

# **A novel method for Handwritten Digit Recognition with Neural Networks**

## **Project Report**

### **1.Introduction**

#### **1.1 Project Overview:**

Character recognition is becoming more and more important in the modern world. It helps humans ease their jobs and solve more complex problems. An example is handwritten character recognition [4] which is widely used in the world. This system is developed for zip code or postal code recognition that can be employed in mail sorting. This can help humans to sort mails with postal codes that are difficult to identify. For more than thirty years, researchers have been working on handwriting recognition. Over the past few years, the number of companies involved in research on handwriting recognition [4] has continually increased. The advance of handwriting processing results from a combination of various elements, for example: improvements in the recognition rates, the use of complex systems to integrate various kinds of information, and new technologies such as high quality high speed scanners and cheaper and more powerful CPUs. Some handwriting recognition system allows us to input our handwriting into the system. This can be done either by controlling a mouse or using a third-party drawing tablet. The input can be converted into typed text or can be left as an "ink object" in our own handwriting. We can also enter the text we would like the system to recognize into any Microsoft Office program file by typing. We can do this by typing 1s and 0s. This works as a Boolean variable. Handwriting recognition is not a new technology, but it has not gained public attention until recently. The ultimate goal of designing a handwriting recognition system with an accuracy rate of 100% is quite illusionary, because even human beings are not able to recognize every handwritten text without any doubt. For example, most people cannot even read their own notes. Therefore there is an obligation for a writer to write clearly. In this paper, both Pattern Recognition and Neural Networks will be defined. Examples of types of Pattern Recognition and Neural Networks will be discussed. The advantages of using Neural Networks to recognize handwritten characters will be listed. Finally, Artificial Neural

Networks, using back-Propagation method will be used to train and identify handwritten digits.

## **2.Existing Problem:**

### **2.1 Existing problem:**

It is a hard task for the machine because **handwritten digits are not perfect** and can vary from person to person. Handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image. The MNIST dataset .

### **2.2 References:**


1. Bienenstock E., Doursat R. 1993, "A shape recognition model using dynamical links' Neural: Computation in Neural Systems, Vol. 5, No. 2, pp. 241~258
2. Buffa F., Porceddu I. 1997 "Temperature forecast and dome seeing minimization. A case study using neural network model',
3. Chang H., Fu H., Xu Y. 1990, "Recognition of handwritten similar Chinese characters by self-growing probabilistic decision-based neural network' International Journal of Neural Parallel & Science Computations, Vol. 9, No. 6, pp.545~561
4. Claus, D. "Handwritten Digit Recognition',
5. Fassnacht C., Zippelius A. 1990, "Recognition and categorization in a structured neural network with attractor dynamics' Neural: Computation in Neural Systems, Vol. 2, pp.63~84
6. Hao Y., Shi Y., Zhang D., Zhu X. 2001, "An effective result-feedback neural algorithm for handwritten character recognition' International Journal of Neural Parallel & Science Computations, Vol. 9z No. 2, pp.139~150

### **2.3 Problem Statement Definition:**

The problem statement is to classify handwritten digits. The goal is to take an image of a handwritten digit and determine what that digit is. The digits range from zero (0) through nine (9).

### 3. Ideation and Proposed Solution:


Template



## Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 🕒 10 minutes to prepare
- 🗓️ 1 hour to collaborate
- 👥 2-8 people recommended



**Before you collaborate**

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →

1


**Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes







PROBLEM

Recognize the digits written by the users



### Key rules of brainstorming

To run a smooth and productive session

 Stay in topic.	 Encourage wild ideas.
 Defer judgment.	 Listen to others.
 Go for volume.	 If possible, be visual.

**Brainstorm**  
Write down any ideas that come to mind that address your problem statement.  
10 minutes

**TIP**  
You can select a sticky note and hit the pencil (switch to sketch) icon to start drawing!

**Mohamed afsal**

- A database to export the recognized text
- Remove noise to improve the accuracy
- Various algorithms can be used
- Scan using mobile phone

**Babu**

- Use image or camera as input
- Digits like 3-8, 1-7 should be trained more
- Scanner can be used to scan the handwritten documents
- Dataset should contain the digits of varying styles

