Team ID: PNT2022TMID25747

PROJECT TITLE: HANDWRITTEN RECOGNITION USING MACHINE LEARNING

1. INTRODUCTION

1.1 Project Overview

The reliance of humans over machines has never been so high such that from object classification in photographs to adding sound to silent movies everything can be performed with the help of deep learning and machine learning algorithms. Likewise, Handwritten text recognition is one of the significant areas of research and development with a streaming number of possibilities that could be attained. Handwriting recognition (HWR), also known as Handwritten Text Recognition (HTR), is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touch-screens and other devices.

1.2 Purpose

Digit recognition is a fundamental, but most challenging in the field of pattern recognition with a large number of useful applications. It has been an intense field of research since the early days of computer science due to it being a natural way of interacting between computers and humans. Digit recognition is the process of detecting and recognizing characters from the input image and converting it into ASCII or other equivalent machine editable form.

In recent years, deep learning-based techniques have been gaining significant interest in the research community for solving a variety of supervised, unsupervised and reinforcement learning problems. One of the most well-known and widely used techniques are convolutional neural networks (CNNs), a kind of neural networks which are able to automatically extract relevant features from input data.

The properties of convolutional neural networks, including the fact that they are able to retrieve features from multidimensional inputs, turn them into a very interesting alternative for solving problems within the field of computer vision. In fact, computer vision has become a testbed for validating novel techniques and innovations in CNNs.

EMIST comprises both handwritten digits and handwritten letters, is significantly larger than MNIST and data comes from a different source.

Handwritten Digit Recognition System involves reception and interpretation of handwritten digits by a machine. Due to variation in shape and orientation of handwritten digits, it is difficult for a machine to interpret handwritten digits. Handwritten digit Recognition has a wide area of research due to its vast applications like automatic bank cheque processing, billing and automatic postal service. In this thesis, an Offline Handwritten Digit Recognition System is presented. The recognition system is broadly divided into 2 parts, the first part is feature

extraction from handwritten images and the second one is classification of feature vectors into digits. We propose descriptors for handwritten digit recognition based on Histogram of Oriented Gradient (HOG) feature. It is one of the widely used feature vectors for object detection in computer vision. For classification of features, linear Proximal Support Vector Machine Classifier is proposed. This is a binary class classifier which is further converted to a 10 class classifier by means of One against all algorithm. Due to small training time, PSVM classifier is preferable over standard Support Vector Machine (SVM) Classifier. The handwritten images both for training and testing are taken from MNIST database. The performance of the system is measured in terms of Sensitivity, Accuracy, Positive Predictively and Specificity.

2. LITERATURE SURVEY

2.1 Existing problem

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 digitized 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

2.2 References

LITERATURE SURVEY:

AUTHOR	TITLE	CONCEPT	ISSUES

T. Wakabayashi and F. Kimura 2007	Handwritten Digit recognition	Digit recognition is used in post offices, in banks for reading cheques, for license plate recognition The digit recognition can be divided into two groups, printed digit. digit recognition and handwritten digit recognition .	That is on the other hand, there are numerous handwriting styles for the same digit; hence more effort is required to find the accurate handwritten digit.
J.Pradeep,E.Srinivasan and S.Himavathi /2011	Diagonal based feature extraction for handwritten alphabets recognition system using neural network	This is an offline handwritten alphabetical character recognition system using multilayer feed Forward neural network is described in paper.	Here the process of extraction was complicated.
Purohit and Chauhan, 2016	Recognition of Handwritten English Numerals Based on Combining Structural and Statistical Features	The actual character analysis is performed	There are numerous handwriting styles for the same digit, so accuracy plays a major role here.

Ishani Patel,Virag Jagtap ,Ompriya Kale 2015	A Survey on Feature Extraction Methods for Handwritten Digits Recognition	A survey on handwritten digit recognition systems with recent techniques, with three well known classifiers namely MLP, SVM and k-NN used for classification.	This survey has overcome all the difficulties of digit recognition.

2.3 Problem Statement Definition

Handwritten digit recognition is the capability of computer applications 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 shapes and sizes.

The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image. Convolutional Neural Network model created using PyTorch library over the MNIST dataset to recognize handwritten digits.

Handwritten Digit Recognition is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defenses, etc, and classify them. into 10 predefined classes (0-9).

This has been a Content of bottomless-exploration in

the field of deep literacy. Number recognition

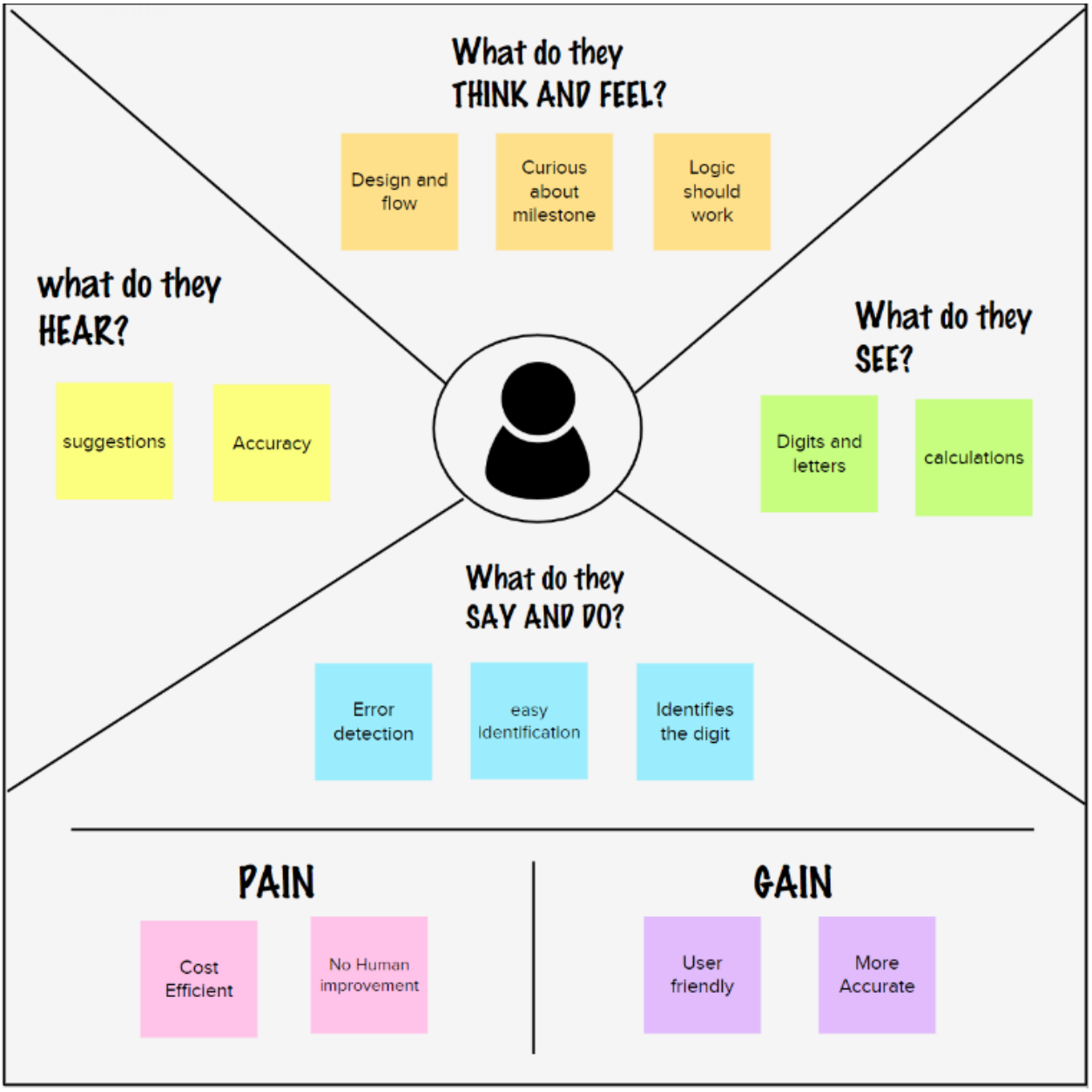
has numerous operations like number plate recognition, postal correspondence sorting, bank check processing, etc. Handwritten number recognition, we face numerous challenges because of different styles of jotting of different peoples as it is not an Optical character recognition.

