PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

Submitted By

PNT2022TMID22980

Bharath Kumar A S - 913119104013

Aswin Vijay T V - 913119104301

Barath Balaji V S - 913119104009

Shiva Rakesh S - 913119104096

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Project Overview

Machine learning and deep learning play an important role in computer technology and Artificial Intelligence. With the use of Deep Learning and Machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas.

Handwritten Digit Recognition is the ability of Computer systems to recognize handwritten digits from various sources, such as images, documents, and so on. This project aims to let users take advantage of machine learning to reduce manual tasks in recognizing digits.

1.2 Purpose

Digit Recognition system are capable of recognizing the digits from different sources like emails, bank cheques, papers, images, etc. and in different real-world scenarios for online handwriting recognition on computer, tablets or systems, recognize number plates of vehicles, processing bank cheque amounts, numeric enteries in forms filled up by hand (tax forms) and so on.

2. LITERATURE SURVEY

2.1 Existing problem

- ✓ The different architectures of CNN, hybrid CNN,CNN - RNN and CNNHMM models, and domain - specific recognition system, are not thoroughly inquired and evolutionary algorithms are not clearly explored for optimizing CNN learning parameters ,the number of layers, learning rate and kernel sizes of convolutional filters.

- ✓ The fluctuation of accuracies for handwritten digits was observed for 15 epochs by varying the hidden layers. There is no clear explanation given for observing variation in the overall classification accuracy by varying the number of hidden layers and batch size.

2.2 References

S.NO	Author Name	Paper Title	Journal/ Conference title	Page No/ Volume No	Year of Publication	Description
	Savita Ahlawat, Amit Choudhary, Anand Nayyar, Saurabh Singh and Byungun Yoon.	Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)	IEEE Sensors Journal		2020	In this paper, with the aim of improving the performance of handwritten digit recognition, they valuated variants of a convolutional neural network to avoid complex preprocessing, costly feature extraction and a complex ensemble (classifier combination) approach

						of a traditional recognition system.
	Vijayalaxmi R Rudraswamimath, Bhavanishankar and Channasandra.	Handwritten Digit Recognition using CNN	International Journal of Innovative Science and Research Technology	Volume-4 Issue-6	2019	In this paper, the most widely used Machine learning algorithms, KNN, SVM, RFC and CNN have been trained and tested on the same data in order acquire the comparison between the classifiers
	Fathma Siddique, Shadman Sakib and Md. Abu Bakr Siddique.	Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers	5th International Conference on Advances in Electrical Engineering (ICAEE)		2019	In this paper, they observed the variation of accuracies of CNN to classify handwritten digits for 15 epochs using various numbers of hidden layers and epochs and

						to make the comparison between the accuracies. For this performance evaluation of CNN, they performed the experiment using Modified National Institute of Standards and Technology(MN IST) dataset.
	Akanksha Gupta, Ravindra Pratap Narwaria and Madhav Singh	Review on Deep Learning Handwritten Digit Recognition using Convolutional Neural Network	International Journal of Recent Technology and Engineering (IJRTE)	Volume -9 Issue-5	2021	In this paper, Object Character Recognition (OCR) is used on printed or documented letters to convert them into text. The database has training image database of 60,000 images and

						testing image database of 10,000 images. The KNN algorithm describes categorical value by making use of majority of votes of K - nearest neighbors, the K value used to differ here.
	Md. Anwar Hossain and Md. Mohon Ali	Recognition of Handwritten Digit using Convolutiona l Neural Network (CNN)	Global Journal of Computer Science and Technology: D Neural & Artificial Intelligence	Volume 19 Issue2	2019	The goal of this work will be to create a model that will be able to identify and determine the handwritten digit from its image with better accuracy using using the concepts of Convolution al Neural Network and MNIST

						dataset. Later it can be extended for character recognition and real-time person’s handwriting. The results can be made more accurate with more convolution layers and more number of hidden neurons.

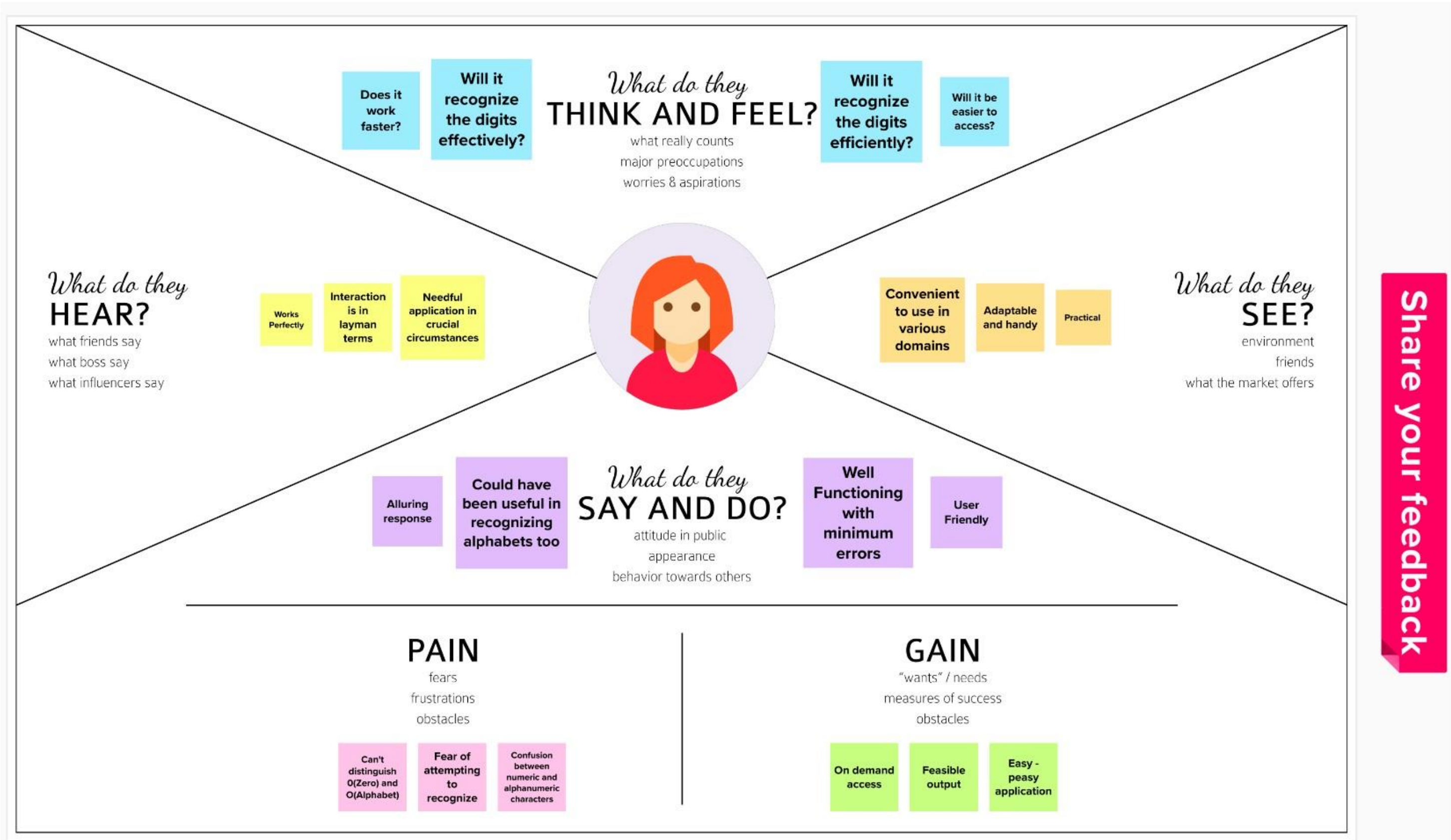
2.3 Problem Statement Definition

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.

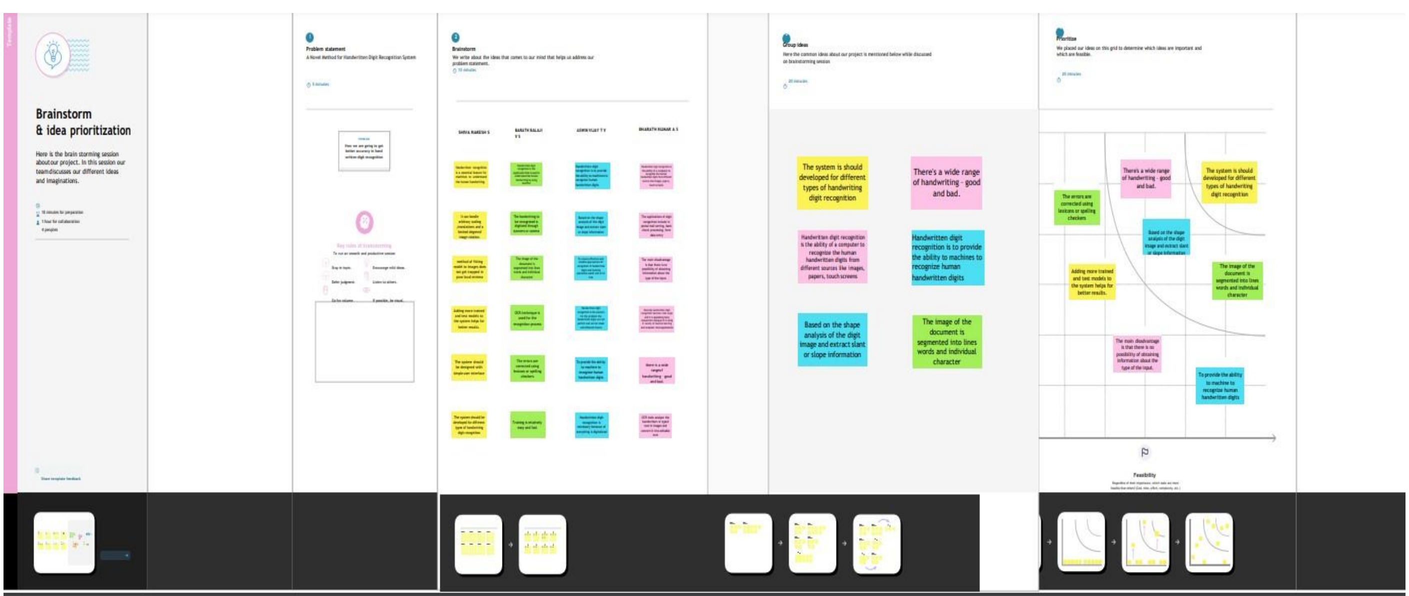
Hence, there comes a need for handwritten digit recognition in many real time applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created 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(User Interface).

