**PROJECT REPORT**

**A Novel Method for Handwritten Digit Recognition System**

Submitted By

 Team ID-PNT2022TMID39952

Ashwini V-511319205002

Hema Priya N-511319205007

Priyanga R-511319205019

Sindhujaa S-510519205028

**1. INTRODUCTION**

1.1 Project Overview

1.2 Purpose

**2. LITERATURE SURVEY**

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

**3. IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

**4. REQUIREMENT ANALYSIS**

4.1 Functional requirement

4.2 Non-Functional requirements

**5. PROJECT DESIGN**

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

**6. PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

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

7.1 Feature

1 7.2 Feature

2 7.3 Database Schema (if Applicable)

**8. TESTING**

8.1 Test Cases

8.2 User Acceptance Testing

**9. RESULTS**

9.1 Performance Metrics

**10. ADVANTAGES & DISADVANTAGES**

**11. CONCLUSION**

**12. FUTURE SCOPE**

**13. APPENDIX**

Source Code

Git Hub & Project Demo Link

**1.** **INTRODUCTION:**

Character recognition is a fundamental, but most challenging in the field of pattern recognition with 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 interactions between computers and humans. More precisely Character recognition is the process of detecting and recognizing characters from the input image and converts it into ASCII or other equivalent machine editable form [1][2].

* 1. **PROJECT OVERVIEW:**

The technique by which a computer system can recognize characters and other symbols written by hand in natural handwriting is called handwriting recognition system. Handwriting recognition is classified into offline system. Handwriting recognition is classified into offline [3]. If handwriting is scanned and then understood by the computer, it is called offline handwriting recognition. In case, the handwriting is recognized while writing through touch pad using stylus pen, it is called online handwriting recognition. From the classifier perspective, character recognition systems are classified into two main categories (analytic). The segmentation free also known as the holistic approach to recognize the character without segmenting it into subunits or characters. Each word is represented as a set of global features, e.g. ascender, loops, cusp, etc.

* 1. **PURPOSE:**

Handwritten character processing systems are domain and application specific, like it is not possible to design a generic system which can process all kinds of handwritten scripts and language. Lots of work has been done on European languages and Arabic (Urdu) language. Whereas domestic languages like Hindi, Punjabi, Bangla, Tamil, Gujarati etc.

1. **LITERATURE SURVEY**

An early notable attempt in the area of character recognition research is by Grimsdale in 1959. The origin of a great deal of research work in the early sixties was based on an approach known as analysis-by-synthesis method suggested by Eden in 1968. The great importance of Eden's work was that he formally proved that all handwritten characters are formed by a finite number of schematic features, a point that was implicitly included in previous works. This notion was later used in all methods in syntactic (structural) approaches of character recognition. K. Gaurav, Bhatia P. K. [5] Et al, this paper deals with the various pre-processing techniques involved in the character recognition with different kind of images ranges from a simple handwritten form based documents and documents containing colored and complex background and varied intensities. In this, different preprocessing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed.

* 1. **EXISTING PROBLEM:**

Pre-processing is the basic phase of character recognition and it's crucial for good recognition rate. The main objective of pre-processing steps is to normalize strokes and remove variations that would otherwise complicate recognition and reduce the recognition rate. These variations or distortions include the irregular size of text, missing points during pen movement collections, jitter present in text, left or right bend in handwriting and uneven distances of points from neighbouring positions.

* 1. **REFERENCES:**

[1] R. Jayadevan, S. R. Kolhe, P. M. Patil, and U. Pal, “Automatic processing of handwritten bank

cheque images: a survey,” *Int. J. Doc. Anal. Recognit. IJDAR*, vol. 15, no. 4, pp. 267–296, 2012.

[2] N. M. Nasrabadi, “Pattern recognition and machine learning,” *J. Electron. Imaging*, vol. 16, no.

4, p. 049901, 2007.

[3] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, p. 436, 2015.

[4] Y. Bengio, “Learning deep architectures for AI,” *Found. Trends® Mach. Learn.*, vol. 2, no. 1,

pp. 1–127, 2009.

[5] R. Plamondon and S. N. Srihari, “Online and off-line handwriting recognition: a comprehensive

survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, 2000.

[6] R. M. Bozinovic and S. N. Srihari, “Off-line cursive script word recognition,” *IEEE Trans.*

*Pattern Anal. Mach. Intell.*, vol. 11, no. 1, pp. 68–83, 1989.

[7] K. Marukawa, M. Koga, Y. Shima, and H. Fujisawa, “A High Speed Word Matching Algorithm

for Handwritten Chinese Character Recognition.,” in *MVA*, 1990, pp. 445–450.

[8] R. W. Smith, J. F. McNamara, and D. S. Bradburn, “Pattern Classification Techniques Applied

to Character Segmentation,” *TRW Financ. Syst. Berkeley CA*, 1998.

[9] F. Lauer, C. Y. Suen, and G. Bloch, “A trainable feature extractor for handwritten digit

recognition,” *Pattern Recognit.*, vol. 40, no. 6, pp. 1816–1824, 2007.

[10] L. Bottou *et al.*, “Comparison of classifier methods: a case study in handwritten digit

recognition,” in *Pattern Recognition, 1994. Vol. 2-Conference B: Computer Vision & Image*

*Processing., Proceedings of the 12th IAPR International. Conference on*, 1994, vol. 2, pp. 77–

82.

[11] G. S. Lopes, D. C. da Silva, A. W. O. Rodrigues, and P. P. Reboucas Filho, “Recognition of

handwritten digits using the signature features and Optimum-Path Forest Classifier,” *IEEE Lat.*

*Am. Trans.*, vol. 14, no. 5, pp. 2455–2460, 2016.