This exploration provides a comprehensive comparison between different machine literacy and deep literacy algorithms for the purpose of handwritten number recognition. For this, we've used Support. Vector Machine, Multilayer Perceptron, and Convolutional Neural Network. The comparison between these algorithms is carried out on the base of their delicacy, crimes, and testing- training time corroborated by plots and maps that have been

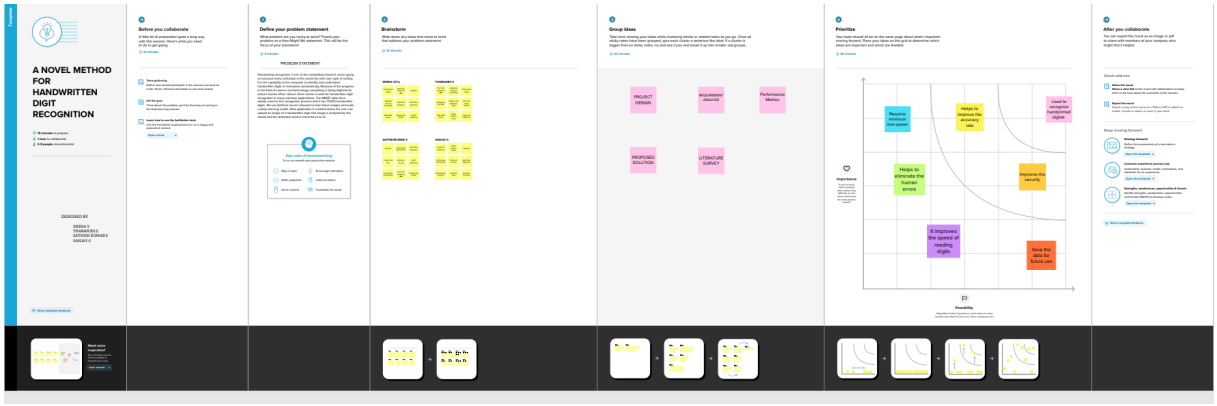
constructed using matplotlib for visualization.

3. IDEATION & PROPOSED SOLUTION

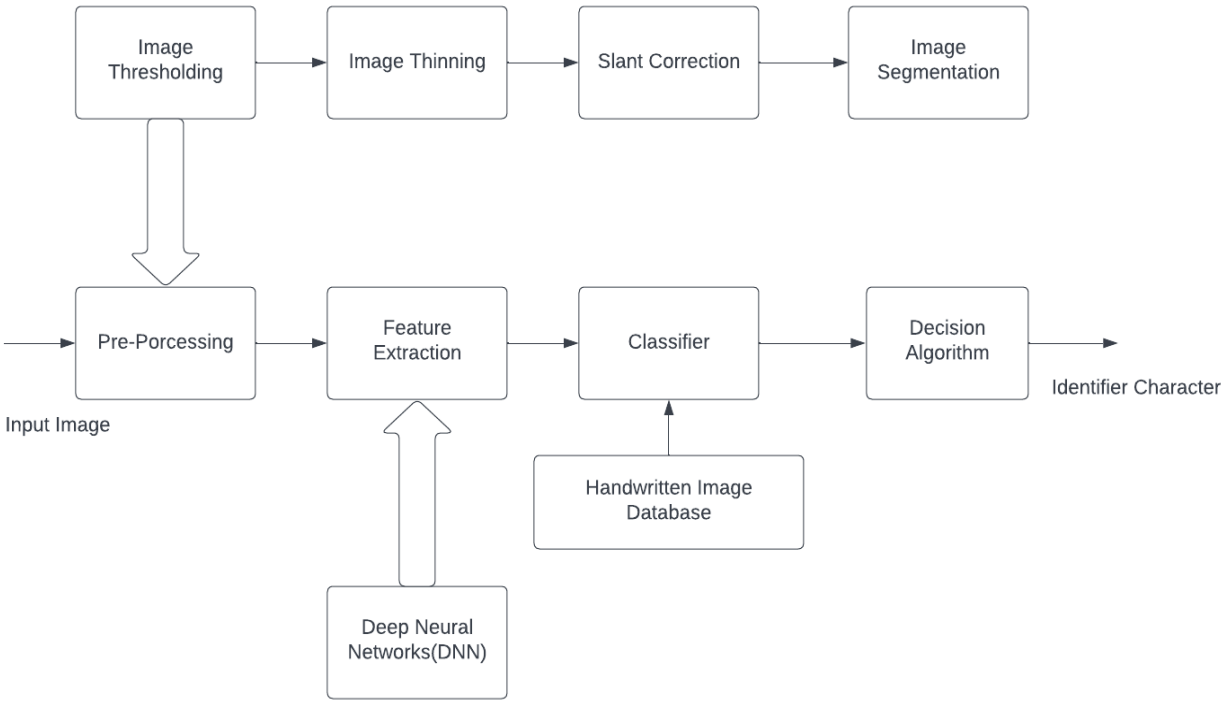
3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