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



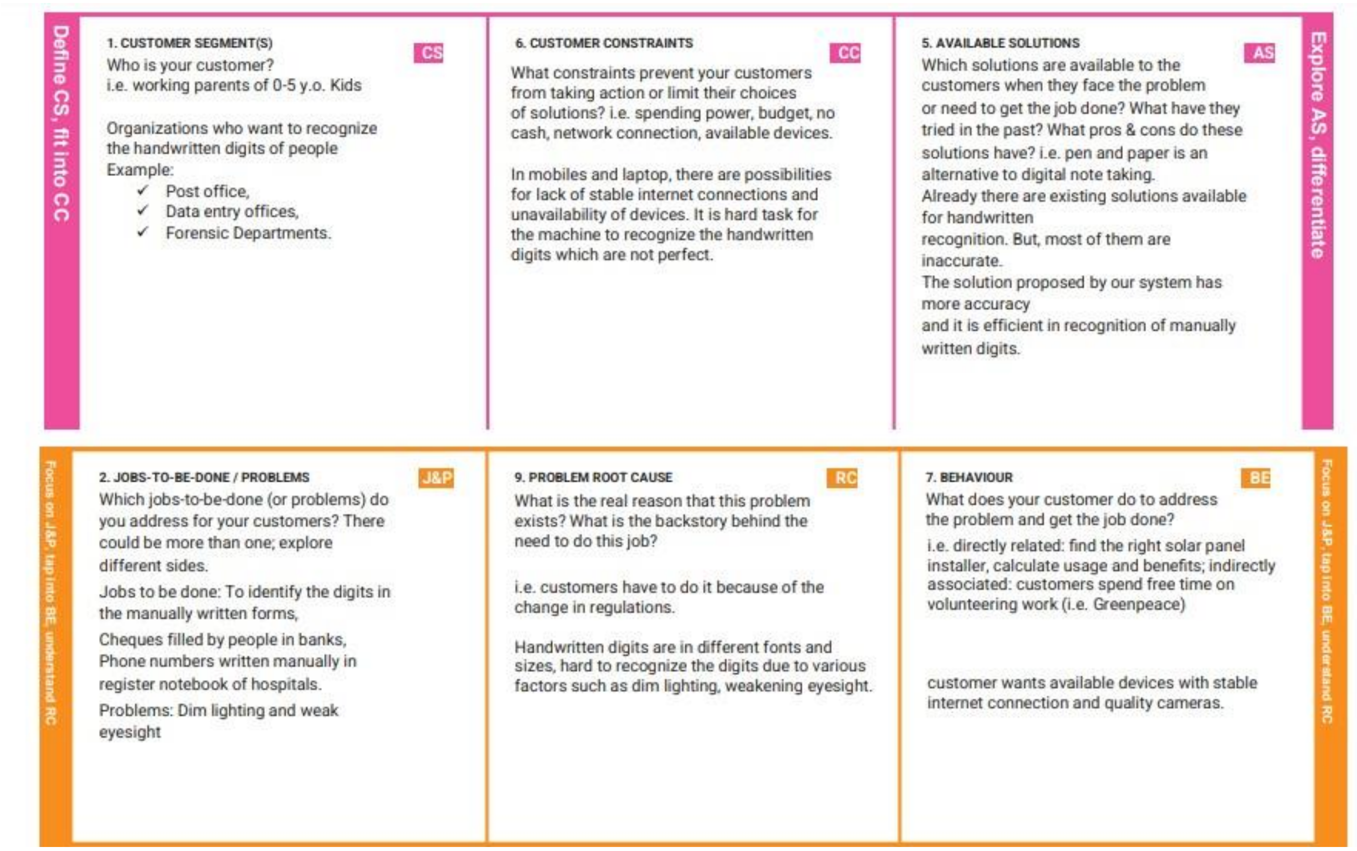
3.3 Proposed Solution

S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	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 user interacts with the UI (User Interface) to upload the image as input. The uploaded image is analyzed by the model which is integrated. Once the model analyses the uploaded image, the prediction is showcased on the UI.
2	Idea / Solution description	Convolutional Neural Networks (CNN) has become one of the most appealing approaches and has been an ultimate factor in a variety of recent success and challenging machine learning applications. In our model we use AlexNet , which is one of the CNN architectures . AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another

		GPU. Not only does this mean that a bigger model can be trained, but it also cuts down on the training time. It also reduces the overfitting problem by Data Augmentation and Dropout.
3	Novelty / Uniqueness	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 is the existing method along with this we add some features to make our project unique among them.
4	Social Impact / Customer Satisfaction	Even the unclear or blurred digits can be recognized after the removal of noise and data preprocessing .One such application is a handwritten digit recognition system that can be used in postal mail sorting, bank check processing, form data entry, etc.,
5	Business Model (Revenue Model)	Handwritten digit recognition is necessary because everything is digitalized. The benefits of handwritten digit recognizer is high. In the banking sector, it is very efficient. It is used to recognize the figures written on cheques.So, Varied handwriting of each and every person in the cheque can be identified. Handwritten addresses are difficult to sort by machine, not

		necessarily because of sloppy handwriting, but because people write all over the envelope.We have hard time segmenting handwritten addresses into their components, such as ZIP code or street address, because very few people print addresses neatly in a prescribed format. So, this problem can be solved using Handwritten digit recognition system.
6	Scalability of the Solution	In our model, AlexNet significantly outperformed as it is trained on a GTX 580 GPU with only 3 GB of memory which couldn’t fit the entire network. So the network was split across 2 GPUs, with half of the neurons(feature maps) on each GPU. So, a greater accuracy can be attained by allowing multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU.

3.4 Problem Solution fit


Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. Kids

Organizations who want to recognize the handwritten digits of people
Example:
- ✓ Post office,
- ✓ Data entry offices,
- ✓ Forensic Departments.

6. CUSTOMER CONSTRAINTS CC

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.

In mobiles and laptop, there are possibilities for lack of stable internet connections and unavailability of devices. It is hard task for the machine to recognize the handwritten digits which are not perfect.

5. AVAILABLE SOLUTIONS AS

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 note taking.
Already there are existing solutions available for handwritten recognition. But, most of them are inaccurate.
The solution proposed by our system has more accuracy and it is efficient in recognition of manually written digits.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

Jobs to be done: To identify the digits in the manually written forms,

Cheques filled by people in banks, Phone numbers written manually in register notebook of hospitals.

Problems: Dim lighting and weak eyesight

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists? What is the backstory behind the need to do this job?

i.e. customers have to do it because of the change in regulations.

Handwritten digits are in different fonts and sizes, hard to recognize the digits due to various factors such as dim lighting, weakening eyesight.

7. BEHAVIOUR BE

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)

customer wants available devices with stable internet connection and quality cameras.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

Advertisement in the market about the efficient recognition of digits.

Articles about the achievements made by our project.

4. EMOTIONS: BEFORE / AFTER EM

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.

Defects are common and our project is not an exception

When the system failed to recognize the digit,

Customer Mentality:
Before:(Failure)
We would give guarantee that it would work most of the time
and if any error occurs, they can contact us at any time.
So, customers can feel at ease.
After:(Failure)
They have no need to panic when the failure occurs
They can easily contact us to rectify the error.
We would solve the defect as soon as possible.

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
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 aims to recognize handwritten digits using machine learning techniques thereby saving costs to the organization improving employee productivity.

In our model we use AlexNet , which is one of the CNN architectures . AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU. Not only does this mean that a bigger model can be trained, but it also cuts down on the training time. It also reduces the overfitting problem by Data Augmentation and Dropout.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

Requires Stable internet connection for image processing.

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

Obtain modern electronic devices and check they are working

Identify strong TR & EM

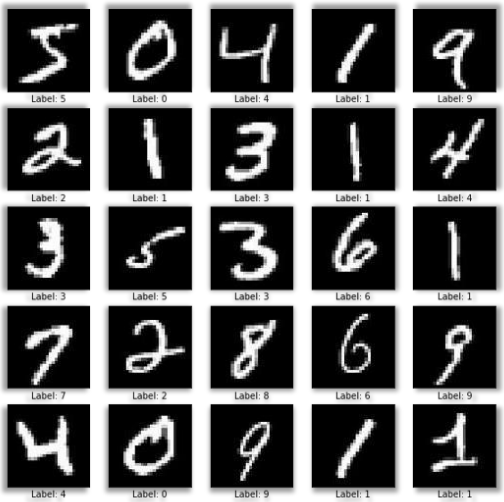
SOLUTION:

MNIST Dataset Description

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. 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 analysed by the model and the detected result is returned on to UI

The MNIST Handwritten Digit Recognition Dataset contains 60,000 training and 10,000 testing labelled handwritten digit pictures.