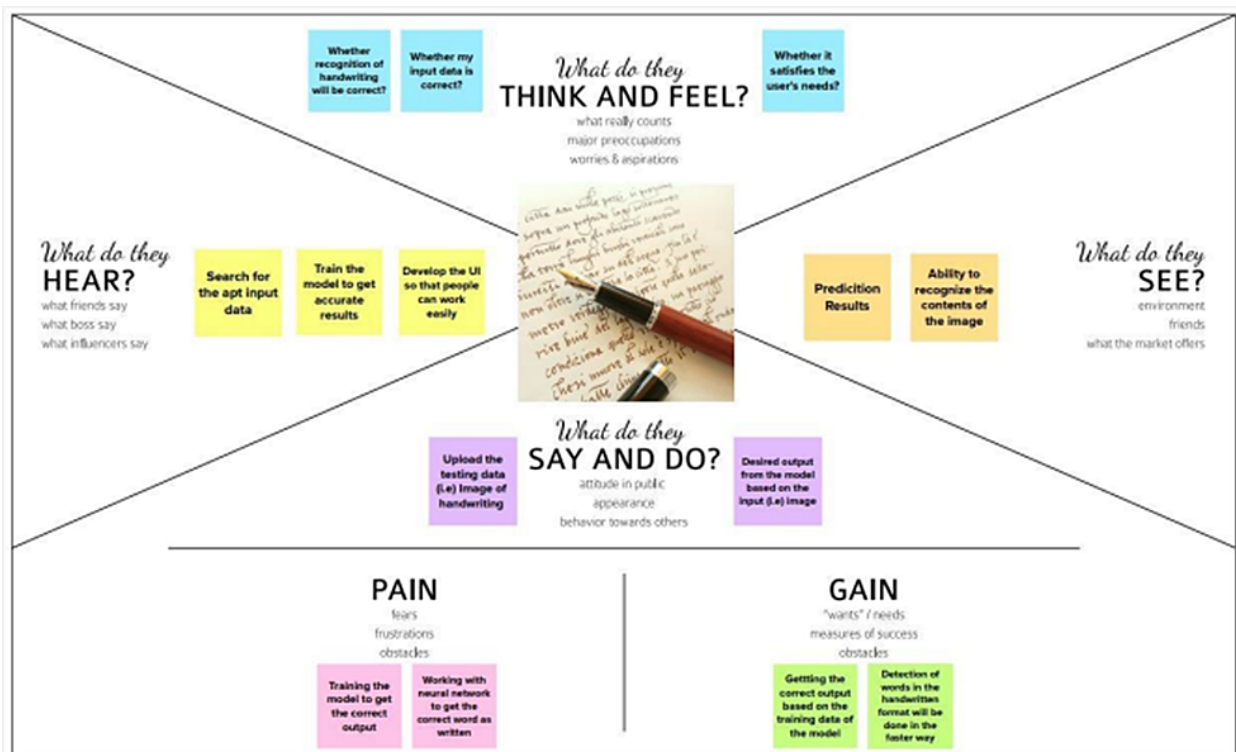
**Suresh Kumar**

- CNN can be used as a model for recognition
- High quality images should be used for training
- Use more images for training
- Train different styles of handwriting

**Mujibur Rahman**

- Segmentation techniques can also be used for recognizing the string of digits
- Feedback from the users can be used for improving the model performance
- Alert user if any digit cannot be scanned
- Test with different styles of handwriting

### 3.1 Empathy map



### 3.2 Literature Survey:

S. No	Title	Year of Publication	Authors	Published on	Theme of Paper	Advantages & Limitations
1.	Exploration of CNN Features for Online Handwriting Recognition.	2020	Subhasis Mandal, S.R. Mahadeva Prasanna and Suresh Sundaram	IEEE	A CNN architecture capable of processing online handwriting without having to convert it to an image.	The proposed CNN characteristics are shown to be effective in character and large.  But, if two words are too near, it is recognized as one word.
2.	An Neural Network based Handwritten Character Recognition system.	2020	S. Mori, C. Y. Suen and Kamamoto	IEEE	It is a type of handwriting recognition that consists of various stages like preprocessing, classification and post processing stages.	This paper presents a novel neural network based offline character recognition system.  But, this does not include feature extraction.
3.	Handwritten Digit Recognition Using KNearest Neighbor Classifier.	2021	Babu, Venkatesh and Chintha	IEEE	To discover minimum distances, a Euclidean minimum distance criterion is utilized, and the digits are classified using a KNN classifier.	The recognition method has an average accuracy of 96.94 percent.  But, The time it takes to classify or estimate something is slow, especially when the training set is huge.

4.	Optical Character Recognition using KNN on Custom Image Dataset.	2021	Hazra, T. K., Singh, D. P., & Daga, N.	IEEE	It works well with multimodal classes due to the fact that its conclusion is based on a small neighborhood of comparable targets.	Regardless of whether the target class is multimodal, the technique can lead to high precision in any instance.  But, the computation cost is rather large.
5.	Handwriting Text Recognition Based on Faster R-CNN.	2020	J. Pradeep	IEEE	Region Proposal Networks (RPN) are a novel network structure that are used for HCR.	This system mainly focuses on maintaining accuracy and also text recognition speed is also increased.  But, if two words are too near, it is recognized as one word.

### 3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Extracting the text from handwritten digits using images.
2.	Idea / Solution description	Various AI/ML packages are used to analyse, train the model based on the problem statement.
3.	Novelty / Uniqueness	It can recognize different varieties of handwriting patterns.
4.	Social Impact / Customer Satisfaction	This system can be used to recognize the number plates in vehicles, postal codes & phone numbers etc...
5.	Business Model (Revenue Model)	We are building a service that can be integrated into any application and pricing model will be based on the usage of the model.
6.	Scalability of the Solution	This can be extended to recognizing letters in English as a base testing and even extended to different languages.