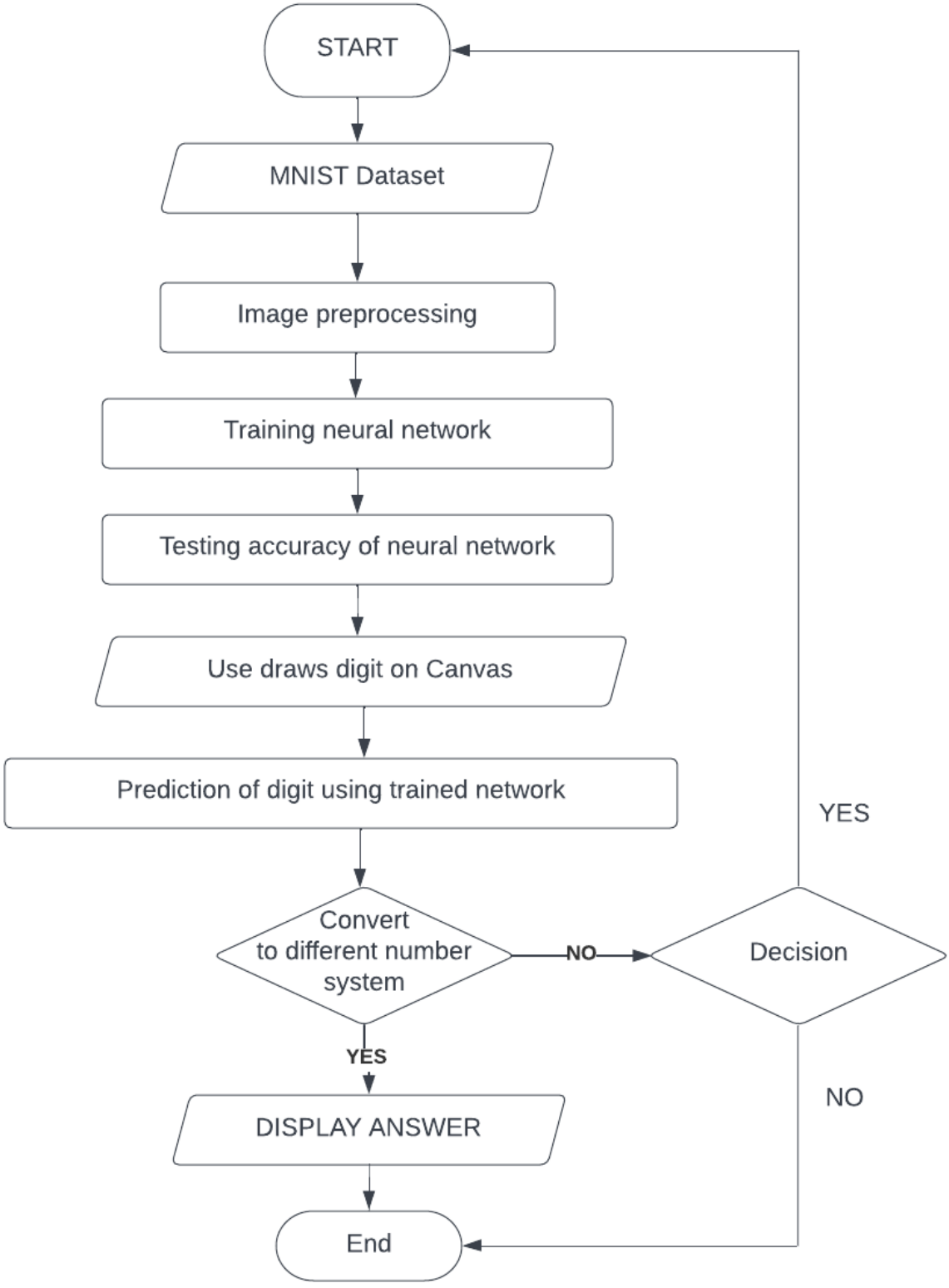
3.3 Proposed Solution



3.4 Problem Solution fit

To reduce error and obtain more efficiency overall, Convolutional Neural Network (CNN) can be used to implement handwritten digit recognition systems. For achieving so, our proposed

system uses CNN with multiple pooling and convolutional layers alongside a kernel of 3x3 size. Our model uses 60,000 28*28 grayscale images during the training process. Our model is trained through a standard 5 epochs to achieve accuracy of the order of 99.16% which is much higher as compared to the traditional algorithms such as SVM, Multilayer Perceptron, Bayes Net, Random Forest, etc. used to implement handwritten digit recognition systems.


START
MNIST Dataset
Image preprocessing
Training neural network
Testing accuracy of neural network
Use draws digit on Canvas
Prediction of digit using trained network
Convert to different number system
NO
Decision
YES
YES
NO
DISPLAY ANSWER
End

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

The point of this venture is to actualize a grouping calculation to perceive manually written character and digit. For example acknowledgment performs the best for all grouping issues reliably. Subsequently, the extent of the undertaking likewise incorporated the rudimentary study the distinctive classifiers and blend techniques, and assess the admonitions around their execution in this specific issue of manually written digit acknowledgement. We could accomplish a precision rate of 78.4%.