Each picture is 28 pixels in height and 28 pixels wide, for a total of 784 (28×28) pixels. Each pixel has a single pixel value associated with it. It indicates how bright or dark that pixel is (larger numbers indicates darker pixel). This pixel value is an integer ranging from 0 to 255.



PROCEDURE

- Install the latest TensorFlow library.
- Prepare the dataset for the model.
- Develop Single Layer Perceptron model for classifying the handwritten digits.
- Plot the change in accuracy per epochs.
- Evaluate the model on the testing data.
- Analyse the model summary.
- Add hidden layer to the model to make it Multi-Layer Perceptron.
- Add Dropout to prevent overfitting and check its effect on accuracy.
- Increasing the number of Hidden Layer neuron and check its effect on accuracy.
- Use different optimizers and check its effect on accuracy.
- Increase the hidden layers and check its effect on accuracy.
- Manipulate the batch size and epochs and check its effect on accuracy.

MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consists of 60,000 training images and 10,000 test images. The artificial neural networks can all most mimic the human brain and are a key ingredient in image processing field.

Handwritten digit recognition using MNIST dataset is a major project made with the help of Neural Network. It basically detects the scanned images of handwritten digits.

We have taken this a step further where our handwritten digit recognition system not only detects scanned images of handwritten digits but also allows writing digits on the screen with the help of an integrated GUI for recognition.

Approach:

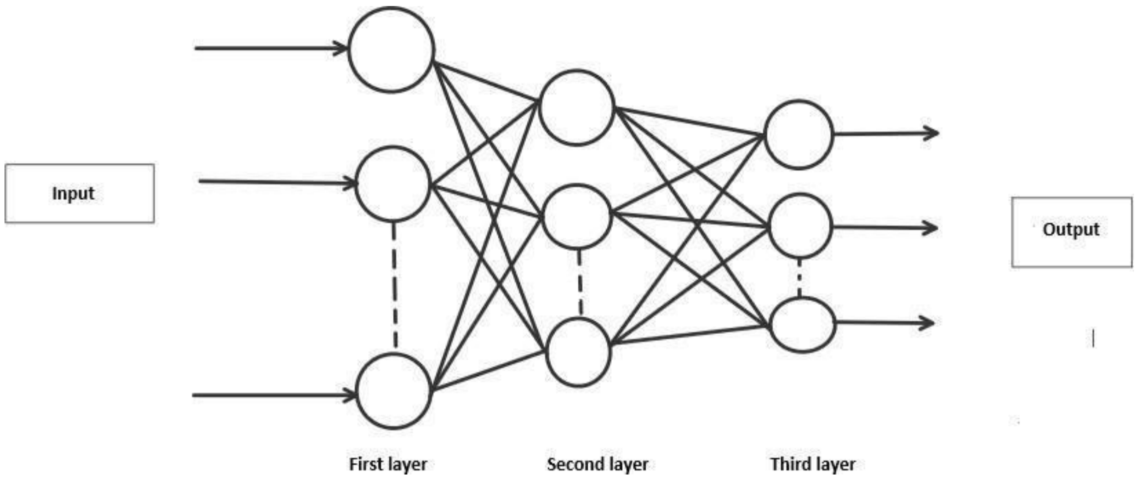
We will approach this project by using a three-layered Neural Network.

- **The input layer:** It distributes the features of our examples to the next layer for calculation of activations of the next layer.
- **The hidden layer:** They are made of hidden units called activations providing nonlinear ties for the network. A number of hidden layers can vary according to our requirements.
- **The output layer:** The nodes here are called output units. It provides us with the final prediction of the Neural Network on the basis of which final predictions can be made.

A neural network is a model inspired by how the brain works. It consists of multiple layers having many activations, this activation resembles neurons of our brain. A neural network tries to learn a set of parameters in a set of data which could help to recognize the underlying relationships. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.

METHODOLOGY:

We have implemented a Neural Network with 1 hidden layer having 100 activation units (excluding bias units). The data is loaded from a .mat file, features(X) and labels(y) were extracted. Then features are divided by 255 to rescale them into a range of [0,1] to avoid overflow during computation. Data is split up into 60,000 training and 10,000 testing examples. Feedforward is performed with the training set for calculating the hypothesis and then backpropagation is done in order to reduce the error between the layers. The regularization parameter lambda is set to 0.1 to address the problem of overfitting. Optimizer is run for 70 iterations to find the best fit model.



ALGORITHM:

Forward Propagation Architecture:

It is a small workflow of how CNN module will extract the features and classify the image based on it. The architecture shows the input layer, hidden layers and output layer of the network. There are many layers involved in the feature extraction phase of the network which involves convolution and subsampling .

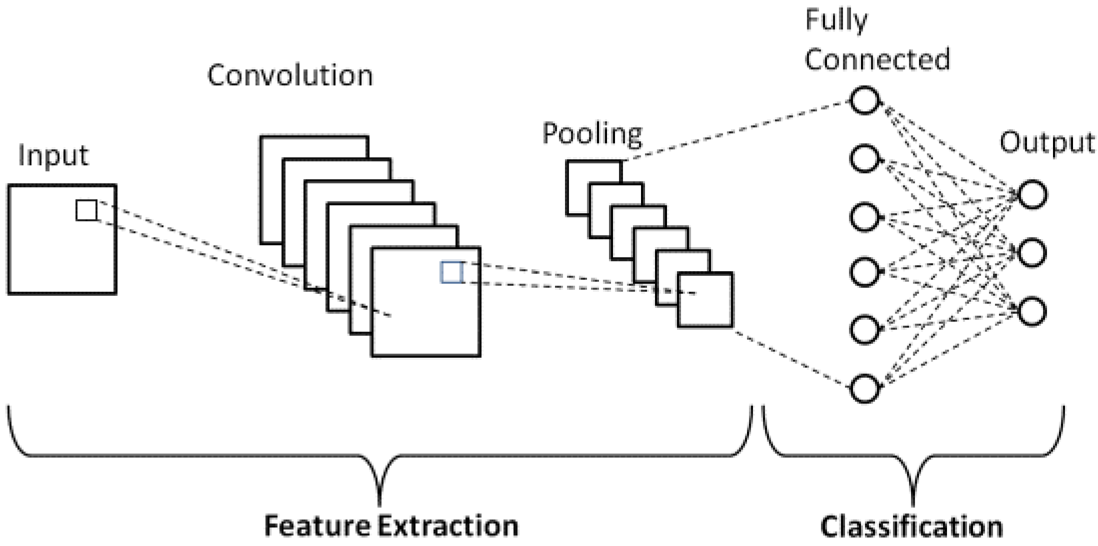
EXPLANATION OF THE PROPOSED SYSTEM

• The first layer of the architecture is the User layer. User layer will comprise of the people who interacts with the app and for the required results.

• The next three layers is the frontend architecture of the application.

The application will be developed using which is the open-source platform for HTML, CSS and JavaScript.

The application is deployed in the localhost which is shown on the browser. Through the app, the user will be able to u1p9load pictures of the handwritten digits and convert it into the digitalized form. • The one in between the database and view layer is the business layer which is the logical calculations on the basis of the request from the client side. It also has the service interface. • The backend layer consists of two datasets: Training Data and Test Data. The MNIST database has been used for that which is already divided into training set of 60,000 examples and test of 10,000 examples. • The training algorithm used is Convolution Neural Network. This will prepare the trained model whichwill be used to classify the digits present in the test data. Thus, we can classify the digits present in the images as: Class 0,1,2,3,4,5,6,7,8,9.



WORKING

• Neural Networks receive an input and transform it through a series of hidden layers.

• Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer.

• Neurons in a single layer function completely independently. • The last fully connected layer is called the "output layer".

Convolution Layer: The Convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume.

During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input and producing a 2- dimensional activation map of that filter.

As a result, the network learns filters that activate when they see some specifictype of feature at some spatial position in the input..

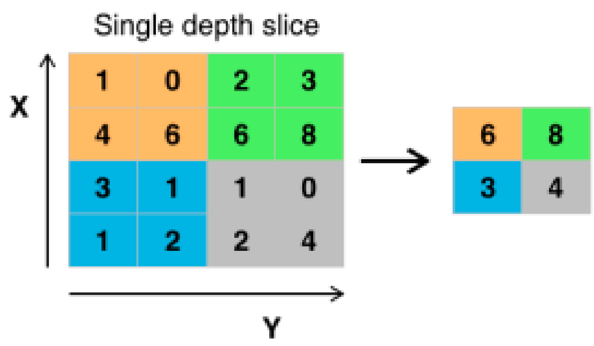
Feature Extraction:

All neurons in a feature share the same weights .In this way all neurons detect the same feature at different positions in the input image. Reduce the number of free parameters.

Subsampling Layer: Subsampling, or down sampling, refers to reducing the overall size of a signal .The subsampling layers reduce the spatial resolution of each feature map. Reduce the effect of noises and shift or distortion invarianceis achieved.

Pooling layer: It is common to periodically insert a Pooling layer in-between successive Conv layer in a Convent architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation.

TensorFlow: TensorFlow is an open-source machine learning library for research and production. TensorFlow offers APIs for beginners and experts to develop for desktop, mobile, web, and cloud. See the sections below to get started. By scanning the numerical digit and convert into png format using python3 command in terminal we can get text output and sound output.



Pooling layer

4. REQUIREMENT ANALYSIS

4.3 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Input Correlation	Digital image correlation is a technique that combines image registration and tracking methods for accurate 2D measurements of changes in images and recognizes the characters from the images.
FR-2	Data Preparation	Data preparation is the process of preparing raw data so that it is suitable for further processing and analysis.
FR-3	Feature Extraction	Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set.
FR-4	Character Classification	In character classification phase, the attributes of the data in the picture are compared to the classes in the database to determine in which class the picture belongs to.