### 3.4 Problem Solution fit:

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> Who is your customer? i.e. working parents of 0-5 y.o. kids  Users who need to recognize the handwritten format which couldn't recognize by them	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.  Quality scanners are needed for the scanning the handwritten format, stable network connection for recognition process.	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking.  It is for more efficient to do it digitally rather than doing it manually then converting into digital format	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> Which jobs to be done (or problems) do you address for your customers? There could be more than one; explore different sides.  * Mapping the right digits based on the recognition * Differentiating digits when they are joined together. * Able to recognize different calligraphic styles.	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations.  Customers may use this software because they may feel difficult to understand the digit format.	<b>7. BEHAVIOUR</b> <span>BE</span> What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)  Customers may approach this software for recognize the digits, pay for it based on the payment scheme and can solve their existing problem.	Focus on J&P, fit into BE, understand RC
Identify strong TR & EM	<b>3. TRIGGERS</b> <span>TR</span> What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.  Vendors might implement their additional functionalities into our software  <b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in-control - use it in your communication strategy & design.  Vendors might feel difficult to understand the handwritten digits and get frustrated but they feel delighted after they use our software.	<b>10. YOUR SOLUTION</b> <span>SL</span> If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits really. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.  Our solution is to give a software for recognizing the handwritten digits which may solve the problems that are faced by a variety of people / vendor / enterprises in the society.	<b>8. CHANNELS of BEHAVIOUR</b> <span>CH</span> <b>1. ONLINE</b> What kind of actions do customers take online? Extract online channels from #7  <b>2. OFFLINE</b> What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.  1. Customers may give their images as an input through online. 2. Customers use their scanning devices to scan the digit format for the recognition process.	Identify strong TR & EM

## 4. Requirement Analysis:

### 4.1 Functional Requirements:

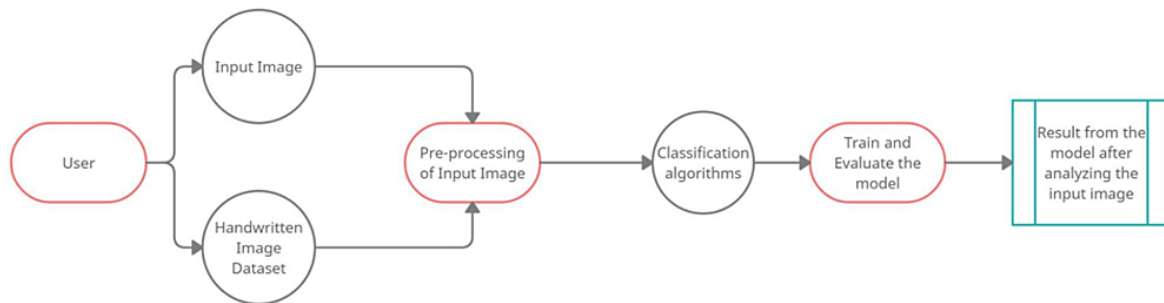
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form. Registration through Gmail.
FR-2	Uploading Images	Image upload from local file directory.
FR-3	Extracting Text	Identifying handwriting patterns and extracting text from it.

### 4.2 Non-functional Requirements:

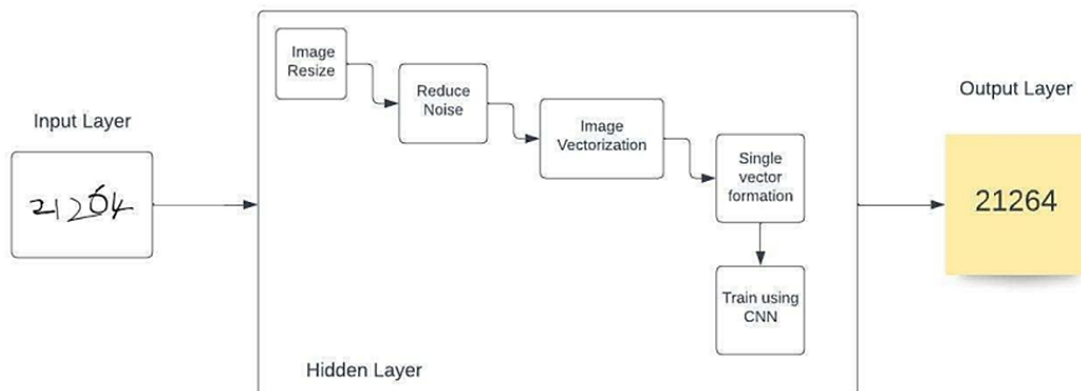
FR No.	Non-Functional Requirement	Description
NFR-1	<b>Usability</b>	Easy to upload the image and view the processed output.
NFR-2	<b>Security</b>	Basic authentication for users using various safety mechanisms like JWT Token, etc.
NFR-3	<b>Reliability</b>	Output produced by the application has high accuracy.
NFR-4	<b>Availability</b>	The application can be deployed on the cloud which makes it possible to access the application from anywhere at any time.
NFR-5	<b>Scalability</b>	Large size of image file support can be provided.