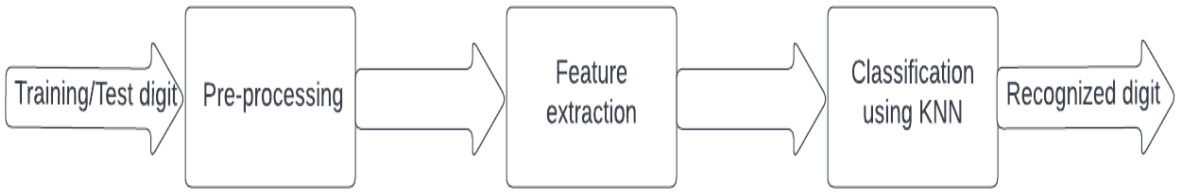
Inputs

Digit, order, pictures

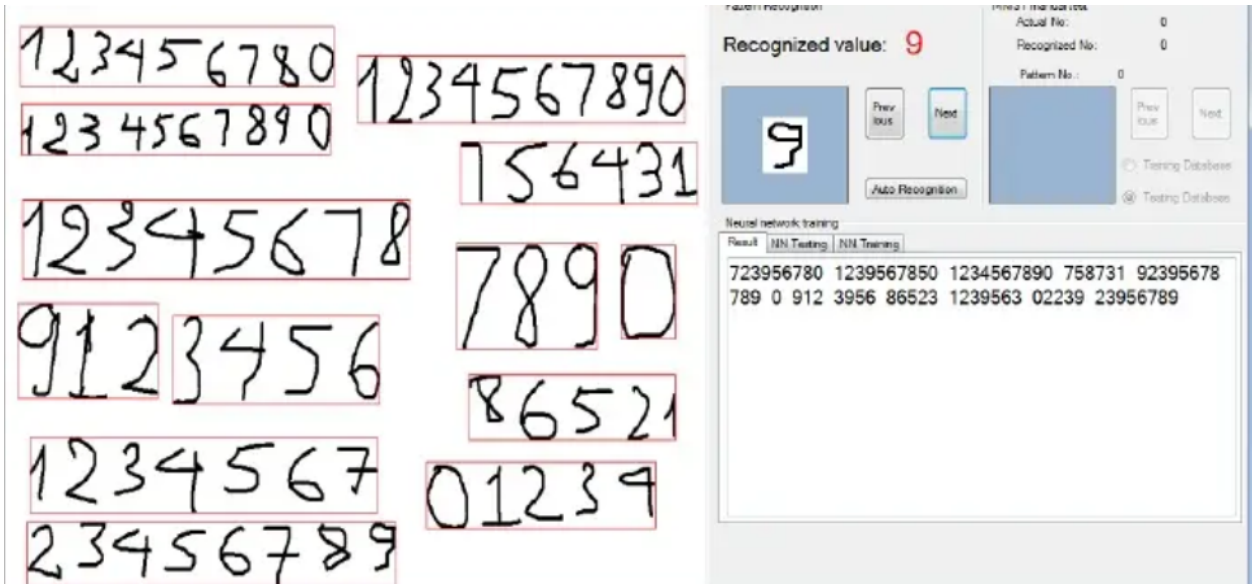
General Constraints, Assumptions and Dependencies

The framework must be prepared completely before use and clients know the framework in particular. When the framework is prepared, before leaving you are required to spare the framework, so the framework is stacked for further use. Also has the upgraded quality.

Processing



OUTPUTS



4.2 Non-Functional requirements

Non-utilitarian essentials may exist for exit running with qualities. A significant part of the time the essentials must be mastered at a structure level of wide rather than at a unit level. State the essentials in the running with locales in quantifiable terms.

Performance

Execution relying upon the connection, high PC execution might include one or a greater amount of the accompanying: Short reaction time for a given bit of work. Execution is described by the measure of valuable work achieved by a PC framework or PC system contrasted with the time and assets utilized.

Reliability

Unwavering quality is a property of any PC related part (programming, or equipment, or a system, for instance) that reliably performs as indicated by its determinations. It has for some time been viewed as one of three related qualities that should be considered when making, purchasing, or utilizing a PC item or part.

Availability

In PC frameworks and systems administration, accessibility is a general term that is utilized to depict the measure of time over a one-year period that the framework assets is accessible in the wake of disappointments in the system. A framework constantly accessible with its all assets are viewed as fruitful.

Security

In registering, security (or PC security) is the strategy for guaranteeing that information put away in a PC can't be perused or bargained by any people without approval. Most PC efforts to establish safety include information encryption and passwords. Information encryption is the interpretation of information into a structure that is indiscernible without a disentangling system. A watchword is a mystery word or expression that gives a client access to a specific project or framework.

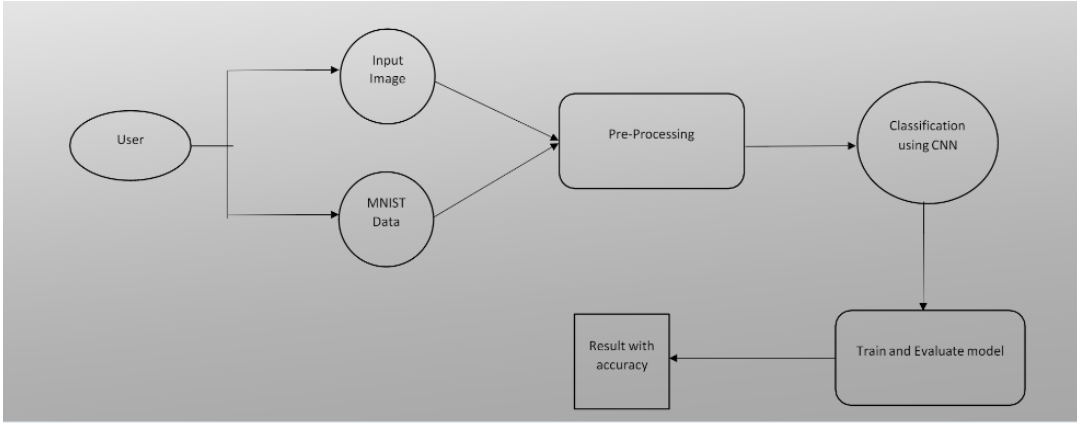
Maintainability

Viability is characterized as the likelihood of performing an effective repair activity inside of a given time. As such, practicality measures the straightforwardness and pace with which a framework can be restored to operational status after a disappointment happens.