4.4 Non-Functional requirements

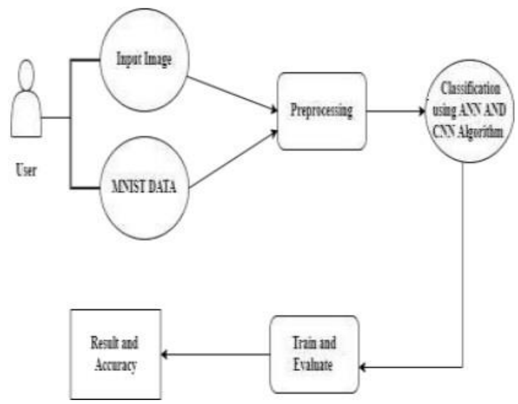
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Handwritten digit recognition is one of the major important issues in pattern recognition applications. Some of the applications for digit recognition include data entry forms, Bank check processing etc,.
NFR-2	**Security**	The applications of handwritten digit recognition can be used in the banking sector where it can be used to maintain the security pin numbers safely. It can be also used for blind-people by using sound output.
NFR-3	**Reliability**	Reliability indicates the probability that the system will perform its intended function for a larger period of sufficient time and also it will operate in a secured environment without any failures.
NFR-4	**Performance**	The standard implementations of neural networks achieve an accuracy of approximately (98–99) percent in correctly classifying the handwritten digits.
NFR-5	**Availability**	The features for handwritten digit recognition have been Acquainted. These features are based on shape analysis of the digit image and extract slant or slope information. They are effective in obtaining good recognition of accuracy.
NFR-6	**Scalability**	The scalability in the task of handwritten digit recognition, using a classifier, has great importance and it makes use of online handwriting recognition on computer tablets, recognizing zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up manually(for example - tax forms) and so on.

5. PROJECT DESIGN

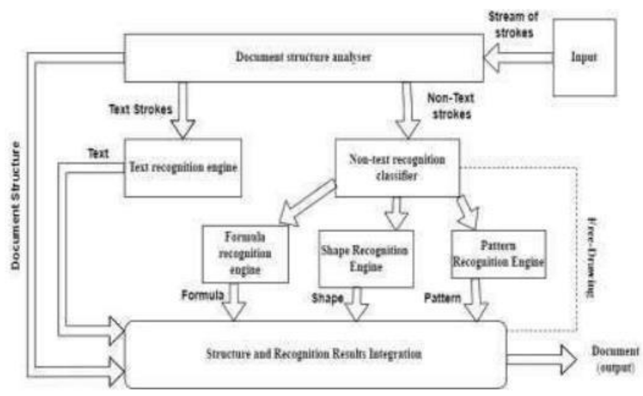
5.1Data Flow Diagrams

Example: (Simplified) FLOW



A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

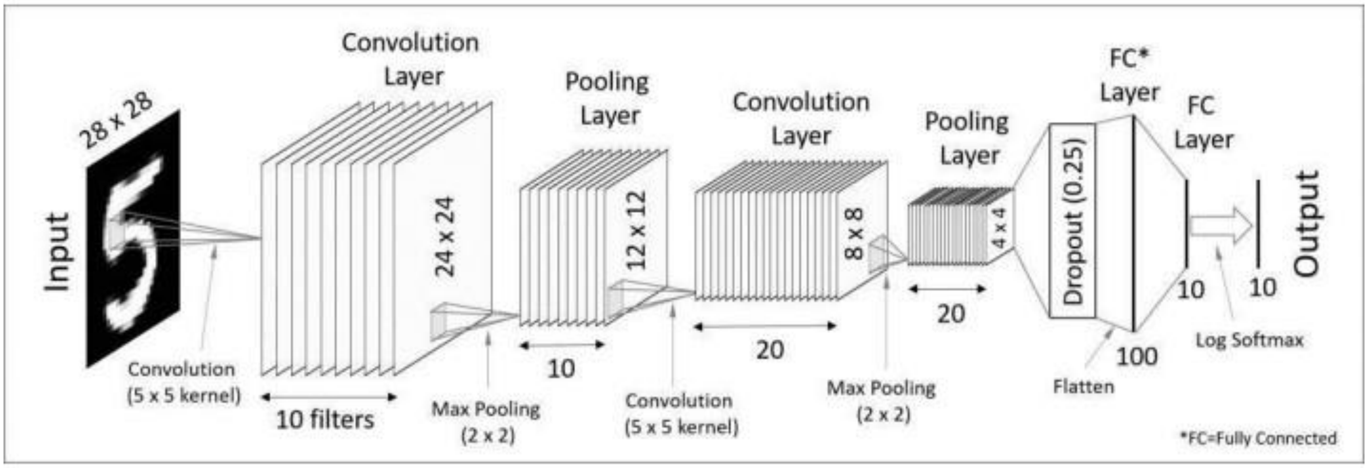
Example: DFD Level 0 (Industry Standard)



5.2 Solution & Technical Architecture

Solution Architecture

Solution Architecture Diagram:



CNN Architecture For Handwritten Digit Recognition

Technology Architecture

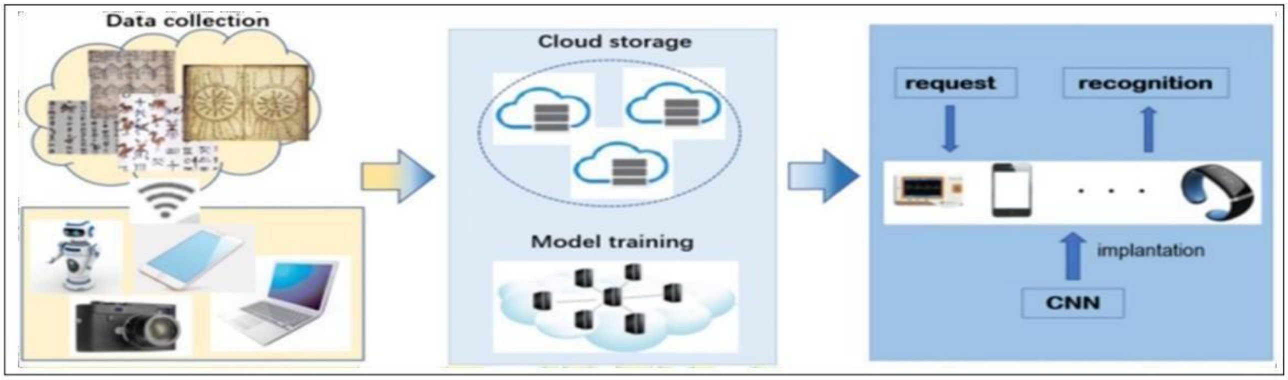


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI	HTML, CSS, JavaScript
2.	Application Logic-1	Model is built	Python
3.	Application Logic-2	Python model is deployed	IBM Watson Studio
4.	File Storage	Predicted outputs of the image are stored in a local folder.	Local Filesystem
5.	Machine Learning Model	To predict the image uploaded by the user.	Image Recognition Model
6.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Flask Cloud Server Configuration : IBM Watson Studio	Local, Cloud Foundry.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Flask
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	High workload can be supported without undergoing any major changes.	Technology used in the architecture is that with Python and the IBM cloud.
4.	Availability	Readily available enables the IT Infrastructure to function when some of the components fail.	Technology used is IBM cloud.
5.	Performance	Performance technology is a field which uses various tools,processes and procedures in a systematic and efficient manner to improve the desired outcomes of individuals and organizations.	Technology used is python.

5.3User Stories

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Home	USN-1	As a user, I can view the guide and awareness to use this application.	I can view the awareness to use this application and its limitations.	Low	Sprint-1
		USN-2	As a user, I'm allowed to view the guided video to use the interface of this application.	I can gain knowledge to use this application by a practical method.	Low	Sprint-1
		USN-3	As a user, I can read the instructions to use this application.	I can read instructions also to use it in a user-friendly method.	Low	Sprint-2
	Recognize	USN-4	As a user, In this prediction page I get to choose the image.	I can choose the image from our local system and predict the output.	High	Sprint-2
	Predict	USN-6	As a user, I'm Allowed to upload and choose the image to be uploaded	I can upload and choose the image from the system storage and also in any virtual storage.	Medium	Sprint-3
		USN-7	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 of the result.	High	Sprint-4
		USN-8	As a user, I can access the MNIST data set	I can access the MNIST data set to produce the accurate result.	Medium	Sprint-3
Customer (Web user)	Home	USN-9	As a user, I can view the guide to use the web app.	I can view the awareness of this application and its limitations.	Low	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Home	USN-1	As a user, I can view the guide and awareness to use this application.	I can view the awareness to use this application and its limitations.	Low	Sprint-1
		USN-2	As a user, I'm allowed to view the guided video to use the interface of this application.	I can gain knowledge to use this application by a practical method.	Low	Sprint-1
		USN-3	As a user, I can read the instructions to use this application.	I can read instructions also to use it in a user-friendly method.	Low	Sprint-2
	Recognize	USN-10	As a user, I can use the web application virtually anywhere.	I can use the application portably anywhere.	High	Sprint-1
		USN-11	As it is an open source, can use it cost freely.	I can use it without any payment to be paid for it to access.	Medium	Sprint-2
		USN-12	As it is a web application, it is installation free	I can use it without the installation of the application or any software.	Medium	Sprint-4
	Predict	USN-13	As a user, I'm Allowed to upload and choose the image to be uploaded	I can upload and choose the image from the system storage and also in any virtual storage.	Medium	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Dashboard	USN-1	As a user, they can see the information regarding the prediction of handwritten digit recognition.	2	High	Pavithrah M, Nandhini S, Lakshmi A, Visaka L
Sprint-1	Launch	USN-2	On clicking the launch button, it will redirect the user to a page where the images to be predicted can be uploaded.	2	High	Pavithrah M, Nandhini S, Lakshmi A, Visaka L
Sprint-2	Upload	USN-3	Users can select the image from the local storage.	2	High	Nandhini S, Visaka L
Sprint-3	Predict	USN-4	Once the image is uploaded, it will predict the respective image.	2	High	Lakshmi A, Pavithrah M
Sprint-4	Display	USN-5	The predicted image will be displayed with the accuracy chart.	2	High	Pavithrah M, Nandhini S, Lakshmi A, Visaka L