[12] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, “A methodology for feature selection

using multiobjective genetic algorithms for handwritten digit string recognition,” *Int. J. Pattern*

*Recognit. Artif. Intell.*, vol. 17, no. 06, pp. 903–929, 2003.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document

recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[14] N. Arica and F. T. Yarman-Vural, “An overview of character recognition focused on off-line

handwriting,” *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 31, no. 2, pp. 216–233,

2001.

[15] H. Murase, M. Shinya, T. Wakahara, and K. Odaka, “Segmentation and recognition of handwritten

character string using linguistic information,” *Trans Inst. Electron. Inf. Commun. Eng.*,

vol. 69, no. 9, pp. 1292–1301, 1986.

[16] D. Lu and Q. Weng, “A survey of image classification methods and techniques for improving

classification performance,” *Int. J. Remote Sens.*, vol. 28, no. 5, pp. 823–870, 2007.

[17] U. Pal, A. Belaıd, and C. Choisy, “Touching numeral segmentation using water reservoir

concept,” *Pattern Recognit. Lett.*, vol. 24, no. 1–3, pp. 261–272, 2003.

[18] “Machine learning,” *Wikipedia*. 14-Jan-2019.

[19] A. Talwar and Y. Kumar, “Machine Learning: An artificial intelligence methodology,” *Int. J.*

*Eng. Comput. Sci.*, vol. 2, no. 12, 2013.

[20] “Deep learning,” *Wikipedia*. 14-Jan-2019.

[21] D. C. Cireşan, U. Meier, L. M. Gambardella, and J. Schmidhuber, “Deep, big, simple neural nets

for handwritten digit recognition,” *Neural Comput.*, vol. 22, no. 12, pp. 3207–3220, 2010.

[22] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Computer*

*Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*,

2005, vol. 1, pp. 886–893.

[23] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Trans. Syst. ManCybern.*, vol.

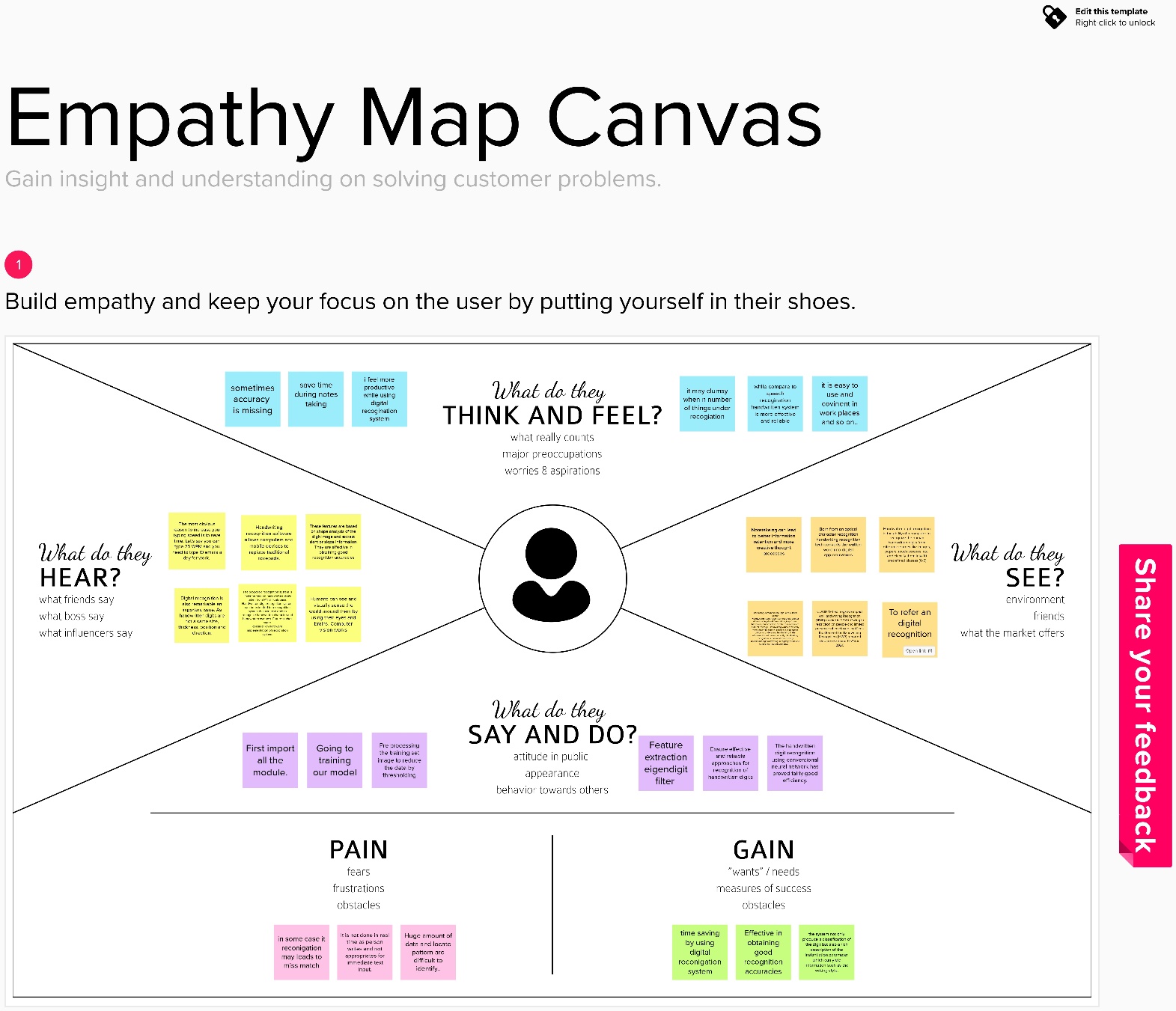
* 1. **PROBLEM STATEMENT:**

This is a collection of thousands of handwritten pictures used to train classification models using machine learning Techniques. As a part of this problem statement, we will train a multilayer perceptron using tensor flow.