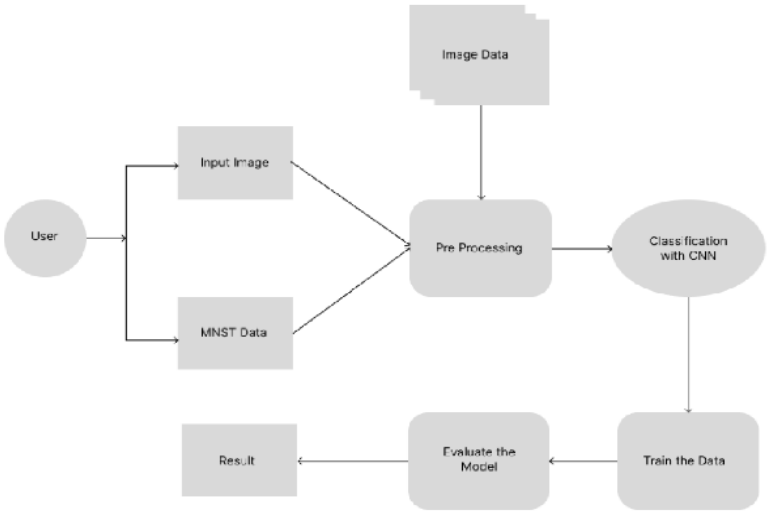
Convenience is a trademark credited to a PC program in the event that it can be utilized as a part of working frameworks other than the one in which it was made without requiring major revamp. Porting is the assignment of doing any work important to make the PC program keep running in the new environment.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



6. CODING & SOLUTIONING (Explain the features added in the project along with code)

6.1 DigitRecogniton.ipynb file

```
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras import layers
keras.backend.set_image_data_format('channels_last')
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, MaxPool2D, Flatten
```

```
import numpy as np
import pandas as pd
```

```
train = pd.read_csv("/content/drive/MyDrive/mnist dataset/train.csv")
test = pd.read_csv("/content/drive/MyDrive/mnist dataset/test.csv")
```

```
train.head()
```

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	..
0	1	0	0	0	0	0	0	0	0	0	..
1	0	0	0	0	0	0	0	0	0	0	..
2	1	0	0	0	0	0	0	0	0	0	..
3	4	0	0	0	0	0	0	0	0	0	..
4	0	0	0	0	0	0	0	0	0	0	..

5 rows × 785 columns

```
test.head()
```

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	.
0	0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	

5 rows × 784 columns

```
image_size = 28*28
image_size
```

```
784
```

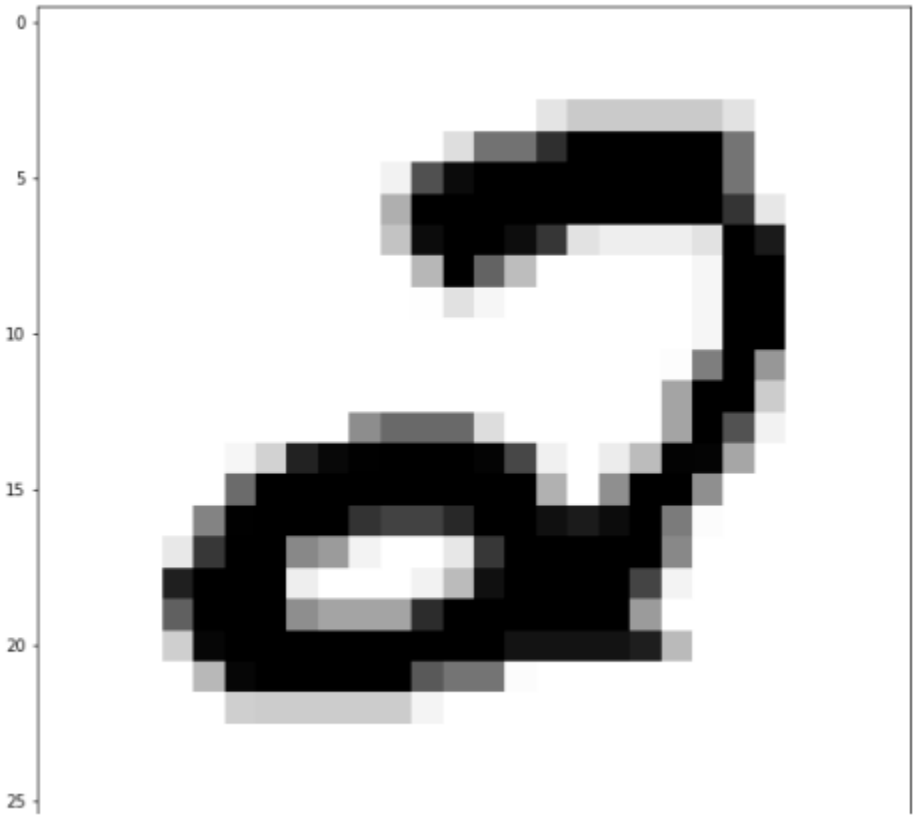
```
X_train = train.drop("label", axis = 1).copy()
X_test = test.copy()
Y_train = train["label"].copy()
```

```
X_train.describe()
```

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	
count	42000.0	42000.0	42000.0	42000.0	42000.0	42000.0	42000.0	42000.0	42000.0	4
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

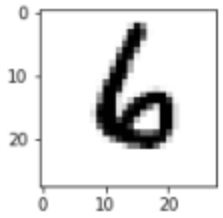
8 rows × 784 columns

```
#Normalize values
X_train = X_train/255.0
X_test = X_test/255.0


#Reshape to 28 * 28 so that we can see the images
X_train = X_train.values.reshape(-1, 28, 28, 1)
X_test = X_test.values.reshape(-1, 28, 28, 1)


import random
no_images=len(X_train)

# Display random Image
fig, ax = plt.subplots(figsize=(10, 10))

plt.imshow(X_train[random.randint(0,no_images), :, :, 0], cmap='Greys', interpolation='nea

# replace random.randint(0,no_images) in code above with a number if you want to see speci
#This dispalys a random image each time

plt.show()
```




```
ig, ax = plt.subplots(figsize=(2,2))
```

```
plt.imshow(X_train[random.randint(0, no_images), :, :, 0], cmap = "Greys", interpolation =

plt.show()
```




```
validation_size = 0.2

X_train, X_val, Y_train, Y_val = train_test_split(X_train, Y_train, test_size = validation_


#Get one-hot encoding
Y_train = keras.utils.to_categorical(Y_train, num_classes = 10)
Y_val = keras.utils.to_categorical(Y_val, num_classes = 10)


def CNN():
    model = Sequential()

   model.add(Conv2D(filters = 32, kernel_size = (4,4), input_shape = (28,28,1), activatio
   model.add(MaxPool2D(pool_size = (2,2)))

   model.add(Flatten())

   model.add(Dense(128, activation = "relu"))
   model.add(Dense(256, activation = "relu"))
   model.add(Dense(512, activation = "relu"))

   model.add(Dense(10, activation = "softmax"))



   return model

model = CNN()


model.compile(optimizer = "Adam", loss = "CategoricalCrossentropy", metrics = "accuracy")


model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 25, 25, 32)        544

 max_pooling2d (MaxPooling2D  (None, 12, 12, 32)       0
 )

 flatten (Flatten)           (None, 4608)              0

 dense (Dense)               (None, 128)               589952

 dense_1 (Dense)             (None, 256)               33024

 dense_2 (Dense)             (None, 512)               131584

 dense_3 (Dense)             (None, 10)                5130

=================================================================
Total params: 760,234
Trainable params: 760,234
Non-trainable params: 0
_________________________________________________________________
```