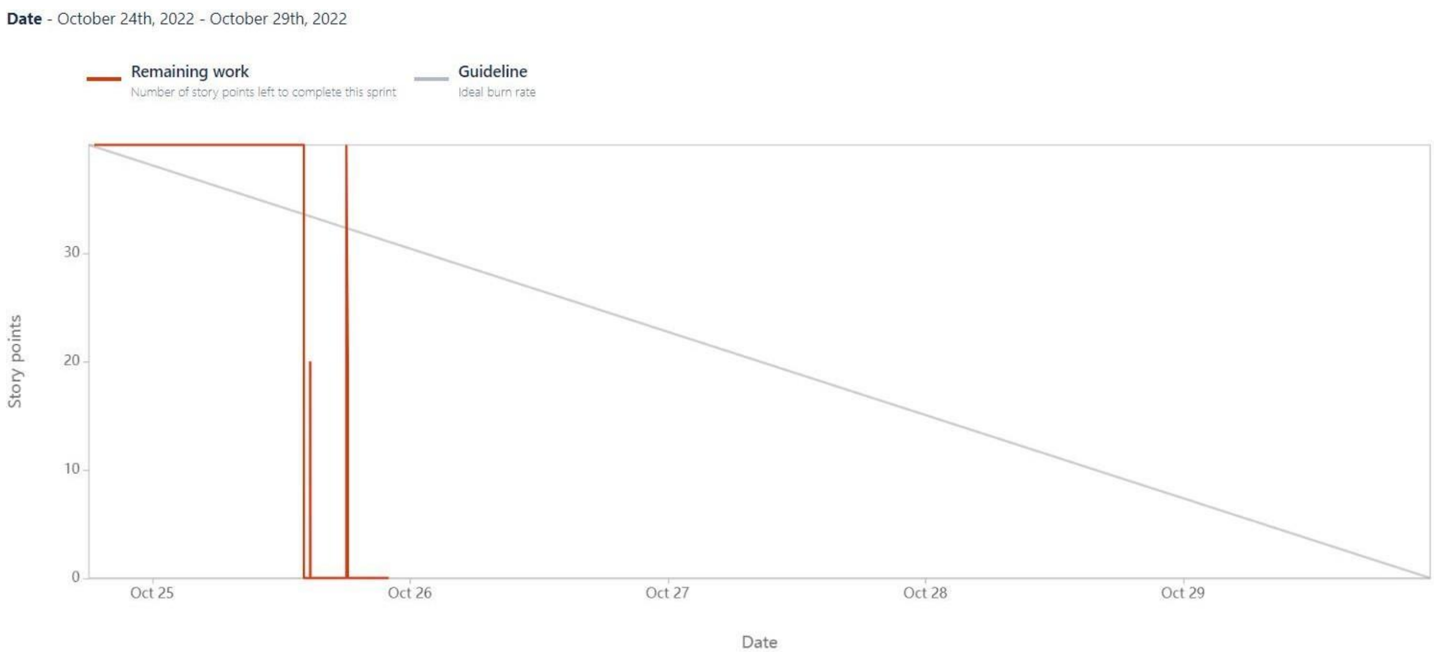
6.2 Sprint Delivery Schedule

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

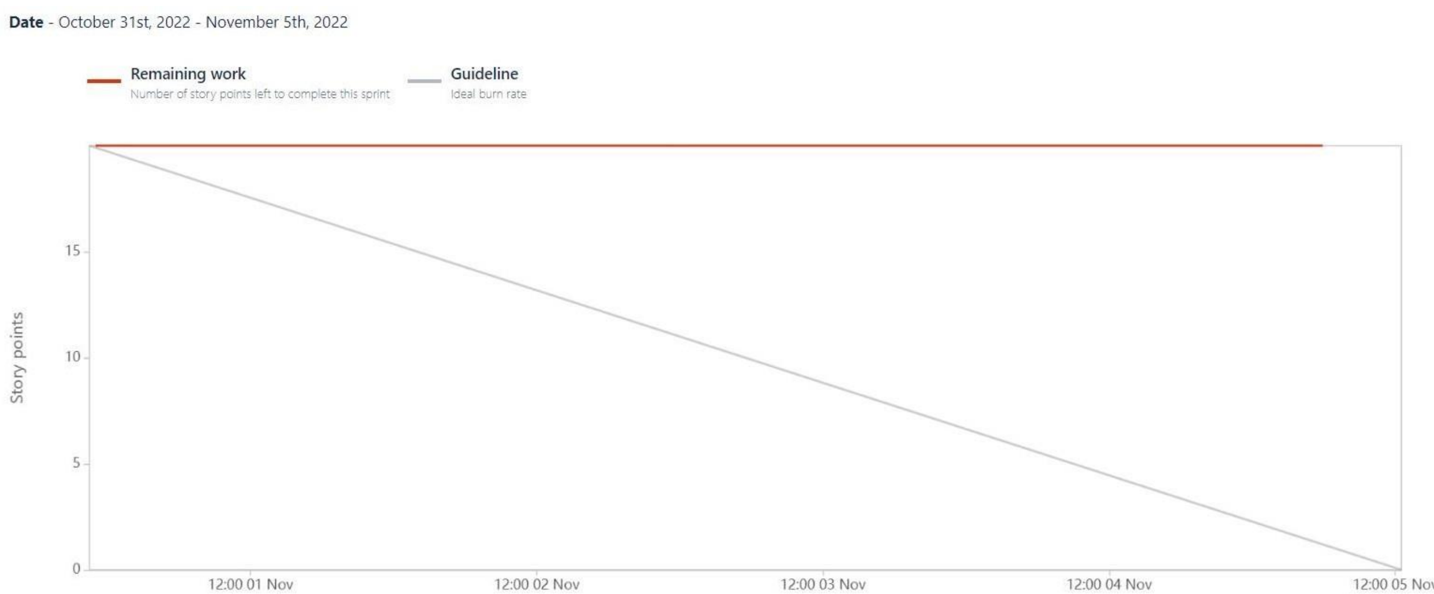
6.3 Reports from JIRA

Velocity Report

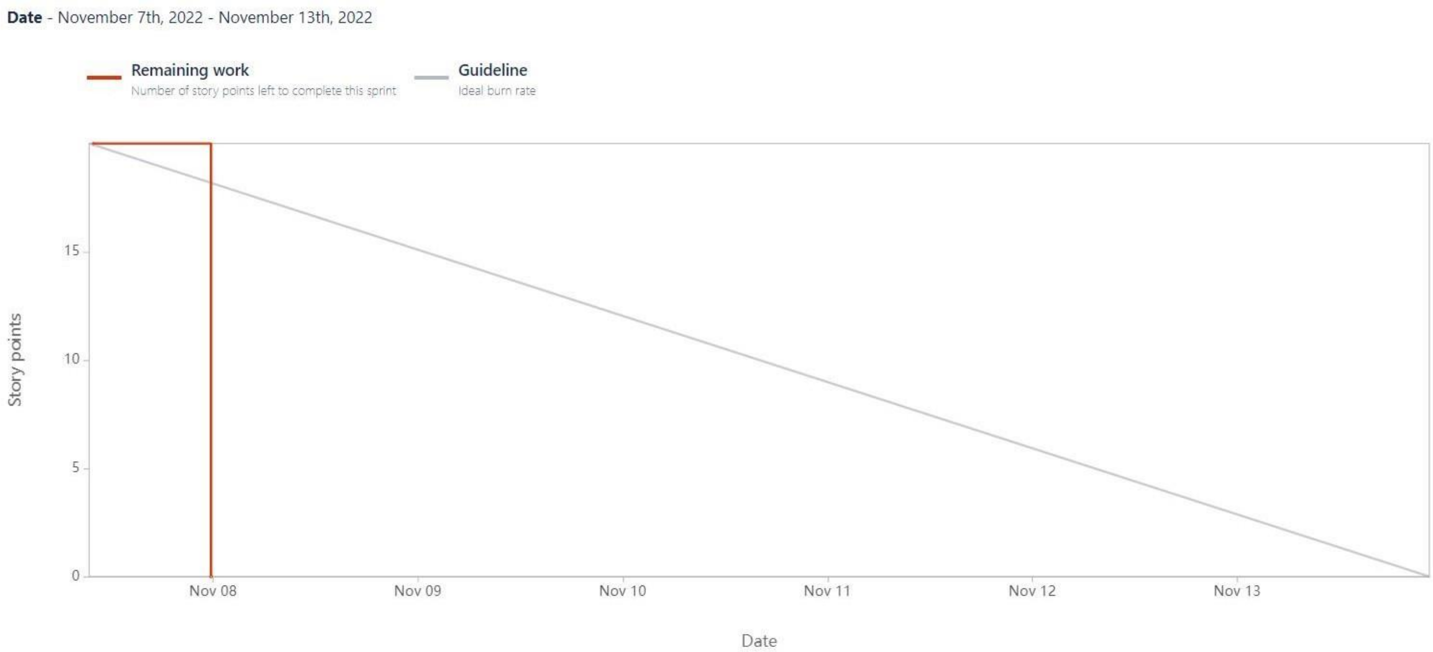
Sprint 1



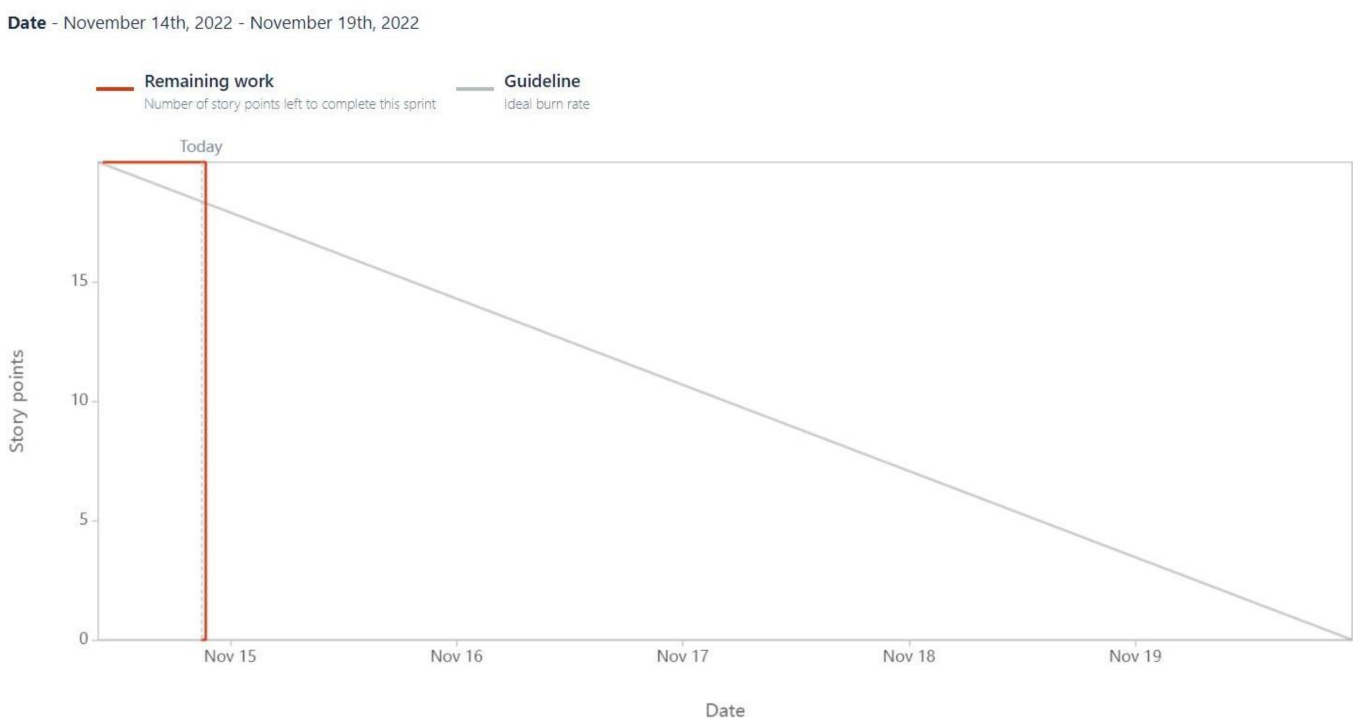
Sprint 2



Sprint 3



Sprint 4



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

```
import torch
import base64
import config
import matplotlib
import numpy as np
from PIL import Image
from io import BytesIO
from train import MnistModel
import matplotlib.pyplot as plt
from flask import Flask, request, render_template, jsonify
matplotlib.use('Agg')


MODEL = None
DEVICE = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')

app = Flask(__name__)


class SaveOutput:
    def __init__(self):
        self.outputs = []

    def __call__(self, module, module_in, module_out):
        self.outputs.append(module_out)

    def clear(self):
        self.outputs = []
```

```python
    img = img.convert('L')
    img = img.resize((28, 28))
    # img.save(r'templates\image.png')
    img = np.array(img)
    img = img.reshape((1, 28, 28))
    img = torch.tensor(img, dtype=torch.float).unsqueeze(0)

    data, probencoded, interpretencoded = mnist_prediction(img)

    response = {
        'data': str(data),
        'probencoded': str(probencoded),
        'interpretencoded': str(interpretencoded),
    }
    return jsonify(response)


@app.route("/", methods=["GET", "POST"])
def start():
    return render_template("default.html")


if __name__ == "__main__":
    MODEL = MnistModel(classes=10)
    MODEL.load_state_dict(torch.load(
        'checkpoint/mnist.pt', map_location=DEVICE))
    MODEL.to(DEVICE)
    MODEL.eval()
    app.run(host=config.HOST, port=config.PORT, debug=config.DEBUG_MODE)
```

```python
def register_hook():
    save_output = SaveOutput()
    hook_handles = []

    for layer in MODEL.modules():
        if isinstance(layer, torch.nn.modules.conv.Conv2d):
            handle = layer.register_forward_hook(save_output)