**3. IDEATION & PROPOSED SOLUTION**

**3.1 EMPATHY MAP :**

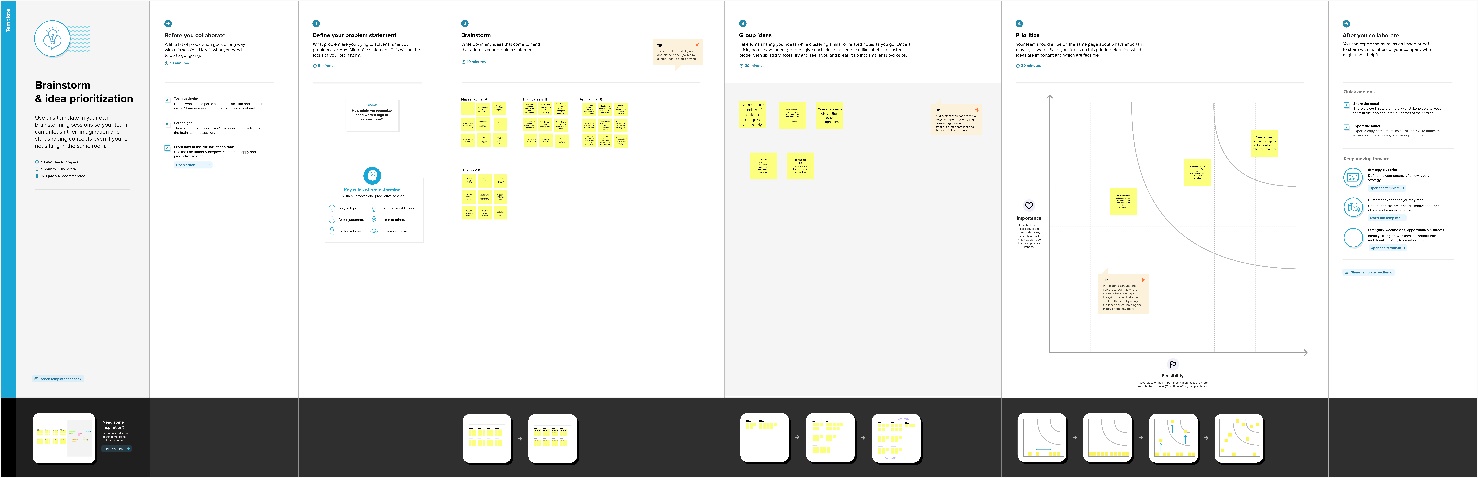
An emoathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user personal, an empathy map can represent a group of users, such as customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.

****

**3.2 BRAIN STROM:**

Brainstorming is a group creativity technique by which efforts are made to find a conclusion for a specific problem by gathering by list of ideas spontaneously contributed by its members.

* Pick an appropriate facilitator.
* Set the agenda.
* Holding the session.



**3.3 PROPOSED SOLUTION:**

Key rationale toward optical character recognition (OCR) from handwritten image includes features extraction technique supported by a classification algorithm for recognition of characters based on the features. Previously, several algorithms for feature classifications and extraction have been utilized for the purpose of character recognition. But, with the advent of CNN in deep learning, no separate algorithms are required for this purpose. However, in the area of computer vision, deep learning is one of the outstanding performers for both feature extraction and classification

However, DNN architecture consists of many nonlinear hidden layers with a enormous number of connections and parameters. Therefore, to train the network with very less amount of samples is a very difficult task. In CNN, only few set of parameters are needed for training of the system. So, CNN is the key solution capable to map correctly datasets for both input and output by varying the trainable parameters and number of hidden layers with high accuracy [[49](https://link.springer.com/article/10.1007/s42452-019-1161-5#ref-CR49)]. Hence, in this work, CNN architecture with Deeplearning4j (DL4J) framework is considered as best fit for the character recognition from the handwritten digit images. For the experiments and verification of system’s performance, the normalized standard MNIST dataset is utilized.

**3.4 PROBLEM SOLUTION FIT:**

**Explore limitation to buy**

**1.**No possibility of obtaining information about the type of input.

**Different from competitors**

**1.**Improved accuracy

**2.**Increased datasets for recognition of digits.

**Who is your customer?**

**1.**Data entry processing.

**2.**Bank check processing

**Focus on problem**

**1.**Understand the relevant information to be useful for user.

**Cause of problem**

**1.**There is a wide range of good and bad handwritting makes tricky to programmers to identify every chharacters.

**Existing behaviour**

**1.**Previous existing system cannot include more number of datasets to compare ans recognise

**2.**Accuracy was not that much effective earlier.

**Design triggers**

**1.**Comparinf with human recognition, it’s easier to recognise with machine.

**Adding emotions**

**1.**Develop a sketch which reflects the writter’s outlays,fears,honesty,mental state etc..

**Solution guess**

**1.**Solution will be the recognition of handwritten digits by various digits of humans to understand by machine.

**Where our customer**

**1.**Banking sector

**2.**Cheque book

**3.**Digit scanning

**4.REQUIREMENT ANALYSIS:**

|  |  |  |
| --- | --- | --- |
| **FR**  **No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail. |
| NFR-2 | **Security** | 1) The system generates a thorough description of the instantiation parameters, which might |

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |
| --- | --- |
| **FR**  **No.** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | Image Data: Handwritten digit recognition refers to a computer's capacity to identify human handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categories them into ten established classifications (0-9).  In the realm of deep learning, this has been the subject of countless studies. |
| FR-2 | Website: Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you've ever visited. The type of hosting determines how much space is allotted to a website on a server. Shared, dedicated, VPS, and reseller hosting are the four basic varieties. |
| FR-3 | Digit Classifier Model: To train a convolutional network to predict the digit from an image, use the MNIST database of handwritten digits. get the training and validation data first. |
| FR-4 | Cloud: The cloud offers a range of IT services, including virtual storage, networking, servers, databases, and applications. In plain English, cloud computing is described as a virtual platform that enables unlimited storage and access to your data over the internet. |
| FR-5 | Modified National Institute of Standards and Technology dataset: The abbreviation MNIST stands for the MNIST dataset. It is a collection of 60,000 tiny square grayscale photographs, each measuring 28 by 28, comprising handwritten single digits between 0 and 9. |