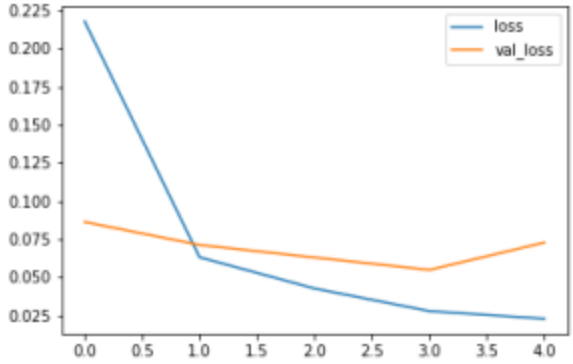
```
from tensorflow.keras.callbacks import EarlyStopping

   early_stop = EarlyStopping(monitor = "val_loss", patience = 1)


   training = model.fit(X_train, Y_train, validation_data = (X_val, Y_val), batch_size = 64,
```

```
Epoch 1/50
525/525 [==============================] - 30s 54ms/step - loss: 0.2180 - accuracy: 0
Epoch 2/50
525/525 [==============================] - 29s 54ms/step - loss: 0.0631 - accuracy: 0
Epoch 3/50
525/525 [==============================] - 26s 49ms/step - loss: 0.0426 - accuracy: 0
Epoch 4/50
525/525 [==============================] - 21s 40ms/step - loss: 0.0277 - accuracy: 0
Epoch 5/50
525/525 [==============================] - 24s 45ms/step - loss: 0.0227 - accuracy: 0
```

```
metrics = pd.DataFrame(model.history.history)
```

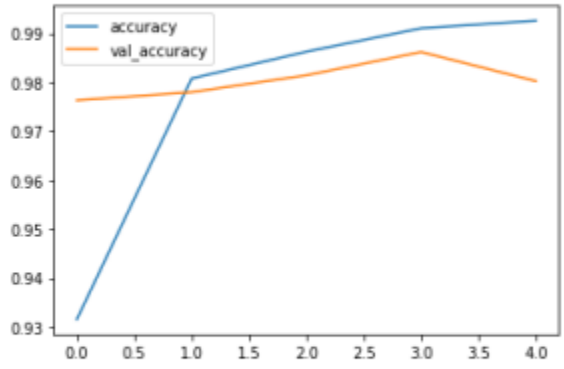
```
metrics[["loss","val_loss"]].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f963fdf09d0>
```




```
metrics[["accuracy", "val_accuracy"]].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f96432e5f50>
```




```
model.metrics_names
```

```
['loss', 'accuracy']
```

```
from sklearn.metrics import classification_report, confusion_matrix
```

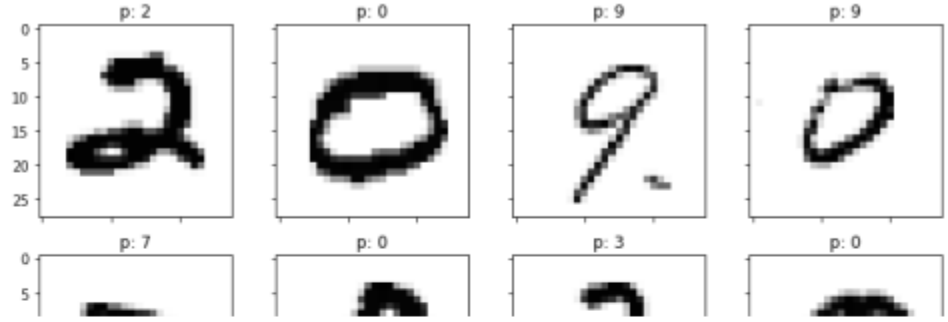
```
Y_val.shape
```

```
(8400, 10)
```

```
def predict(model, X, imgs):
    s = int(np.sqrt(imgs))
    fig, ax = plt.subplots(s, s, sharex=True, sharey=True, figsize=(15, 15))
    ax = ax.flatten()
    preds = model.predict(X[:imgs])
    for i in range(imgs):
        y_pred = np.argmax(preds[i])
        img = X[i].reshape(28, 28)
        ax[i].imshow(img, cmap='Greys', interpolation='nearest')
        ax[i].set_title(f'p: {y_pred}')
```

```
predict(model, X_test, 25)
```

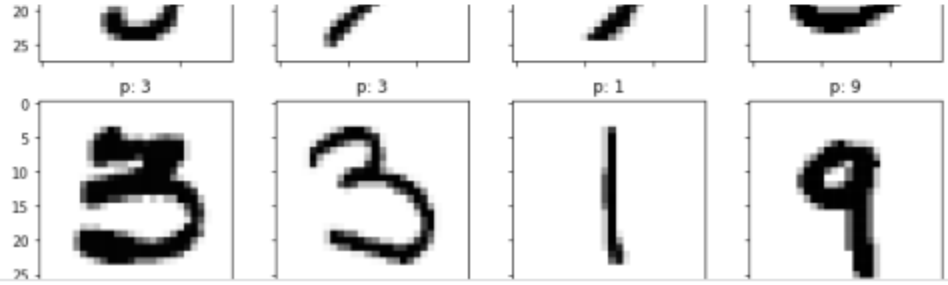
```
1/1 [==============================] - 0s 117ms/step
```

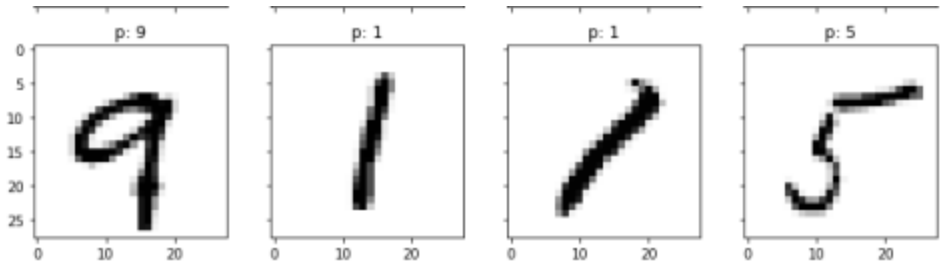



```
y_pred = model.predict(X_test)
y_pred = np.argmax(y_pred, axis=1)
```

```
875/875 [==============================] - 7s 8ms/step
```

```
file_name = "submission.csv"
y_pred = pd.Series(y_pred, name='Label')
sub = pd.concat([pd.Series(range(1, 28001), name="ImageId"), y_pred], axis=1)
sub.to_csv(file_name, index=False)
```





