

Handwritten Text Recognition using Deep Learning

A PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY

Submitted by

Medicharla Satya(20B81A1272)

Maddipati Komali(20B81A1265)

Punnana Venkat Naidu(20B81A1299)

Tadianda H.S.V.Aditya Raja(20B81A12B0)

Rayi Sri Teja(20B81A1285)

Under the Esteemed Guidance of

Mr. G. Vihari

Assistant Professor



DEPARTMENT OF INFORMATION TECHNOLOGY

SIR C. R. REDDY COLLEGE OF ENGINEERING

APPROVED BY AICTE

ELURU – 534007

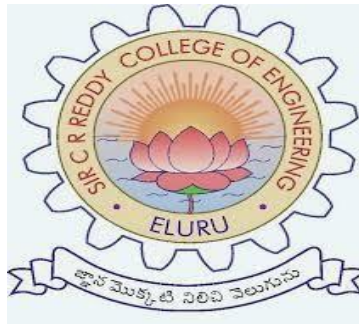
2023 – 2024

SIR C.R. REDDY COLLEGE OF ENGINEERING

DEPARTMENT OF INFORMATION TECHNOLOGY

Affiliated to Jawaharlal Nehru Technological University, Kakinada

Eluru-534007



BONAFIDE CERTIFICATE

This is to certify that the project report entitled “**HANDWRITTEN TEXT RECOGNITION USING DEEP LEARNING**” being submitted by **M.Satya(20B81A1272)**, **M.Komali (20B81A1265)**, **P.Venkat Naidu(20B81A1299)**, **T.H.S.V.Aditya Raja(20B81A12B0)**, **R.Sri Teja(20B81A1285)** in partial fulfillment for the Bachelor of Technology in Information Technology to the Jawaharlal Technological University is a record of bonafide work carried out under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree.

Mr. G. VIHARI

Project Guide & Asst.Prof

DEPARTMENT OF IT

Sir CRR College Of Engineering

Dr. A. YESU BABU

Prof & HOD

DEPARTMENT OF IT

Sir CRR College Of Engineering

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

First and foremost, I give thanks to almighty God for all his blessings that he bestowed upon me without which I won't be where I am today.

We express my sincere thanks to our principal, **Dr. K.VENKATESWARA RAO** for providing me the necessary infrastructure and his guidance in the efficient completion of the project work.

We owe a great deal of gratitude to our beloved Head Of Department, **Dr. A.YESUBABU** for his continuous support in the journey of my course, for his patience, motivation, enthusiasm and immense knowledge.

We owe a great deal of gratitude to my project guide **Mr. G. VIHARI** for his continuous support in the journey of my course, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all the time of work and completing this project successfully.

Our special thanks to librarian **Smt. D. LAKSHMI KUMARI**, and to the entire library staff Sir C.R.R College of Engineering, for providing the necessary library facilities.

We express our earnest thanks to faculty members and non-teaching staff of IT for extending their valuable support.

PROJECT MEMBERS

MEDICHARLA SATYA	(20B81A1272)
MADDIPATI KOMALI	(20B81A1265)
PUNNANA VENKATA NAIDU	(20B81A1299)
TADINADA H.V.S. ADITYA RAJA	(20B81A12B0)
RAYI SRI TEJA	(20B81A1285)

ABSTRACT

Handwriting Detection is a technique or ability of a computer to receive and interpret intelligible handwritten input from source such as paper documents, touch screen, photographs etc. Handwritten Text recognition is one of area pattern recognition. The purpose of pattern recognition is to categorize or classification data or object of one of the classes or categories. Traditional systems of handwriting recognition have relied on handcrafted features and a large amount of prior knowledge. Training an character recognition (OCR) system based on these prerequisites is a challenging task. Convolutional neural networks (CNNs) Optical and recurrent neural network (RNN) are very effective in perceiving the structure of hand written characters/words. Research in the handwriting recognition field is focused on deep learning techniques and has achieved breakthrough performance in the last few years. Still, the rapid growth in the amount of handwritten data and the availability of massive processing power demands improvement in recognition accuracy and deserves further investigation. Convolutional neural networks are very effective in perceiving the structure of hand written characters/words in ways that help in automatic extraction of distinct features and make CNN the most suitable for solving handwriting recognition problems. This system will be applied to detect the writings of different format. The development of handwriting is more sophisticated, which is found various kinds of handwritten character such as digit, numeral, cursive script, symbols, and scripts including English and other languages.