**Non-functional Requirements:**

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

|  |  |  |
| --- | --- | --- |
|  |  | reveal information like the writing style, in addition to a categorization of the digit. 2) The generative models are capable of segmentation driven by recognition.  3) The procedure uses a relatively. |
| NFR-3 | **Reliability** | The samples are used by the neural network to automatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantity of training instances.  Numerous techniques and algorithms, such as Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc., can be used to recognise handwritten numbers. |
| NFR-4 | **Accuracy** | With typed text in high-quality photos, optical character recognition (OCR) technology offers accuracy rates of greater than 99%. However, variances in spacing, abnormalities in handwriting, and the variety of human writing styles result in less precise character identification. |
| NFR-5 | **Availability** |  |

**5.PROJECT DESIGN:**

**5.1 DATA FLOW DIAGRAM:**

It's easy to understand the flow of data through systems with the right [data flow diagram software](https://www.lucidchart.com/pages/examples/data-flow-diagram-software). This guide provides everything you need to know about data flow diagrams, including definitions, history, and symbols and notations. You'll learn the different levels of a DFD, the difference between a logical and a physical DFD and tips for making a DFD.

# Example: DFD Level 0 (Industry Standard)

**Simplified diagram:**

# 5.2 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 |

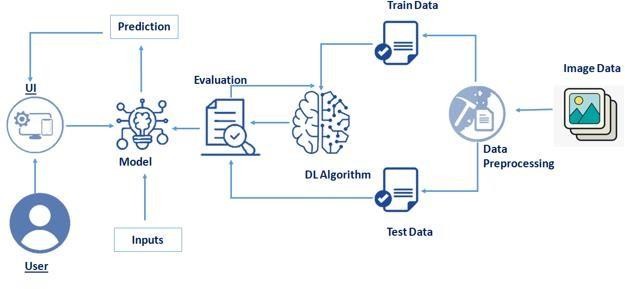
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | USN-3 | As a user, I can read the instructions to use this application. | I can read instructions also to use it in a user- friendly method. | Low | Sprint-2 |
|  | Recognize | USN-4 | As a user, In this prediction page I get 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 |

* 1. **SOLUTION ACHITECTURE:**

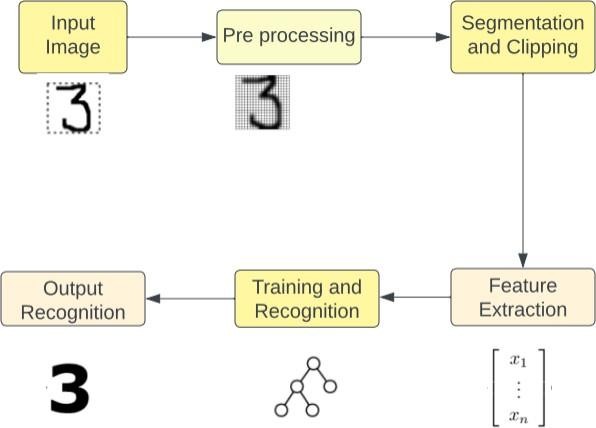
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

* Find the best tech solution to solve existing business problems.
* Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
* Define features, development phases, and solution requirements.
* Provide specifications according to which the solution is defined, managed, and delivered.

# Solution Architecture Diagram:



*Figure 1: Architecture of the Handwritten digit recognition system.*



*Figure 2: Model Architecture of the Deep Learning model used for Handwritten digit recognition.*

**6.PROJECT PLANNING & SCHEDULING**

**6.1 SPRINT PLAN:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Data Collection | USN-1 | As a user, I can collect the dataset from various resources with different handwritings. | 10 | Low | Ashwini V |
| Sprint-1 | Data Preprocessing | USN-2 | As a user, I can load the dataset, handling the missing data, scaling and split data into train and test. | 10 | Medium | N.Hemapriya |
| Sprint-2 | Model Building | USN-3 | As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit. | 5 | High | Priyanga R |
| Sprint-2 | Add CNN layers | USN-4 | Creating the model and adding the input, hidden, and output layers to it. | 5 | High | S.Sindhujaa |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-2 | Compiling the model | USN-5 | With both the training data defined and model defined, it's time to configure the learning process. | 2 | Medium | Ashwini V |
| Sprint-2 | Train & test the model | USN-6 | As a user, let us train our model with our image dataset. | 6 | Medium | N.Hemapriya |
| Sprint-2 | Save the model | USN-7 | As a user, the model is saved & integrated with an android application or web application in order to predict something. | 2 | Low | R.Priyanga |
| Sprint-3 | Building UI Application | USN-8 | As a user, I will upload the handwritten digit image to the application by clicking a upload button. | 5 | High | S.Sindhujaa |
| Sprint-3 |  | USN-9 | As a user, I can know the details of the fundamental usage of the application. | 5 | Low | AshwinV |
| Sprint-3 |  | USN-10 | As a user, I can see the predicted / recognized digits in the application. | 5 | Medium | N.Hemapriya,  S.Sindhujaa |
| Sprint-4 | Train the model on IBM | USN-11 | As a user, I train the model on IBM and integrate flask/Django with scoring end point. | 10 | High | Ashwini V  N.Hemapriya |
| Sprint-4 | Cloud Deployment | USN-12 | As a user, I can access the web application and make the use of the product from anywhere. | 10 | High | R.priyanga  S.Sindhujaa |