✓ 0s completed at 11:50

6.2

index.html

>

```
<html>

<head>
    <title>Handwritten Digit Recognition Web Application</title>

    <meta name="viewport" content="width=device-width">
    <!-- GoogleFont -->
    <link
href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&displa
y=swap" rel="stylesheet">
    <link
href="https://fonts.googleapis.com/css2?family=Varela+Round&display=s
wap" rel="stylesheet">
    <link
href="https://fonts.googleapis.com/css2?family=Source+Code+Pro:wght@5
00&display=swap" rel="stylesheet">
    <link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:
400,700|Pacifico&display=swap" rel="stylesheet">
    <!-- bootstrap -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstra
p.min.css"
```

```
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x
9JvoRxT2MZw1T" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{
url_for('static',filename='css/style.css') }}">
    <!-- fontawesome -->
    <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+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8
abtTE1Pi6jizo" crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popp
er.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4
sF86dIHNDz0W1" crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.
min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf
4x0xIM+B07jRM" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>

</head>

<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>

    <h1 class="welcome">IBM PROJECT-A Novel Method For Handwritten
Digit Recognition
        <div id="team_id">TEAM ID : PNT2022TMID18806</div>
    </h1>
    <section id="title">
        <h4 class="heading">Handwritten Digit Recognition
Website</h4>
        <br><br>
        <p>
            Here Select the Image to be Predicted and Press the
button 'Read & Predict' to presdict the Digit that is to be
recognized
        </p>

    </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" src="" width="100px" height="100px"
/>
                <div class="buttons_div">
                    <button type="submit" class="btn btn-dark"
id="predict_button">Read & Predict</button>
                    <button type="button" class="btn btn-dark"
id="clear_button">  Clear  </button>
                </div>
            </form>
        </div>
    </section>

</body>
```

```
</html>
```

Predict.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Recognized Digit</title>
</head>

<style>
    body {
        background-image: url('static\images\Tech_Number.jpgs');
        background-repeat: no-repeat;
        background-size: cover;
    }

    #rectangle {
        width: 500px;
        height: 200px;
        background-color: #202de4;
        border-radius: 25px;
        position: absolute;
        top: 25%;
        left: 50%;
        opacity: 0.8;
        transform: translate(-50%, -50%);
    }

    #ans {
        text-align: center;
        font-size: 40px;
        margin: 0 auto;
        padding: 3% 5%;
```

```
        padding-top: 15%;
        color: white;
    }
</style>

<body>
    <div id="rectangle">
        <h1 id="ans">Recognized Digit is : {{num}}</h1>
    </div>
</body>

</html>
```

Index.css

```
#clear_button{
    margin-left: 15px;
    font-weight: bold;
    color: blue;
  }

  #confidence{
    font-family: 'Josefin Sans', sans-serif;
    margin-top: 7.5%;
  }

  #content{
    margin: 0 auto;
    padding: 2% 15%;
    padding-bottom: 0;
  }

  .welcome{
     text-align: center;
     position: relative;
     color: honeydew;
     background-color: silver;
     padding-top: 1%;
     padding-bottom: 1%;
```

```
    font-weight: bold;
    font-family: 'Prompt', sans-serif;
}

#team_id{
    text-align: right;
    font-size: 25px;
    padding-right: 3%;
}

#predict_button{
  margin-right: 15px;
  color: blue;
  font-weight: bold;
}

#prediction_heading{
  font-family: 'Josefin Sans', sans-serif;
  margin-top: 7.5%;
}

#result{
  font-size: 5rem;
}

#title{
  padding: 1.5% 15%;
  margin: 0 auto;
  text-align: center;
}

.btn {
    font-size: 15px;
    padding: 10px;
    -webkit-appearance: none;
    background: #eee;
    border: 1px solid #888;
    margin-top: 20px;
    margin-bottom: 20px;
}
```

```css
.buttons_div{
  margin-bottom: 30px;
  margin-right: 80px;
}

.heading{
  font-family: 'Varela Round', sans-serif;
  font-weight: 700;
  font-size: 2rem;
  display: inline;
}

.leftside{
  text-align: center;
  margin: 0 auto;
  margin-top: 2%;
  /* padding-left: 10%; */
}

#frame{
  margin-right: 10%;
}

.predicted_answer{
  text-align: center;
  margin: 0 auto;
  padding: 3% 5%;
  padding-top: 0;
  /* padding-left: 10%; */
}

p{
  font-family: 'Source Code Pro', monospace,sans-serif;
  margin-top: 1%;
}

@media (min-width: 720px) {
  .leftside{
    padding-left: 10%;
```

```
    }
  }
Footer
© 2022 GitHub, Inc.
Footer navigation
Terms
Privac
```

App.py

```
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 = 'C:\\Users\\sneha\\Desktop\\Digit Recognition Flask
App\\Datasets'


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

model = load_model("mnistData.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"]
```

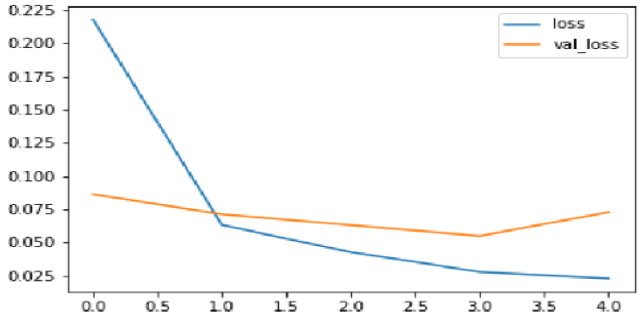
```
        filepath = secure_filename(f.filename)
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))

        upload_img = os.path.join(UPLOAD_FOLDER, filepath)
        img = Image.open(upload_img).convert("L")
        img = img.resize((28, 28))

        im2arr = np.array(img)
        im2arr = im2arr.reshape(1, 28, 28, 1)

        pred = model.predict(im2arr)

        num = np.argmax(pred, axis=1)

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


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

7. RESULTS

7.1 Performance Metrics

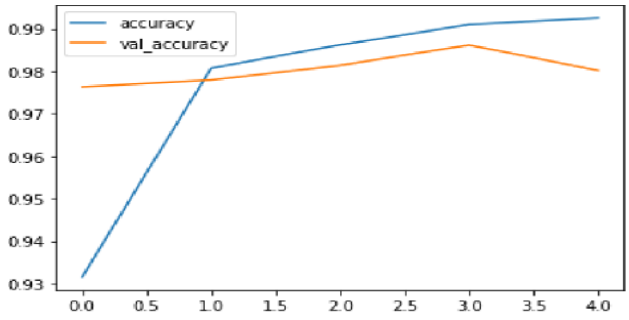
```
metrics[["loss","val_loss"]].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f963fdf09d0>
```




```
metrics[["accuracy", "val_accuracy"]].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f96432e5f50>
```




```
model.metrics_names
```

8. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

1) The system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style.