KEYWORDS: CNN, RNN, CTC, Tensor Flow, OCR, SoftMax

DECLARATION

We hereby declare that the dissertation entitled **CRR IT HANDBOOK** submitted to the B. Tech degree is my original work and the dissertation has not formed the basis for the award of any degree, fellowship or any other similar titles.

MEDICHARLA SATYA	(20B81A1272)
MADDIPATI KOMALI	(20B81A1265)
PUNNANA VENKATA NAIDU	(20B81A1299)
TADINADA H.V.S. ADITYA RAJA	(20B81A12B0)
RAYI SRI TEJA	(20B81A1285)

TABLE OF CONTENTS

ACKNOWLEDGEMENT	I
ABSTRACT	II
DECLARATION	III
TABLE OF CONTENT	IV
LIST OF FIGURES	VI
LIST OF ABBRIVATIONS	VII
1. INTRODUCTION	1
2. LITERATURE SURVEY	5
3. EXISTING SYSTEM	9
4. PROPOSED SYSTEM	10
4.1 PROBLEM STATEMENT	10
4.2 SCOPE	10
4.3 OBJECTIVE	10
4.4 PROPOSED SYSTEM	11
5. REQUIREMENT ANALYSIS	13
5.1 INTRODUCTION	13
5.2 FUNCTIONAL REQUIREMENTS	13
5.3 NON-FUNCTIONAL REQUIREMENTS	13
5.4 TECHNICAL REQUIREMENTS	13
6. DESIGN AND METHODOLOGIES	15
6.1 PROCESS MODEL	15
6.2 UML DIAGRAMS	21
7. IMPLEMENTATION	23
7.1 MODULES	23
7.2 TECHNOLOGIES DESCRIPTION	23

	V
7.3 LIBRARIES	25
7.4 INTRODUCTION TO PYTHON	25
8. MODULE DIVISION	28
8.1 MODULES	28
8.1.1 DATA LOADER MODULE	28
8.1.2 MAIN MODULE	28
8.1.3 MODEL MODULE	29
8.1.4 SAMPLE PRE-PROCESSOR MODULE	29
8.1.5 SPELL CHECKER MODULE	29
8.1.6 UPLOAD MODULE	29
9. SAMPLE CODE	30
10. TESTING	31
11. RESULTS&DISCUSSION	34
11.1 INPUT SCREENS	34
11.2 OUTPUT SCREENS	35
12. CONCLUSION AND FUTURE SCOPE	40
13. REFERENCES	41

LIST OF FIGURES

Figure No	Description	Page No
1.1	Venn-Diagram for AI, ML, NLP&DL	2
1.2	DL – Neural Network Architecture	2
1.3	Convolutional Neural Network Architecture	4
4.1	Image of word taken from IAM Dataset	11
4.2	Block diagram of proposed handwritten character classification	12
6.1	Model	16
6.2	Process Flow Diagram for Model Creation	17
6.3	Proposed Model Architecture	17
6.4	Model Architecture	19
6.5	Sample Proposed Performance	20
6.6	LSTM Architecture	20
6.7	Control Flow Diagram for HCR System	21
6.8	Deployment Diagram	22
6.9	Class Diagram	22

LIST OF ABBRIVATIONS

DL	Deep Learning
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
OCR	Optical Character Recognition
API	Application Programming Interface

1.INTRODUCTION

1.1 OVERVIEW

An OCR system depends mainly on the extraction of features and discrimination/classification of these features (based on patterns). Handwritten OCR have received increasing attention as a subfield of OCR.

Handwritten Text Recognition is a technology that is much needed in this world as of today. Before proper implementation of this technology, we have relied on writing texts with our own hands which can result in errors. It is difficult to store and access physical data with efficiency. Manual labour is required to maintain proper organization of the data. Throughout history, there has been severe loss of data because of the traditional method of storing data. Modern day technology is letting people store the data over machines, where the storage, organization and accessing of data is relatively easier.

Adopting the use of Handwritten Text Recognition software, it is easier to store and access data that was traditionally stored. Furthermore, it provides more security to the data. One such example of Handwritten text Recognition software is the Google Lens. The aim of our project is to make an application that can recognize the handwriting using concepts of deep learning. We are thinking by approaching our problem using CNN as they provide better accuracy over such tasks.

1.2 BACKGROUND

1.2.1 ARTIFICIAL INTELLEGEANCE

Artificial Intelligence is a branch of computer science that endeavors to replicate or simulate human intelligence in a machine, so machines can perform tasks that typically require human intelligence. Some programmable functions of AI systems include planning, learning, reasoning, problem solving, and decision making.