**6.2 SPRINT DELIVERY:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

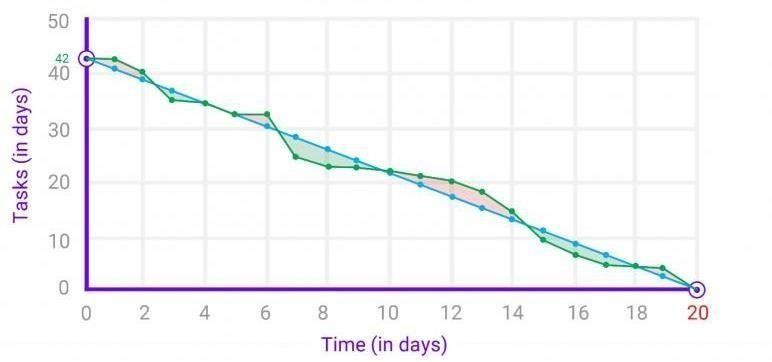
**Velocity:**

Imagine we have a 10-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)

Average Velocity = 20 / 6 = 3.33

**Burndown Chart:**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile [software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum.](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/) However, burn down charts can be applied to any project containing measurable progress over time.



**7 CODING AND SOLUTION:**

{

"cells": [

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Import the necessary packages"

]

},

{

"cell\_type": "code",

"execution\_count": 17,

"metadata": {},

"outputs": [],

"source": [

"import numpy as npm\n",

"import pandas as pds\n",

"import matplotlib.pyplot as plt\n",

"from keras.utils import np\_utils\n",

"from tensorflow.keras.datasets import mnist\n",

"from tensorflow.keras.models import Sequential\n",

"from tensorflow.keras.layers import Conv2D, Dense, Flatten\n",

"from tensorflow.keras.optimizers import Adam\n",

"from tensorflow.keras.models import load\_model\n",

"from PIL import Image, ImageOps"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Load data"

]

},

{

"cell\_type": "code",

"execution\_count": 2,

"metadata": {},

"outputs": [],

"source": [

"(X\_train, y\_train), (X\_test, y\_test) = mnist.load\_data()"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Data Analysis"

]

},

{

"cell\_type": "code",

"execution\_count": 3,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"(60000, 28, 28)\n",

"(10000, 28, 28)\n"

]

}

],

"source": [

"print(X\_train.shape)\n",

"print(X\_test.shape)"

]

},

{

"cell\_type": "code",

"execution\_count": 4,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3,\n",

" 18, 18, 18, 126, 136, 175, 26, 166, 255, 247, 127, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 30, 36, 94, 154, 170,\n",

" 253, 253, 253, 253, 253, 225, 172, 253, 242, 195, 64, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 49, 238, 253, 253, 253, 253,\n",

" 253, 253, 253, 253, 251, 93, 82, 82, 56, 39, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 18, 219, 253, 253, 253, 253,\n",

" 253, 198, 182, 247, 241, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 80, 156, 107, 253, 253,\n",

" 205, 11, 0, 43, 154, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 1, 154, 253,\n",

" 90, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 139, 253,\n",

" 190, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 190,\n",

" 253, 70, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 35,\n",

" 241, 225, 160, 108, 1, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 81, 240, 253, 253, 119, 25, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 45, 186, 253, 253, 150, 27, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 16, 93, 252, 253, 187, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 249, 253, 249, 64, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 46, 130, 183, 253, 253, 207, 2, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 39,\n",

" 148, 229, 253, 253, 253, 250, 182, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 114, 221,\n",

" 253, 253, 253, 253, 201, 78, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 23, 66, 213, 253, 253,\n",

" 253, 253, 198, 81, 2, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 18, 171, 219, 253, 253, 253, 253,\n",

" 195, 80, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 55, 172, 226, 253, 253, 253, 253, 244, 133,\n",

" 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 136, 253, 253, 253, 212, 135, 132, 16, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0]], dtype=uint8)"

]

},

"execution\_count": 4,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"X\_train[0]"

]

},

{

"cell\_type": "code",

"execution\_count": 5,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5"

]

},

"execution\_count": 5,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_train[0]"

]

},

{

"cell\_type": "code",

"execution\_count": 6,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.image.AxesImage at 0x1ed310b5930>"

]

},

"execution\_count": 6,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {},

"output\_type": "display\_data"

}

],

"source": [

"plt.imshow(X\_train[0])"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Data Pre-Processing"

]

},

{

"cell\_type": "code",

"execution\_count": 7,

"metadata": {},

"outputs": [],

"source": [

"X\_train = X\_train.reshape(60000, 28, 28, 1).astype('float32')\n",

"X\_test = X\_test.reshape(10000, 28, 28, 1).astype('float32')"

]

},

{

"cell\_type": "code",

"execution\_count": 8,

"metadata": {},

"outputs": [],

"source": [

"number\_of\_classes = 10\n",

"Y\_train = np\_utils.to\_categorical(y\_train, number\_of\_classes)\n",

"Y\_test = np\_utils.to\_categorical(y\_test, number\_of\_classes)"

]

},

{

"cell\_type": "code",

"execution\_count": 9,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

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

]

},

"execution\_count": 9,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"Y\_train[0]"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Create model"

]

},

{

"cell\_type": "code",

"execution\_count": 10,

"metadata": {},

"outputs": [],

"source": [

"model = Sequential()\n",

"model.add(Conv2D(64, (3, 3), input\_shape=(28, 28, 1), activation=\"relu\"))\n",

"model.add(Conv2D(32, (3, 3), activation=\"relu\"))\n",

"model.add(Flatten())\n",

"model.add(Dense(number\_of\_classes, activation=\"softmax\"))"

]

},

{

"cell\_type": "code",

"execution\_count": 11,

"metadata": {},

"outputs": [],

"source": [

"model.compile(loss='categorical\_crossentropy', optimizer=\"Adam\", metrics=[\"accuracy\"])"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Train the model"

]

},

{

"cell\_type": "code",

"execution\_count": 12,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Epoch 1/5\n",