            hook_handles.append(handle)
    return save_output


def module_output_to_numpy(tensor):
    return tensor.detach().to('cpu').numpy()


def autolabel(rects, ax):
    """Attach a text label above each bar in *rects*, displaying its height."""
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{0:.2f}'.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')


def prob_img(probs):
    fig, ax = plt.subplots()
    rects = ax.bar(range(len(probs)), probs)
```

8. TESTING

8.1 Test Cases

Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
Homepage_TC_OO1	Functional	Home Page	Verify user is able to see the Homepage when clicked on the link	Home Page should be displayed.	Working as expected	Pass
Homepage_TC_OO2	UI	Home Page	Verify the UI elements in Homepage	Application should show below UI elements: a.choose file button b.predict button c.clear button	Working as expected	Pass
Homepage_TC_OO3	Functional	Home Page	Verify user is able to choose file from the local system and click on predict	Choose file popup screen must be displayed and user should be able to click on predict button	Working as expected	Pass
Homepage_TC_OO4	Functional	Home page	Verify user able to selectinvalid file format	Application won't allow to attach formats other than ".png, .jiff, .pjp, .jpeg, .jpg, .pjpeg"	Working as expected	Pass
Predict_TC_OO5	Functional	Predict page	Verify user is able to navigate to thepredict to and view the predicted result	User must be navigated to the predict page and must view the predictedresult	Working as expected	Pass

8.2 User Acceptance Testing

Defect Analysis

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

Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Client Application	5	0	0	5
Security	5	0	0	5
Final Report Output	5	0	0	5
Performance	5	0	0	5

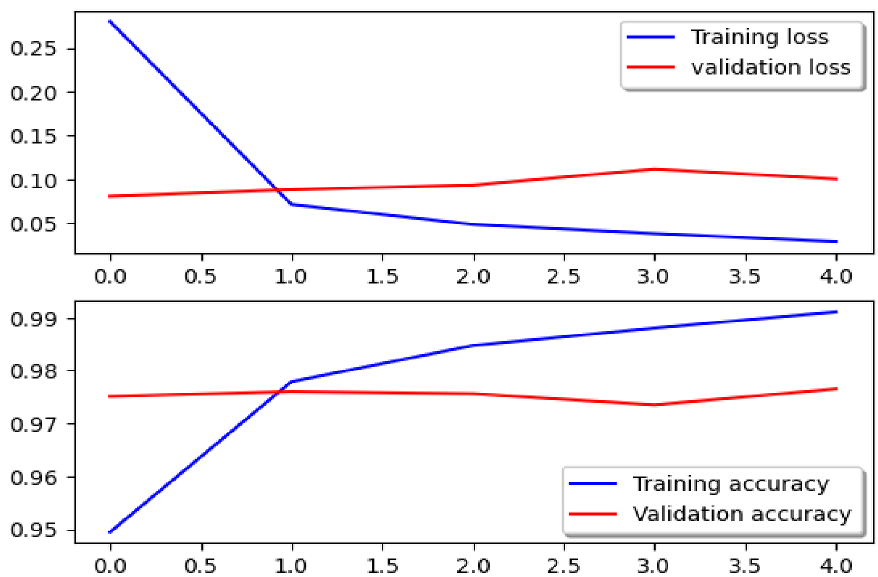
9. RESULTS

9.1 Performance Metrics

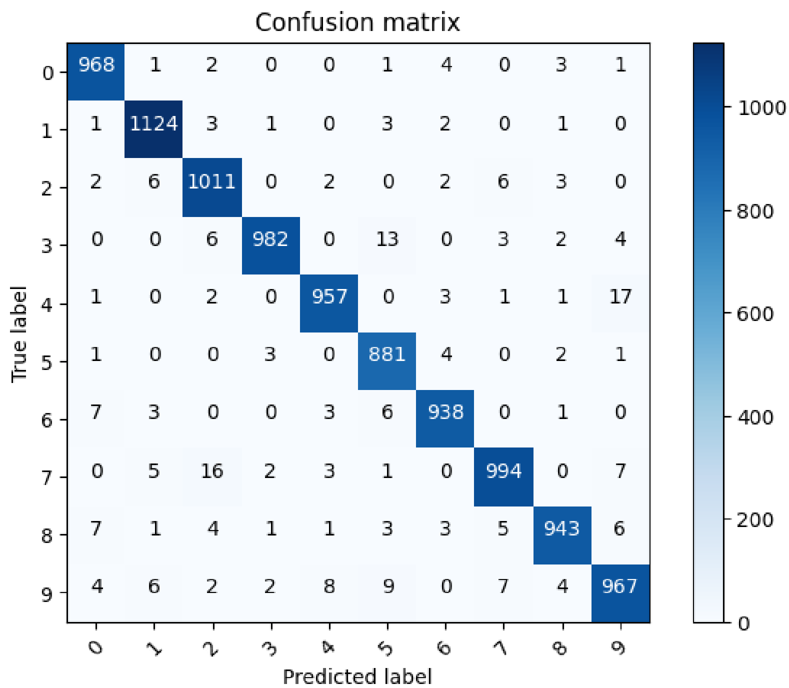
Model Summary:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 64)        640