## 5. Project Design:

### 5.1 Data flow diagrams:



### 5.2 Solution and technical architecture:



### 5.3 component and technologies:



S.No	Component	Description	Technology
1.	User Interface	How user interacts with application	React JS
2.	Application Logic-1	Logic for a process in the application	Python
3.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
4.	Machine Learning Model	Purpose of Machine Learning Model	Image Recognition Model (CNN)
5.	Infrastructure (Server / Cloud)	Application Deployment on Cloud	IBM Cloud

## 5.4 Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Keras, tensorflow, pandas, numpy
2.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Flask Microservice

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Keras, tensorflow, pandas, numpy
2.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Flask Microservice

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

### Velocity:

Imagine we have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \text{sprint duration} / \text{Velocity}$$

$$= 20 / 6 = 3.33$$

## 6.2 Milestone and Activity List:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Data Collection & pre processing	USN-1	As a user, I can upload any kind of image with the pre-processing step is involved in it.	10	High
Sprint-1		USN-2	As a user, I can upload the image in any resolution.	10	High
Sprint-2	Building the Machine learning model	USN-3	As a user, I will get a application with ML model which provides high accuracy of recognized handwritten digit	3	Medium
Sprint-2		USN-4	As a user, I can pass the handwritten digit image for recognizing the digit.	2	Medium

### Sprint -1:

Import the necessary package:

```
import matplotlib.pyplot as plt
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
```

Load the data::

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

Data Analysis:

```
print(X_train.shape)
```

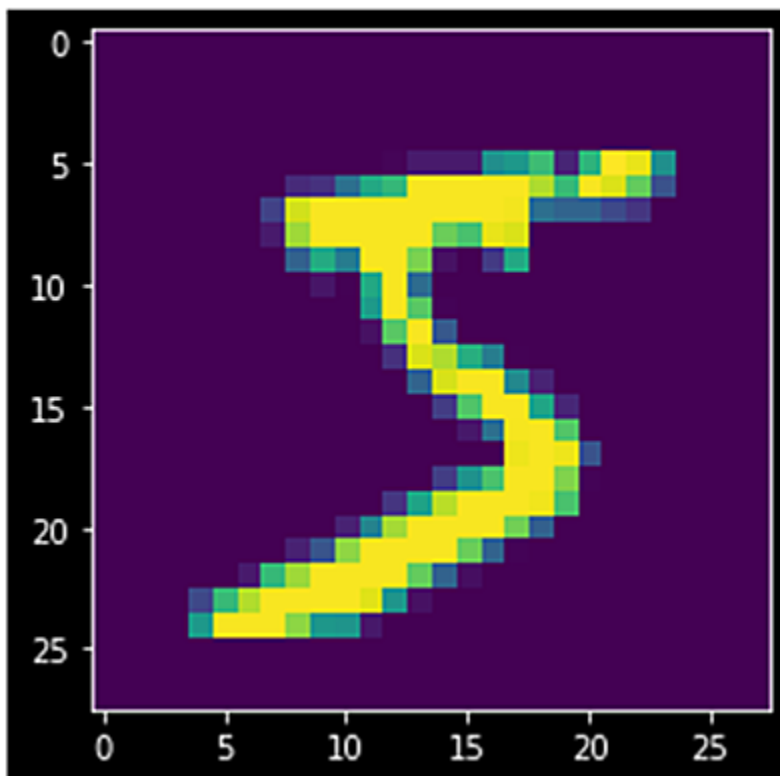
```
print(X_test.shape)
```

```
X_train[0]
```

```
y_train[0]
```

```
plt.imshow(X_train[0])
```

```
<matplotlib.image.AxesImage at 0x299039b90f0>
```



Data Pre-processing:

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

## **Sprint -2:**

### **Create Model:**

```
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"))
```

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

### **Train Model:**

```
model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test, Y_test))
```

```
Epoch 1/5
1875/1875 [=====] - 13s 5ms/step - loss: 0.2126 - accuracy: 0.9506 - val_loss: 0.1034 - val_accuracy: 0.9682
Epoch 2/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.0670 - accuracy: 0.9797 - val_loss: 0.0881 - val_accuracy: 0.9750
Epoch 3/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.0442 - accuracy: 0.9855 - val_loss: 0.1156 - val_accuracy: 0.9713
Epoch 4/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.0341 - accuracy: 0.9894 - val_loss: 0.0914 - val_accuracy: 0.9767
Epoch 5/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.0267 - accuracy: 0.9920 - val_loss: 0.0862 - val_accuracy: 0.9802

<keras.callbacks.History at 0x1f0ea5c37f0>
```

### **Test the Model:**

```
metrics = model.evaluate(X_test, Y_test, verbose=0)
```

```
print("Metrics (Test Loss & Test Accuracy): ")
print(metrics)
```