"1875/1875 [==============================] - 16s 5ms/step - loss: 0.2158 - accuracy: 0.9518 - val\_loss: 0.0964 - val\_accuracy: 0.9707\n",

"Epoch 2/5\n",

"1875/1875 [==============================] - 9s 5ms/step - loss: 0.0682 - accuracy: 0.9794 - val\_loss: 0.0674 - val\_accuracy: 0.9805\n",

"Epoch 3/5\n",

"1875/1875 [==============================] - 9s 5ms/step - loss: 0.0478 - accuracy: 0.9844 - val\_loss: 0.0852 - val\_accuracy: 0.9759\n",

"Epoch 4/5\n",

"1875/1875 [==============================] - 9s 5ms/step - loss: 0.0336 - accuracy: 0.9893 - val\_loss: 0.1202 - val\_accuracy: 0.9719\n",

"Epoch 5/5\n",

"1875/1875 [==============================] - 9s 5ms/step - loss: 0.0270 - accuracy: 0.9914 - val\_loss: 0.1036 - val\_accuracy: 0.9777\n"

]

},

{

"data": {

"text/plain": [

"<keras.callbacks.History at 0x1ed3324f7f0>"

]

},

"execution\_count": 12,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"model.fit(X\_train, Y\_train, batch\_size=32, epochs=5, validation\_data=(X\_test,Y\_test))"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Test the model"

]

},

{

"cell\_type": "code",

"execution\_count": 13,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Metrics (Test Loss & Test Accuracy): \n",

"[0.1035672277212143, 0.9776999950408936]\n"

]

}

],

"source": [

"metrics = model.evaluate(X\_test, Y\_test, verbose=0)\n",

"print(\"Metrics (Test Loss & Test Accuracy): \")\n",

"print(metrics)"

]

},

{

"cell\_type": "code",

"execution\_count": 14,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"1/1 [==============================] - 0s 177ms/step\n",

"[[6.43197941e-15 8.71634543e-21 7.98728167e-11 7.08215517e-12\n",

" 2.27718335e-18 1.36703092e-15 2.37176042e-22 1.00000000e+00\n",

" 4.51405352e-13 4.25453591e-13]\n",

" [4.56659687e-15 1.54588287e-10 1.00000000e+00 1.20107971e-13\n",

" 1.86926159e-19 3.90255250e-20 1.16102319e-11 4.27834925e-23\n",

" 7.33884963e-17 1.86307852e-23]\n",

" [1.37352282e-10 9.99961138e-01 3.40877750e-06 1.50240779e-12\n",

" 1.99599867e-07 1.10004057e-05 6.72304851e-11 7.78906983e-09\n",

" 2.42337919e-05 3.74607870e-13]\n",

" [1.00000000e+00 5.39840355e-16 1.03082355e-10 4.23198737e-17\n",

" 8.17481194e-10 2.49619574e-12 1.66041558e-09 5.06253395e-17\n",

" 3.02219919e-13 5.55243709e-08]]\n"

]

}

],

"source": [

"prediction = model.predict(X\_test[:4])\n",

"print(prediction)"

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"[7 2 1 0]\n",

"[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]\n",

" [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]\n",

" [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]\n",

" [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]\n"

]

}

],

"source": [

"print(numpy.argmax(prediction, axis=1))\n",

"print(Y\_test[:4])"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Save the model"

]

},

{

"cell\_type": "code",

"execution\_count": 16,

"metadata": {},

"outputs": [],

"source": [

"model.save(\"model.h5\")"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Test the saved model"

]

},

{

"cell\_type": "code",

"execution\_count": 22,

"metadata": {},

"outputs": [],

"source": [

"model=load\_model(\"model.h5\")"

]

},

{

"cell\_type": "code",

"execution\_count": 23,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"1/1 [==============================] - 0s 435ms/step\n",

"0 8\n",

"Name: Label, dtype: int64\n"

]

}

],

"source": [

"img = Image.open(\"sample.png\").convert(\"L\")\n",

"img = img.resize((28, 28))\n",

"img2arr = npm.array(img)\n",

"img2arr = img2arr.reshape(1, 28, 28, 1)\n",

"results = model.predict(img2arr)\n",

"results = npm.argmax(results,axis = 1)\n",

"results = pds.Series(results,name=\"Label\")\n",

"print(results)"

]

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3.10.8 ('venv': venv)",

"language": "python",

"name": "python3"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.10.8"

},

"orig\_nbformat": 4,

"vscode": {

"interpreter": {

"hash": "72cf82f53b15019b5b640600623df8bcf4d62c2c60fee1ea51c8c07b395bb5c2"

}

}

},

"nbformat": 4,

"nbformat\_minor": 2 }

**8. TESTING**

* 1. **TEST CASE:**

{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": []

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

}

},

"cells": [

{

"cell\_type": "markdown",

"source": [

"Sprint-2\n",

"Team ID-PNT2022TMID39529"

],

"metadata": {

"id": "G4T0oInf8hgU"

}

},

{

"cell\_type": "markdown",

"source": [

"Import necessary package"

],

"metadata": {

"id": "BU0PScv980Zy"

}

},

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {

"id": "JYsTTpKb8UFP"

},

"outputs": [],

"source": [

"import numpy\n",

"import matplotlib.pyplot as plt\n",

"from keras.utils import np\_utils\n",

"from tensorflow.keras.datasets import mnist\n",

"from tensorflow.keras.models import Sequential\n",

"from tensorflow.keras.layers import Conv2D, Dense, Flatten\n",

"from tensorflow.keras.optimizers import Adam"

]

},

{

"cell\_type": "markdown",

"source": [

"Load data"

],

"metadata": {

"id": "uApZbwMa9Bb4"

}

},

{

"cell\_type": "code",

"source": [

"(X\_train, y\_train), (X\_test, y\_test) = mnist.load\_data()\n"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "564SXrH088jn",

"outputId": "2d4a2d41-9b61-4bdd-a4b8-b5133adc9c67"