2) The generative models can perform recognition driven segmentation

3) The method involves a relatively small number of parameters and hence training is relatively easy and fast.

4) Unlike many other recognition schemes, it does not rely on some form of pre-normalization of input images, but can handle arbitrary scalings, translations and a limited degree of image rotation. We have demonstrated that our method of fitting models to images does not get trapped in poor local minima.

DISADVANTAGES:

a) The extensive use of information and communication technology has generated large volumes of data storage.

b) The data repositories might contain a massive amount of useful information. In order to extract useful knowledge from these data repositories for making better decisions, necessitate the need for proper methods of extracting knowledge.

c) Machine learning is an important technique which extracts necessary knowledge and information such as association, patterns, changes and anomalies from various data repositories.

9. CONCLUSION

With the advancement in the AI field a wide range of Modeling of data and collection of data are required in order to create a Best Recognition system. So ,We created an AI model to recognize the Handwritten Digit which helps us to recognize the most complicated Digit with training the Datasets. Support vector machines are one of the basic classifiers that's why it's faster than most algorithms and in this case, gives the maximum training accuracy rate but due to its simplicity, it's not possible to classify complex and ambiguous images as accurately as achieved with MLP and CNN algorithms. We have found that CNN gave the most accurate results for handwritten digit recognition

10. FUTURE SCOPE

The task of handwritten digit recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example - tax forms) and so on.

11. APPENDIX

[1] Fischer, A., Frinken, V., Bunke, H.: Hidden Markovmodels for off-line cursive handwritingrecognition, in C.R. Rao (ed.): Handbook of Statistics 31, 421 – 442, Elsevier, 2013.

[2] Frinken, V., Bunke, H.: Continuous handwrittenscript recognition, in Doermann, D., Tombre, K.(eds.): Handbook of Document Image Processing and Recognition, Springer Verlag, 2014.
[3] S. Günter and H. Bunke. A new combinationscheme for HMM-based classifiers and itsapplication to handwriting recognition. In Proc.16th Int. Conf. on Pattern Recognition, volume 2,pages 332–337. IEEE, 2002.
[4] U.-V. Marti and H. Bunke. Text line segmentationand word recognition in a system for generalwriter independent handwriting recognition. InProc. 6th Int. Conf. on Document Analysis andRecognition, pages 159– 163.
[5] M. Liwicki and H. Bunke, “Iam-ondb - an on-lineEnglish sentence database acquired from thehandwritten text on a whiteboard,” in ICDAR, 2005.
[6] A. Graves and J. Schmidhuber, “Offlinehandwriting recognition with multidimensionalrecurrent neural networks,” in Advances in neural information processing systems, 2009, pp. 545–552.
[7] Nafiz Arica, and Fatos T. Yarman-Vural, —OpticalCharacter Recognition for Cursive Handwriting,‖IEEE Transactions on Pattern Analysis andMachine Intelligence, vol.24, no.6, pp. 801-113,June 2002.
[8] Anita Pal and DavashankarSingh,”Handwritten English Character Recognition Using NeuralNetwork '', International Journal of Computer Science and Communication, pp: 141- 144, 2011.
[9] Sandhya Arora, “Combining Multiple Feature Extraction Techniques for Handwritten Devnagari Character Recognition”, IEEE Region 10 Colloquium and the Third ICIIS, Kharagpur, INDIA, December 2008.
[10]Om Prakash Sharma, M. K. Ghose, Krishna Bikram Shah, “An Improved Zone Based Hybrid Feature Extraction Model for Handwritten Alphabets Recognition UsingEuler Number”, International Journal of Soft Computing and Engineering (ISSN: 2231 - 2307), Vol. 2, Issue 2, pp. 504- 508, May 2012. 29
[11]N. Venkata Rao and Dr.A.S.C.S.Sastry - Optical Character Recognition Technique Algorithms‖-2016 Journal of Theoretical and Applied Information Technology.
[12]Ganapathy, V., & Liew, K. L. (2008). Handwritten Character Recognition Using Multiscale Neural Network Training Technique. World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering, 2(3), 638-643.
[13]Grady, J. O. (2010). System Requirements Analysis. Amsterdam, Netherlands: Elsevier.
[14]Gribaudo, M. (2013). Theory and Application of Multi-Formalism Modeling. Hershey, PA: IGI Global.
[15]Gunawan, T. S., Noor, A. F. R. M., &Kartiwi, M. (2018). Development of english handwritten recognition using deep neural network. Indonesian Journal of Electrical Engineering and Computer Science, 10(2), 562-568.

[16]Hertz, J. A. (2018). Introduction to the theory of neural computation. Boca Raton: CRC Press.
[17]Hamid, N. A., &Sjarif, N. N. A. (2017). Handwritten recognition using SVM, KNN and neural network. arXiv preprint arXiv:1702.00723.
[18]Kumar, P., Saini, R., Roy, P. P., & Pal, U. (2018). A lexicon-free approach for 3D handwriting recognition using classifier combination. Pattern Recognition Letters, 103, 1-7.
[19]Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., &Maglogiannis, I. (2018). Artificial Neural Networks and Machine Learning – ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings. Basingstoke, England: Springer.
[20]Larasati, R., &KeungLam, H. (2017, November). Handwritten digits recognition using ensemble neural networks and ensemble decision tree. In 2017 International Conference on Smart Cities, Automation & Intelligent Computing Systems (ICONSONICS) (pp. 99-104). IEEE.
[21]Lee, J. H., Shin, J., &Realff, M. J. (2018). Machine learning: Overview of the recent progresses and implications for the process systems engineeringfield. Computers & Chemical Engineering, 114,111-121