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

```
1/1 [=====] - 0s 264ms/step
[[8.46943826e-13 1.57253368e-19 1.96990776e-14 3.01160138e-12
 1.78030464e-18 4.28635279e-16 1.02099006e-19 1.00000000e+00
 2.31007786e-13 1.16059251e-09]
 [3.43382928e-13 7.29512642e-13 1.00000000e+00 2.59724435e-18
 7.18828121e-19 4.43095160e-20 1.57180150e-12 2.10340672e-20
 9.12680796e-15 2.57497593e-20]
 [7.42934214e-10 9.99712765e-01 3.03818706e-06 6.55358634e-13
 1.32370133e-05 4.26156277e-10 6.16142026e-10 1.36882345e-05
 2.57250038e-04 1.04902729e-12]
 [9.99999762e-01 2.01685658e-18 1.22698598e-08 2.35469518e-14
 3.93878913e-13 1.61292490e-09 1.53220476e-08 1.24054740e-08
 5.34298192e-13 2.85961761e-07]]
```

```
print(numpy.argmax(prediction, axis=1))
print(Y_test[:4])
```

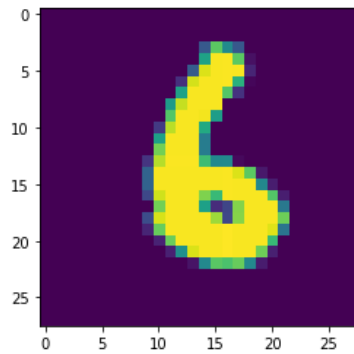
```
[7 2 1 0]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

### Sprint 3:

Image in 36th position in training dataset

```
import matplotlib.pyplot as plt
plt.imshow(x_train[36])
```

```
<matplotlib.image.AxesImage at 0x7f1b52d77710>
```



## Reshaping the data :

As we are using Deep learning neural network, the input for this network to get trained on should be of higher dimensional. Our dataset is having three-dimensional images so we have to reshape them too higher dimensions

```
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:

```
no_of_classes=10
y_train=np_utils.to_categorical(y_train,no_of_classes)
y_test=np_utils.to_categorical(y_test,no_of_classes)
y_test[3]

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

## 7. Coding and Solutioning:

### Index:

The index. html page is the most common name used for the default page shown on a website if no other page is specified when a visitor requests the

site. In other words, index. html is the name used for the homepage of the website.

## home.html

```
<html>
  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
  />
    <title>Handwritten Digit Recognition</title>
    <link rel="icon" type="image/svg" sizes="32x32"
href="{{url_for('static',filename='images/icon.svg')}}" />
    <link rel="stylesheet" href="{{url_for('static',filename='css/main.css')}}" />
    <script src="https://unpkg.com/feather-icons"></script>
    <script defer src="{{url_for('static',filename='js/script.js')}}"></script>
  </head>
  <body>
    <div class="container">
      <div class="heading">
        <h1 class="heading__main">Handwritten Digit Recognizer</h1>
        <h2 class="heading__sub">Easily analyze and detect handwritten
digits</h2>
      </div>
      <div class="upload-container">
        <div class="form-wrapper">
          <form class="upload" action="/predict" method="post"
enctype="multipart/form-data">
            <label id="label" for="upload-image"><i data-feather="file-
plus"></i>Select File</label>
            <input type="file" name="photo" id="upload-image" hidden />
            <button type="submit" id="up_btn"></button>
          </form>
          
        </div>
      </div>
    </div>
  </body>
</html>
```

## Predict .html

```
<html>
  <head>
    <title>Prediction | Handwritten Digit Recognition</title>
    <link rel="stylesheet" href="{{url_for('static',filename='css/predict.css')}}" />
    <link rel="icon" type="image/svg" sizes="32x32"
href="{{url_for('static',filename='images/icon.svg')}}" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  </head>
  <body>
    <div class="container">
      <h1>Prediction</h1>
      <div class="result-wrapper">
        <div class="input-image-container">
          
        </div>
        <div class="result-container">
          <div class="value">{{best.0}}</div>
          <div class="accuracy">{{best.1}}%</div>
        </div>
      </div>
      <h1>Other Predictions</h1>
      <div class="other_predictions">
        {% for x in others %}
          <div class="value">
            <h2>{{x.0}}</h2>
            <div class="accuracy">{{x.1}}%</div>
          </div>
        {% endfor %}
      </div>
    </div>
  </body>
</html>
```

```
        </div>
    </div>
</body>
</html>
```

## Flask code:(app.py)

```
from flask import Flask,render_template,request
from recognizer import recognize
```

```
app=Flask(__name__)
```

```
@app.route('/')
def main():
```

```
    return render_template("home.html")
```

```
@app.route('/predict',methods=['POST'])
```

```
def predict():
```

```
    if request.method=='POST':
```

```
        image = request.files.get('photo', "")
```

```
        best, others, img_name = recognize(image)
```

```
        return render_template("predict.html", best=best, others=others,
img_name=img_name)
```

```
if __name__=="__main__":
```

```
    app.run()
```

## Recognizer.py

```
import os
```

```
import random
```



```

import string
from pathlib import Path
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps

def random_name_generator(n: int) -> str:
    """
    Generates a random file name.

