &nbsp;</button>
        </div>
      </form>
    </div>
```

```

        <input id="image" type="file" name="image" accept="image/png, image/jpeg" onchange="preview()"><br><br>
        <img id="frame" width="100px" height="100px"/>
        <div class="buttons_div">
            <button type="submit" class="btn btn-dark">Predict</button>
            <button type="button" class="btn btn-dark">&nbsp;Clear &nbsp;</button>
        </div>
    </form>
</div>
</section>

<!--<section id="content">

    <div class="leftside">
    <form action="/predict" method="POST" enctype="multipart/form-data">
    <label>Select a image:</label>
    <input id="image" type="file" name="image" accept="image/png, image/jpeg" onchange="preview()"><br><br>
    <img id="frame" width="100px" height="100px"/>
    <div class="buttons_div">
        <button type="submit" class="btn btn-dark" id="predict_button">Predict</button>
        <button type="button" class="btn btn-dark" id="clear_button">&nbsp;Clear &nbsp;</button>
        <button type="submit" class="btn btn-light">Predict</button>
        <button type="button" class="btn btn-light">&nbsp;Clear &nbsp;</button>
    </div>
    </form>
    </div>
</section-->

</body>

</html>

```

## PREDICT PAGE

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prediction</title>
</head>

<style>
    body{
        background-image: url('static/images/bc2.jpg');
        background-repeat: no-repeat;
        background-size: cover;
    }

    #rectangle{
        width:400px;
        height:150px;
        background-color: #000000;
        border-radius: 15px;
        position:absolute;
        box-shadow: 0px 0px 10px 5px white;
        top:25%;
        left:50%;
        transform:translate(-50%,-50%);
    }

    #head{
        text-align: center;
        font-size: 30px;
        margin: 0 auto;
        padding: 3% 5%;
    }

```

```

position: absolute;
box-shadow: 0px 0px 10px 5px ■ white;
top: 25%;
left: 50%;
transform: translate(-50%, -50%);
}

#head{
text-align: center;
font-size: 30px;
margin: 0 auto;
padding: 3% 5%;
font-family: Arial, Helvetica, sans-serif;
color: ■ white;
}

#num{
font-size: 50px;
}

</style>

<body>

<div id="rectangle">
|
<h1 id="head">Predicted Number : <br><center id="num">{{num}}</center></h1>
|
</div>

</body>
</html>

```

# GitHub

<https://github.com/IBM-EPBL/IBM-Project-48667-1660811150>

**PROJECT DEMO VIDEO**

<https://drive.google.com/file/d/1Dbf64bIZYxdfL7NdJeN5H5353BYfLP45/view?usp=sharing>