 conv2d_1 (Conv2D)           (None, 24, 24, 32)        18464

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 10)                184330

=================================================================
Total params: 203,434
Trainable params: 203,434
Non-trainable params: 0
_________________________________________________________________
None
```

Accuracy:



Confusion Matrix:



Classification Report:

	precision	recall	f1-score	support
0	0.98	0.99	0.98	980
1	0.98	0.99	0.99	1135
2	0.97	0.98	0.97	1032
3	0.99	0.97	0.98	1010
4	0.98	0.97	0.98	982
5	0.96	0.99	0.97	892
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

Performance Metrics Result:

Locust Test Report

During: 11/15/2022, 10:52:19 PM - 11/15/2022, 10:56:36 PM

Target Host: http://127.0.0.1:5000/

Script: locustfile.py

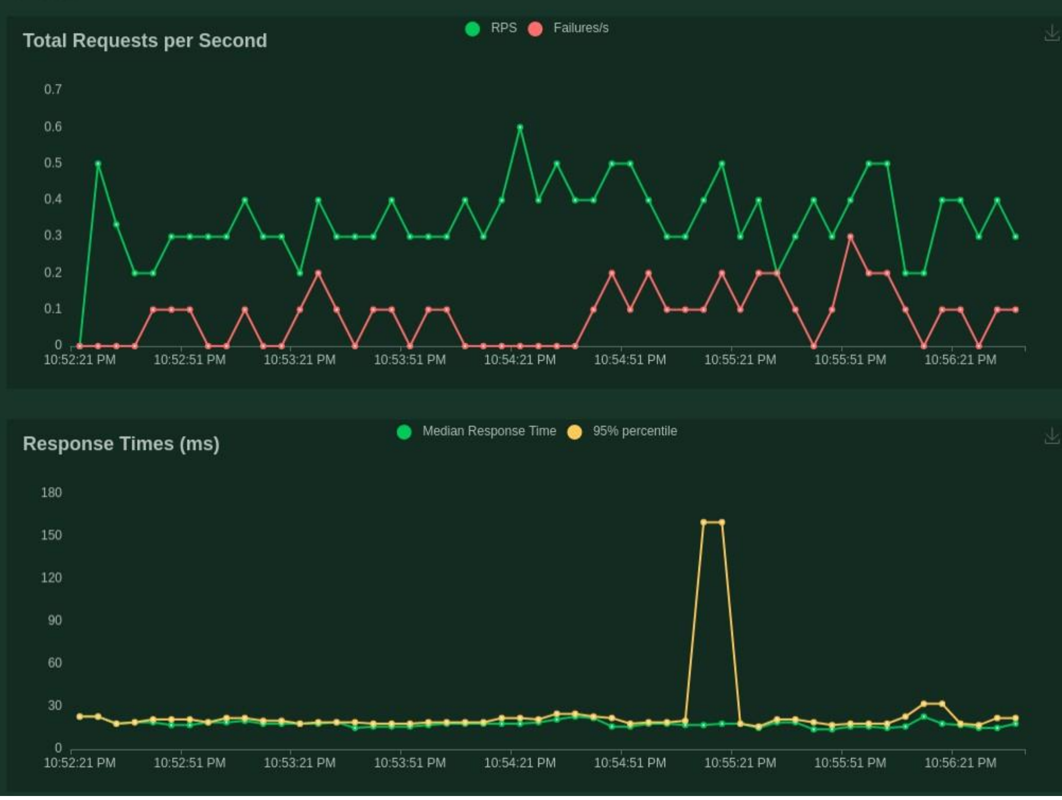
Request Statistics

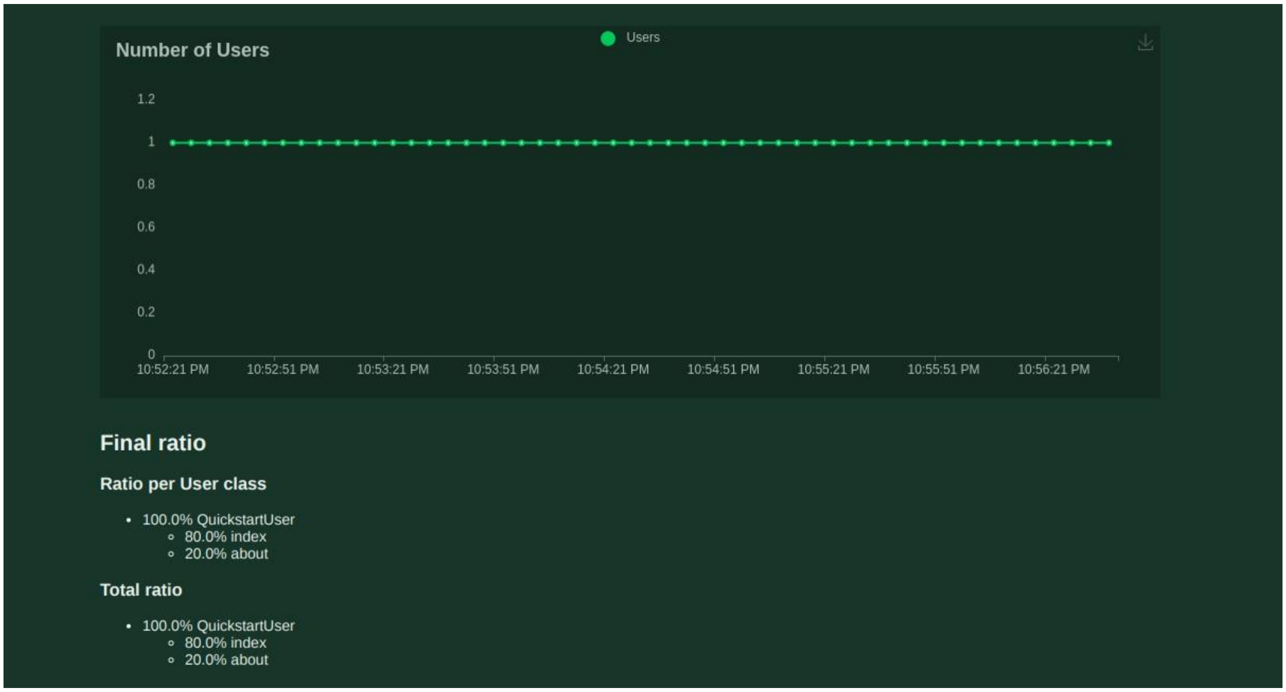
Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	67	0	17	12	24	5875	0.3	0.0
GET	//predict	23	23	21	11	163	265	0.1	0.1
	Aggregated	**90**	**23**	**18**	**11**	**163**	**4441**	**0.4**	**0.1**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	//	18	18	19	19	22	23	25	25
GET	//predict	15	15	16	16	17	32	160	160
	Aggregated	**17**	**18**	**18**	**19**	**22**	**23**	**160**	**160**

Charts





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 high performance server for faster predictions.
- ✓ Prone to occasional errors.

11. CONCLUSION

This project demonstrated a web application that uses machine learning to recognie 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.

13. APPENDIX

Source Code

HTML AND CSS:

index.html:

```
            if (dot_flag) {
                ctx.beginPath();
                ctx.fillStyle = x;
                ctx.fillRect(currX, currY, 2, 2);
                ctx.closePath();
                dot_flag = false;
            }
        }
        if (res == 'up' || res == "out") {
            flag = false;
        }
        if (res == 'move') {
            if (flag) {
                prevX = currX;
                prevY = currY;
                currX = e.clientX - canvas.offsetLeft;
                currY = e.clientY - canvas.offsetTop;
                draw();
            }
        }
    }
</script>

<body onload="init()"style="background-color:#E1F70C  ;">

    <center>
        <h1> Artificial Intelligence Project- Handwritten Digit Recognition using <span id="text">PyTorch CNN</span></h1>
<h2 style="color:red;">Team id:PNT2022TMID22980</h2>
    </center>
    <div id="side">
        <h4 style="color:black;" id='text'> Draw a Digit in the center of the Box.. </h4>
        <canvas id="can" width="200px" height="200px"></canvas>
        <img id="canvasimg">
        <div style="margin-top: 10;">
            <button class="ripple" id="btn" onclick="save()"> predict </button>
             

            <button id="clr" onclick="erase()" > clear </button>
            <h3 id="prediction"></h3>
        </div>
    </div>
    <div>
        <img id="probs" src="" alt="" height="45%" width="35%">
        <img id="interpret" src="" alt="" height="45%" width="35%">
    </div>