    Args:
        n (int): Length the of the file name.

    Returns:
        str: The file name.
    """
    return "".join(random.choices(string.ascii_uppercase + string.digits, k=n))

def recognize(image: bytes) -> tuple:
    """
    Predicts the digit in the image.

    Args:
        image (bytes): The image data.

    Returns:
        tuple: The best prediction, other predictions and file name
    """

    model=load_model(Path("./model/model.h5"))

    img = Image.open(image).convert("L")

    # Generate a random name to save the image file.

```

```
img_name = random_name_generator(10) + '.jpg'
if not os.path.exists(f"./static/data/"):
    os.mkdir(os.path.join('./static/', 'data'))
img.save(Path(f"./static/data/{img_name}"))
```

**# Convert the Image to Grayscale, Invert it and Resize to get better prediction.**

```
img = ImageOps.grayscale(img)
img = ImageOps.invert(img)
img = img.resize((28, 28))
```

**# Convert the image to an array and reshape the data to make prediction.**

```
img2arr = np.array(img)
img2arr = img2arr / 255.0
img2arr = img2arr.reshape(1, 28, 28, 1)
```

```
results = model.predict(img2arr)
best = np.argmax(results,axis = 1)[0]
```

**# Get all the predictions and it's respective accuracy.**

```
pred = list(map(lambda x: round(x*100, 2), results[0]))
```

```
values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
others = list(zip(values, pred))
```

**# Get the value with the highest accuracy**

```
best = others.pop(best)
```

```
return best, others, img_name
```

**output:**

# Handwritten Digit Recognizer

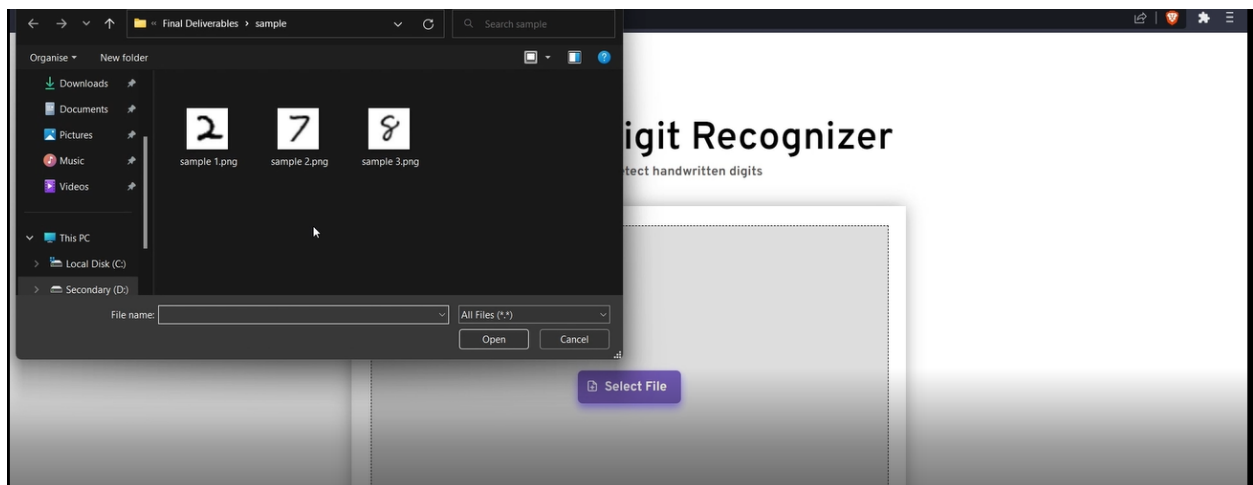
Easily analyze and detect handwritten digits

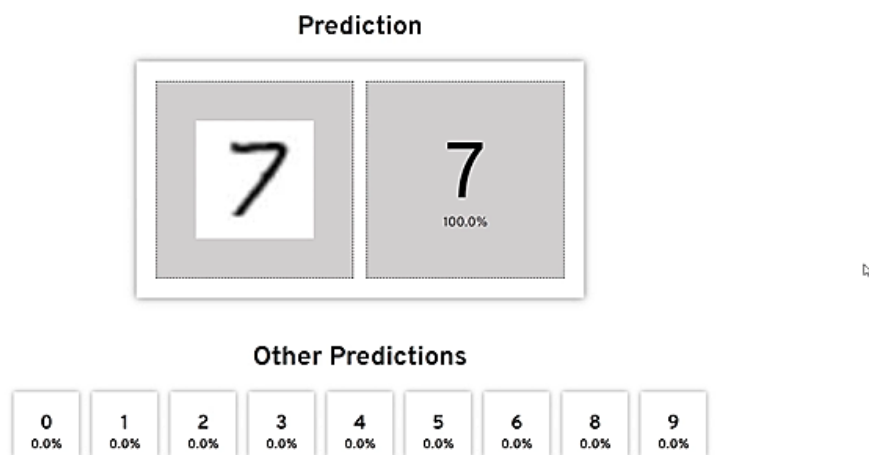
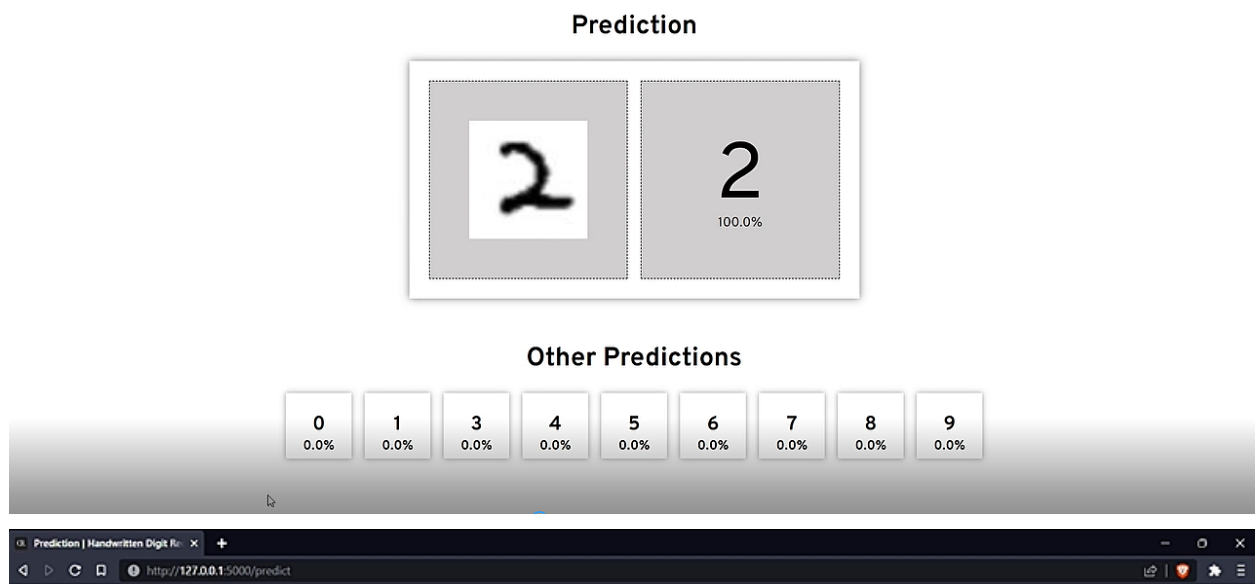
Select File

Object Demo



Handwritten Digit Recognizer





---

## 8. Testing:

```
import cv2
import numpy as np
from keras.datasets import mnist
from keras.layers import Dense, Flatten, MaxPooling2D, Dropout
from keras.layers.convolutional import Conv2D
from keras.models import Sequential
from tensorflow.keras.utils import to_categorical
```

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

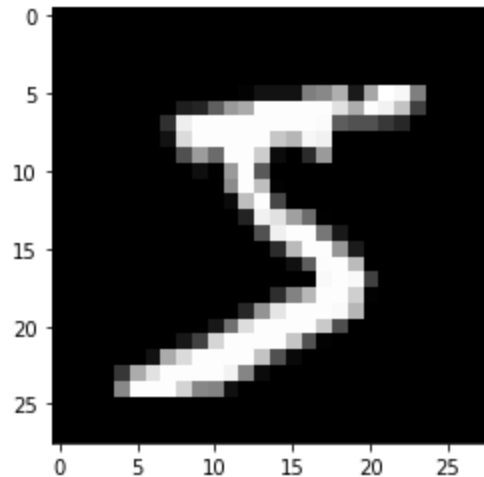
```
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

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

```
plt.imshow(X_train[0], cmap="gray")
```

```
plt.show()
```

```
print (y_train[0])
```



5

```
print ("Shape of X_train: {}".format(X_train.shape))
```

```
print ("Shape of y_train: {}".format(y_train.shape))
```

```
print ("Shape of X_test: {}".format(X_test.shape))
```

```
print ("Shape of y_test: {}".format(y_test.shape))
```

```
Shape of X_train: (60000, 28, 28)
```

```
Shape of y_train: (60000,)
```

```
Shape of X_test: (10000, 28, 28)
```

```
Shape of y_test: (10000,)
```

```
# Reshaping so as to convert images for our model
```

```

X_train = X_train.reshape(60000, 28, 28, 1)
X_test = X_test.reshape(10000, 28, 28, 1)
print ("Shape of X_train: {}".format(X_train.shape))
print ("Shape of y_train: {}".format(y_train.shape))
print ("Shape of X_test: {}".format(X_test.shape))
print ("Shape of y_test: {}".format(y_test.shape))