},

"execution\_count": 2,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz\n",

"11490434/11490434 [==============================] - 0s 0us/step\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"print(X\_train.shape)\n",

"print(X\_test.shape)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "3Vl-bd-B9IzV",

"outputId": "844bda4c-b885-4059-9782-a2328fbb2ed8"

},

"execution\_count": 3,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"(60000, 28, 28)\n",

"(10000, 28, 28)\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"X\_train[0]"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "D7e8u2qK9Sg8",

"outputId": "8bb4a4d1-9e73-4218-8014-82972773795b"

},

"execution\_count": 4,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3,\n",

" 18, 18, 18, 126, 136, 175, 26, 166, 255, 247, 127, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 30, 36, 94, 154, 170,\n",

" 253, 253, 253, 253, 253, 225, 172, 253, 242, 195, 64, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 49, 238, 253, 253, 253, 253,\n",

" 253, 253, 253, 253, 251, 93, 82, 82, 56, 39, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 18, 219, 253, 253, 253, 253,\n",

" 253, 198, 182, 247, 241, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 80, 156, 107, 253, 253,\n",

" 205, 11, 0, 43, 154, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 1, 154, 253,\n",

" 90, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 139, 253,\n",

" 190, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 190,\n",

" 253, 70, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 35,\n",

" 241, 225, 160, 108, 1, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 81, 240, 253, 253, 119, 25, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 45, 186, 253, 253, 150, 27, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 16, 93, 252, 253, 187, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 249, 253, 249, 64, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 46, 130, 183, 253, 253, 207, 2, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 39,\n",

" 148, 229, 253, 253, 253, 250, 182, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 114, 221,\n",

" 253, 253, 253, 253, 201, 78, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 23, 66, 213, 253, 253,\n",

" 253, 253, 198, 81, 2, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 18, 171, 219, 253, 253, 253, 253,\n",

" 195, 80, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 55, 172, 226, 253, 253, 253, 253, 244, 133,\n",

" 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 136, 253, 253, 253, 212, 135, 132, 16, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0],\n",

" [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,\n",

" 0, 0]], dtype=uint8)"

]

},

"metadata": {},

"execution\_count": 4

}

]

},

{

"cell\_type": "code",

"source": [

"y\_train[0]"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "1VsWb5sm9VcB",

"outputId": "b7521eea-b655-4144-c4b5-7a1772c31270"

},

"execution\_count": 5,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"5"

]

},

"metadata": {},

"execution\_count": 5

}

]

},

{

"cell\_type": "code",

"source": [

"plt.imshow(X\_train[0])"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 282

},

"id": "mXNmeoHL9bYS",

"outputId": "801b729b-8e15-42f6-8943-331e9f8be4ad"

},

"execution\_count": 6,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<matplotlib.image.AxesImage at 0x7f34394baa90>"

]

},

"metadata": {},

"execution\_count": 6

},

{

"output\_type": "display\_data",

"data": {

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

],

"image/png": "\n"

},

"metadata": {

"needs\_background": "light"

}

}

]

},

{

"cell\_type": "markdown",

"source": [

"Data pre processing"

],

"metadata": {

"id": "ViwBwMJf9nM7"

}

},

{

"cell\_type": "code",

"source": [

"X\_train = X\_train.reshape(60000, 28, 28, 1).astype('float32')\n",

"X\_test = X\_test.reshape(10000, 28, 28, 1).astype('float32')"

],

"metadata": {

"id": "PA7e1C1S9ftz"

},

"execution\_count": 7,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"number\_of\_classes = 10\n",

"Y\_train = np\_utils.to\_categorical(y\_train, number\_of\_classes)\n",

"Y\_test = np\_utils.to\_categorical(y\_test, number\_of\_classes)"

],

"metadata": {

"id": "ps3cxWyZ9rMu"

},

"execution\_count": 8,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"Y\_train[0]"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "lPpmgI5M9vlY",

"outputId": "24f2b170-c2c1-41a5-f492-7f637fc8ba19"

},

"execution\_count": 9,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

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

]

},

"metadata": {},

"execution\_count": 9

}

]

},

{

"cell\_type": "markdown",

"source": [

"Create model"

],

"metadata": {

"id": "kv6cYS9593Xz"

}

},

{

"cell\_type": "code",

"source": [

"model = Sequential()\n",

"model.add(Conv2D(64, (3, 3), input\_shape=(28, 28, 1), activation=\"relu\"))\n",

"model.add(Conv2D(32, (3, 3), activation=\"relu\"))\n",

"model.add(Flatten())\n",

"model.add(Dense(number\_of\_classes, activation=\"softmax\"))"

],

"metadata": {

"id": "WkvfmMs-9y0P"

},

"execution\_count": 10,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model.compile(loss='categorical\_crossentropy', optimizer=\"Adam\", metrics=[\"accuracy\"])"

],

"metadata": {

"id": "nS6zuQzF96Up"

},

"execution\_count": 11,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model.fit(X\_train, Y\_train, batch\_size=32, epochs=5, validation\_data=(X\_test,Y\_test))"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "r1InLlgH9-2j",

"outputId": "3a6e787b-6740-461c-c5f2-11fcd9369ade"

},

"execution\_count": 12,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Epoch 1/5\n",

"1875/1875 [==============================] - 202s 107ms/step - loss: 0.2126 - accuracy: 0.9521 - val\_loss: 0.0894 - val\_accuracy: 0.9732\n",

"Epoch 2/5\n",

"1875/1875 [==============================] - 204s 109ms/step - loss: 0.0623 - accuracy: 0.9814 - val\_loss: 0.1000 - val\_accuracy: 0.9713\n",

"Epoch 3/5\n",

"1875/1875 [==============================] - 204s 109ms/step - loss: 0.0425 - accuracy: 0.9864 - val\_loss: 0.0851 - val\_accuracy: 0.9758\n",