</body>
```

```
function erase() {
    ctx.clearRect(0, 0, w, h);
    document.getElementById("canvasimg").style.display = "none";
    document.getElementById("prediction").style.display = "none";
    document.getElementById("probs").style.display = "none";
    document.getElementById("interpret").style.display = "none";
    b = document.getElementsByTagName("body")[0];
    b.querySelectorAll('a').forEach(n => n.remove());
}

function save() {
    document.getElementById("prediction").style.display = "block";
    document.getElementById("probs").style.display = "block";
    document.getElementById("interpret").style.display = "block";
    var final_image = canvas.toDataURL();
    var a = document.createElement('a');
    a.href = final_image;
    a.download = 'process.png';
    document.body.appendChild(a);
    // a.click();
    $.ajax({
        url: "{{ url_for('process') }}",
        type: 'POST',
        data: final_image,
        success: function (response) {
            endresult = JSON.parse(JSON.stringify(response))
            console.log(endresult)
            $('#prediction').html('Prediction is: <span id="text">' + endresult.data + '</span>')
            $('#probs').prop('src', 'data:image/png;base64,' + endresult.probencoded)
            $('#interpret').prop('src', 'data:image/png;base64,' + endresult.interpretencoded)
        }
    });
}

function findxy(res, e) {
    if (res == 'down') {
        prevX = currX;
        prevY = currY;
        currX = e.clientX - canvas.offsetLeft;
        currY = e.clientY - canvas.offsetTop;

        flag = true;
        dot_flag = true;
        if (dot_flag) {
            ctx.beginPath();
            ctx.fillStyle = x;
```

```
<html>
<script type="text/javascript" src="{{url_for('static', filename='jquery.min.js') }}"></script>
<link rel="stylesheet" type="text/css" href="{{url_for('static', filename='style.css') }}">
<script type="text/javascript">
    var canvas, ctx, flag = false,
        prevX = 0,
        currX = 0,
        prevY = 0,
        currY = 0,
        dot_flag = false;

    var x = "red",
        y = 8;

    function init() {
        canvas = document.getElementById('can');
        document.getElementById("probs").style.display = "none";
        document.getElementById("interpret").style.display = "none";
        ctx = canvas.getContext("2d");
        w = canvas.width;
        h = canvas.height;

        canvas.addEventListener("mousemove", function (e) {
            findxy('move', e)
        }, false);
        canvas.addEventListener("mousedown", function (e) {
            findxy('down', e)
        }, false);
        canvas.addEventListener("mouseup", function (e) {
            findxy('up', e)
        }, false);
        canvas.addEventListener("mouseout", function (e) {
            findxy('out', e)
        }, false);
    }


    function draw() {
        ctx.beginPath();
        ctx.moveTo(prevX, prevY);
        ctx.lineTo(currX, currY);
        ctx.strokeStyle = x;
        ctx.lineWidth = y;
        ctx.stroke();
        ctx.closePath();
    }

    function erase() {
```

Styles.css

```css
/* Ripple effect */
.ripple {
    background-position: center;
    transition: background 0.8s;
  }
  /* .ripple:hover {

  } */
  .ripple:active {
    background: #25282b radial-gradient(circle, transparent 1%, #47a7f5 1%) center/15000%;
    /* background-color: #6eb9f7; */
    background-size: 100%;
    transition: background 0s;
  }

  /* Button style */
  button {
    border: none;
    border-radius: 2px;
    padding: 12px 18px;
    font-size: 16px;
    text-transform: uppercase;
    cursor: pointer;
    color: white;
    background-color: #2196f3;
    box-shadow: 0 0 4px #999;
    outline: none;
  }

  /* Span style */
#text {
    color: #4DAF74;
}

/* Canvas style */
#can {
    margin-top: 10;
    border:5px solid;
}

/* canvasimage style */
#canvasimg{
    position:absolute;
    display:none;
}

body{
    font-family: sans-serif;
}

#side{
  float: left;
  padding-left:15%;
}
```

App.py

```python
import torch
from torch import nn, optim
from torch.utils import data

from utils import *
import pandas as pd
import numpy as np
from os import makedirs
from typing import Union
import matplotlib.pyplot as plt
from dataclasses import dataclass

import warnings
warnings.filterwarnings('ignore')


class MnistModel(nn.Module):
    """
    Custom CNN Model for Mnist
    """

    def __init__(self, classes: int) -> None:
        super(MnistModel, self).__init__()

        self.classes = classes

        # initialize the layers in the first (CONV => RELU) * 2 => POOL + DROP
        # (N,1,28,28) -> (N,16,24,24)
        self.conv1A = nn.Conv2d(
            in_channels=1, out_channels=16, kernel_size=5, stride=1, padding=0)
        # (N,16,24,24) -> (N,32,20,20)
        self.conv1B = nn.Conv2d(
            in_channels=16, out_channels=32, kernel_size=5, stride=1, padding=0)
        # (N,32,20,20) -> (N,32,10,10)
        self.pool1 = nn.MaxPool2d(kernel_size=2)
        self.act = nn.ReLU()
        self.do = nn.Dropout(0.25)

        # initialize the layers in the second (CONV => RELU) * 2 => POOL + DROP
        # (N,32,10,10) -> (N,64,8,8)
        self.conv2A = nn.Conv2d(
            in_channels=32, out_channels=64, kernel_size=3, stride=1, padding=0)
        # (N,64,8,8) -> (N,128,6,6)
```

```python
def test_loop_fn(test, model, device):
    """
    Testing Loop

    Args:
        test: Test DataFrame
        model: NN Model
        device: Device (CPU/CUDA)

    Returns:
        List of Predicted Labels
    """
    model.eval()
    # convert test data to FloatTensor
    test = torch.as_tensor(test)
    test = test.to(device, dtype=torch.float)

    # Get predictions
    pred = model(test)
    # Get predictions from the maximum value
    _, predlabel = torch.max(pred.data, 1)
    # converting to list
    predlabel = predlabel.tolist()

    # Plotting the predicted results
    L = 5
    W = 5
    _, axes = plt.subplots(L, W, figsize=(12, 12))
    axes = axes.ravel()

    for i in np.arange(0, L * W):
        axes[i].imshow(test[i].cpu().detach().numpy().reshape(28, 28))
        axes[i].set_title("Prediction Class = {:0.1f}".format(predlabel[i]))
        axes[i].axis('off')

    plt.suptitle('Predictions on Test Data')
    plt.subplots_adjust(wspace=0.5)
    plt.show()

    return predlabel
```

```python
def eval_loop_fn(data_loader, model, device):
    """
    Evaluation Loop

    Args:
        data_loader: Evaluation Data Loader
        model: NN Model
        device: Device (CPU/CUDA)

    Returns:
        List of Target Labels and True Labels
    """
    # full list of targets, outputs
    fin_targets = []
    fin_outputs = []
    # set model to eveluate
    model.eval()  # as model is set to eval, there will be no optimizer and scheduler update

    # iterate over data loader
    for _, (ids, targets) in enumerate(data_loader):
        ids = ids.to(device, dtype=torch.float)
        targets = targets.to(device, dtype=torch.long)

        outputs = model(x=ids)
        loss = loss_fn(outputs, targets)
        loss.backward()

        # Get predictions from the maximum value
        _, outputs = torch.max(outputs.data, 1)

        # appending the values to final lists
        fin_targets.append(targets.cpu().detach().numpy())
        fin_outputs.append(outputs.cpu().detach().numpy())
    return np.vstack(fin_outputs), np.vstack(fin_targets)


def test_loop_fn(test, model, device):
    """
    Testing Loop

    Args:
```

```python
def train_loop_fn(data_loader, model, optimizer, device, scheduler=None):
    """
    Training Loop

    Args:
        data_loader: Train Data Loader
        model: NN Model
        optimizer: Optimizer
        device: Device (CPU/CUDA)
        scheduler: Scheduler. Defaults to None.
    """
    # set model to train
    model.train()
    # iterate over data loader
    train_loss = []
    for ids, targets in data_loader:
        # sending to device (cpu/gpu)
        ids = ids.to(device, dtype=torch.float)
        targets = targets.to(device, dtype=torch.long)

        # Clear gradients w.r.t. parameters
        optimizer.zero_grad()
        # Forward pass to get output/logits
        outputs = model(x=ids)
        # Calculate Loss: softmax --> negative log likelihood loss
        loss = loss_fn(outputs, targets)
        train_loss.append(loss)
        # Getting gradients w.r.t. parameters
        loss.backward()
        optimizer.step()

        if scheduler is not None:
            # Updating scheduler
            if type(scheduler).__name__ == 'ReduceLROnPlateau':
                scheduler.step(loss)
            else:
                scheduler.step()
    print(f"Loss on Train Data : {sum(train_loss)/len(train_loss)}")
```

```python
    def forward(self, x: torch.Tensor) -> torch.Tensor:

        # build the first (CONV => RELU) * 2 => POOL layer set
        x = self.conv1A(x)
        x = self.act(x)
        x = self.conv1B(x)
        x = self.act(x)
        x = self.pool1(x)
        x = self.do(x)

        # build the second (CONV => RELU) * 2 => POOL layer set
        x = self.conv2A(x)
        x = self.act(x)
        x = self.conv2B(x)
        x = self.act(x)
        x = self.pool2(x)
        x = self.do(x)

        # build our FC layer set
        x = x.view(x.size(0), -1)
        x = self.dense3(x)
        x = self.act(x)
        x = self.do(x)

        # build the softmax classifier
        x = nn.functional.log_softmax(self.dense4(x), dim=1)

        return x


class MnistDataset(data.Dataset):
    """
    Custom Dataset for Mnist
    """

    def __init__(self, df: pd.DataFrame, target: np.array, test: bool = False) -> None:
        self.df = df
        self.test = test

        # if test=True skip this step
        if not self.test:
```

GitHub & Project Demo Link

GitHub Link

https://github.com/IBM-EPBL/IBM-Project-24536-1659944244

Demo Video

https://github.com/IBM-EPBL/IBM-Project-24536-1659944244/tree/main/Final%20Deliverables/demo_video