```

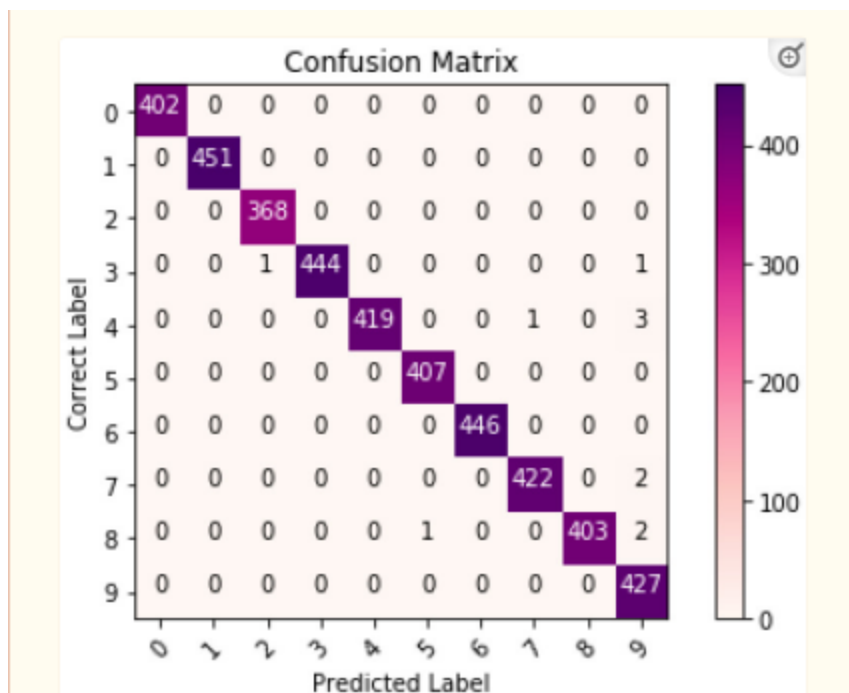
```

Shape of X_train: (60000, 28, 28, 1)
Shape of y_train: (60000,)
Shape of X_test: (10000, 28, 28, 1)
Shape of y_test: (10000,)

```

## 9. Results:

### Performance metrics:



Accuracy: 0.8666666666666667

Precision:

0.8666666666666667 Recall:

0.8666666666666667

Specificity0.8666666666666667

F1 score: 0.8666666666666667

## **10. Advantages and Disadvantages:**

### **Advantages:**

This approach has many 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 ...

### **Disadvantages:**

The issue is that there's a wide range of handwriting – good and bad. This makes it tricky for programmers to provide enough examples of how every character might look. Plus, sometimes, characters look very similar, making it hard for a computer to recognise accurately.

## **11. Conclusion:**

In this paper, we presented a novel convolutional neural network architecture based on data preparation, receptive field, data augmentation, optimization, normalization, and regularization techniques for handwritten digit recognition. To guarantee the dataset does not contain any unnecessary details and that it is fit for applying in our CNN model, data preparation is conducted as an essential first step in our proposed model. Without applying data preparation to the raw data, it is highly possible that unnecessary data leads to misleading results. In our work, filter sizes are determined by calculating the size of the ERF. Calculating this size can help in enhancing the performance of our CNN. In ERF, the process is started with a proposed filter to convolve the input image and gain its feature map. Then, this process is repeated with the next layers to gain deeper feature maps until getting the output image with an effective receptive field with a size of  $22 \times 22$ . Max-pooling with a  $2 \times 2$  filter and stride = 2 is used to increase the size of the receptive field. Proposed CNN architecture has achieved recognition accuracy of 99.98% on the MNIST handwritten digit dataset, and 99.40% with the same dataset contaminated with 50% noise. Our experimental results show that using data augmentation with CNN gives better recognition accuracy compared to CNN without

data augmentation. Utilizing the data augmentation technique has helped to expand our training dataset, resulting in improving the performance and classification capability of our model. We also presented the RMSprop optimizer to restrict the oscillations in the vertical direction and take larger steps in the horizontal direction in order to converge substantially faster to the global minimum point. On the very competitive MNIST handwritten digits benchmark, our proposed CNN model has achieved superiority over state-of-the-art methods for handwritten digits recognition. In our experiments, batch normalization has been used to improve the training performance and enhance the stability of our model. With the usage of batch normalization, we can speed up the training, reduce training and testing time, in addition to lowering the sensitivity initialization. In order to avoid overfitting and underfitting, an early stopping technique has used to determine the optimal number of training epochs.

## **12. Future Works:**

We believe that our proposed model can further be applied to other datasets. In contrast, as a future work, we find that it is worth taking further actions to improve our model performance in terms of how to perfectly learn and extract the local features in the hidden layers, and how to enhance the recognition ability in the fully connected layers to avoid mislabeling problems.

## **13.Appendix:**

### **Project Demo Link:**

[https://drive.google.com/file/d/1F1gJ0hgMUJbstja9N66XA3211EnbR33V/view?usp=share\\_link](https://drive.google.com/file/d/1F1gJ0hgMUJbstja9N66XA3211EnbR33V/view?usp=share_link)

<https://github.com/IBM-EPBL/IBM-Project-44676-1660726085>