"Epoch 4/5\n",

"1875/1875 [==============================] - 198s 105ms/step - loss: 0.0346 - accuracy: 0.9890 - val\_loss: 0.1026 - val\_accuracy: 0.9763\n",

"Epoch 5/5\n",

"1875/1875 [==============================] - 205s 109ms/step - loss: 0.0253 - accuracy: 0.9916 - val\_loss: 0.0980 - val\_accuracy: 0.9777\n"

]

},

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<keras.callbacks.History at 0x7f3434c97a90>"

]

},

"metadata": {},

"execution\_count": 12

}

]

},

{

"cell\_type": "markdown",

"source": [

"Test the model"

],

"metadata": {

"id": "GajjLCQFCtcV"

}

},

{

"cell\_type": "code",

"source": [

"metrics = model.evaluate(X\_test, Y\_test, verbose=0)\n",

"print(\"Metrics (Test Loss & Test Accuracy): \")\n",

"print(metrics)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "0vwkOGRvB-YJ",

"outputId": "6106f6ac-25a8-465e-9e3b-d1c8e2c73bdf"

},

"execution\_count": 13,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Metrics (Test Loss & Test Accuracy): \n",

"[0.09800956398248672, 0.9776999950408936]\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"prediction = model.predict(X\_test[:4])\n",

"print(prediction)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "CvzsM8cRCXb0",

"outputId": "b20715c7-016d-47d6-c7a0-5f0d975cfe41"

},

"execution\_count": 14,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"1/1 [==============================] - 0s 128ms/step\n",

"[[2.9166860e-11 7.8674158e-21 6.8197568e-09 9.6392165e-09 1.2284079e-15\n",

" 2.9200296e-16 1.2543796e-18 1.0000000e+00 6.8977801e-12 8.5674190e-11]\n",

" [2.2934597e-05 6.7787391e-07 9.9997628e-01 4.7655574e-10 7.8161123e-12\n",

" 3.1815811e-15 4.4551697e-08 2.4422626e-14 6.2133083e-08 5.8212260e-17]\n",

" [4.5624104e-09 9.9988461e-01 3.9433194e-09 3.0301920e-12 9.3741266e-07\n",

" 1.5071736e-10 1.4758221e-10 3.4230155e-10 1.1441124e-04 2.0948462e-11]\n",

" [1.0000000e+00 8.1227650e-20 6.9074791e-13 4.1791107e-17 6.7886536e-15\n",

" 4.3934654e-14 2.0079671e-11 8.7713697e-16 4.4645889e-16 4.8445514e-14]]\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"print(numpy.argmax(prediction, axis=1))\n",

"print(Y\_test[:4])"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "i-JtdfDqC8fD",

"outputId": "221ddb4f-97ed-4da2-a80a-b1d5033d5dc3"

},

"execution\_count": 15,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"[7 2 1 0]\n",

"[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]\n",

" [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]\n",

" [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]\n",

" [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]\n"

]

}

]

}

]

}

1. **RESULT:**

we have successfully built a Python deep learning project on handwritten digit recognition app. We have built and trained the Convolutional neural network which is very effective for image classification purposes. Later on, we build the GUI where we draw a digit on the canvas then we classify the digit and show the results.

1. **ADVANTAGE AND DISADVANTAGE:**

To combat the loss of life due to illegible script, health care facilities are currently implementing digital data strategies, such as electronic health records (EHR). A 2013 survey found that 71 percent of physicians have adopted EHR. Such policies also help facilities lessen their dependence on hard paper, mitigate regulatory violations and compliance risks, as well as forgeries that lead to fraud.

This comes at a time when vehicle infotainment systems are becoming increasingly sophisticated. One technology that helps ensure a safe, enjoyable user experience is handwriting recognition capabilities.

1. **CONCLUSIONS:**

The paper discusses in detail all advances in the area of handwritten character recognition. The most accurate solution provided in this area directly or indirectly depends upon the quality as well as the nature of the material to be read. Various techniques have been described in this paper for character recognition in handwriting recognition system A sort comparison is shown between the different methods proposed so far in table 1. From the study done so far, it is analysed that the selection of the classification as well as the

feature extraction techniques needs to be proper in order to

attain good rate in recognizing the character. Studies in the

paper reveals that there is still scope of enhancing the

algorithms as well as enhancing the rate of recognition of

characters. The paper discusses in detail all advances in the

area of handwritten character recognition. The most accurate

solution provided in this area directly or indirectly depends

upon the quality as well as the nature of the material to be

read. Various techniques have been described in this paper

for character recognition in handwriting recognition system.

A sort comparison is shown between the different methods

proposed so far in table 1. From the study done so far, it is

analysed that the selection of the classification as well as the

feature extraction techniques needs to be proper in order to

attain good rate in recognizing the character. Studies in the

paper reveals that there is still scope of enhancing the

algorithms as well as enhancing the rate of recognition of characters.

**12.Future Scope:**

The future development of the applications based on algorithms of deep and machine learning is practically boundless. In the future, we can work on a denser or hybrid algorithm than the current set of algorithms with more manifold data to achieve the solutions to many problems. In future, the application of these algorithms lies from the public to high-level authorities, as from the differentiation of the algorithms above and with future development we can attain high-level functioning applications which can be used in the classified or government agencies as well as for the common people, we can use these algorithms in hospitals application for detailed medical diagnosis, treatment and monitoring the patients, we can use it in surveillances system to keep tracks of the suspicious activity under the system, in fingerprint and retinal scanners, database filtering applications, Equipment checking for national forces and many more problems of both major and minor category. The advancement in this field can help us create an environment of safety, awareness and comfort by using these algorithms in day-to-day application and high-level application (i.e., corporate level or Government level). Application-based on artificial intelligence and deep learning is the future of the technological world because of their absolute accuracy and advantages over many major problems.

Git hub link:

http://github.com/IBM-EPBL/IBM-Project-13832-1659532724