Artificial intelligence systems are powered by algorithms, using techniques such as machine learning, deep learning and rules. Machine learning algorithms feed computer data to AI systems, using statistical techniques to enable AI systems to learn. Through machine learning, AI systems get progressively better at tasks, without having to be specifically programmed to do so.

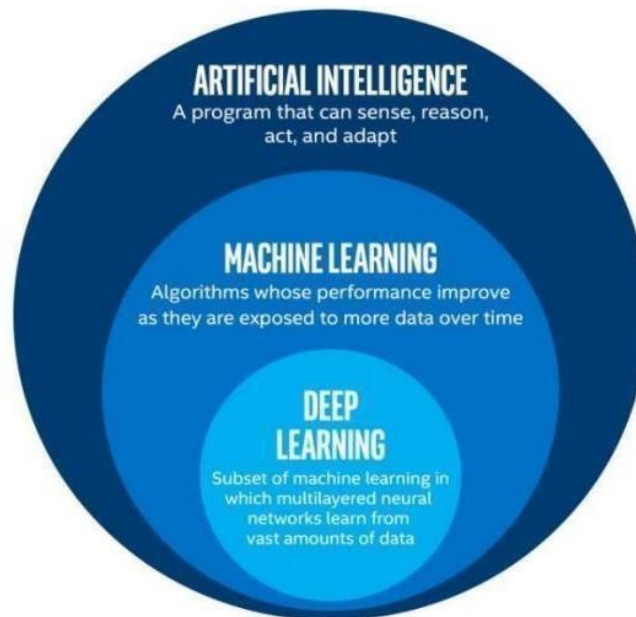


Figure 1.1: Venn Diagram for AI, ML, NLP & DL

1.2.2 DEEP LEARNING

Deep Learning is a subset of Machine Learning, which on the other hand is a subset of Artificial Intelligence. Deep learning algorithms attempt to draw similar conclusions as humans would by continually analyzing data with a given logical structure. To achieve this, deep learning uses a multi-layered structure of algorithms called neural networks.

The design of the neural network is based on the structure of the human brain. Just as we use our brains to identify patterns and classify different types of information, neural networks can be taught to perform the same tasks on data.

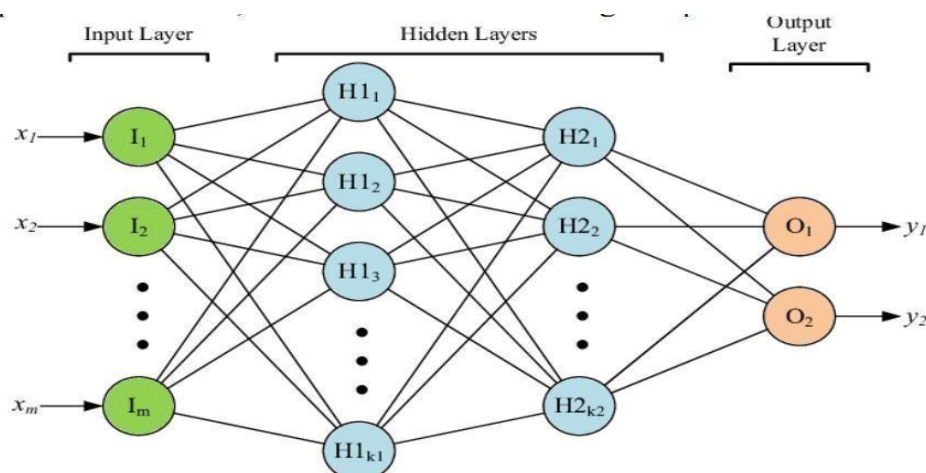


Figure 1.2: DL – Neural Network Architecture

1.2.2 NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) refers to the branch of computer science and more specifically, the branch of artificial intelligence or AI concerned with giving computers the ability to understand text and spoken words in much the same way human being scan. NLP combines computational linguistics rule-based modelling of human language with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to ‘understand’ its full meaning, complete with the speaker or writer’s intent and sentiment.

NLP drives computer programs that translate text from one language to another, respond to spoken commands, and summarize large volumes of text rapidly—even in real time. There’s a good chance we’ve interacted with NLP in the form of voice operated GPS systems, digital assistants, speech-to-text dictation software, customer service chatbots, and other consumer conveniences. But NLP also plays a growing role in enterprise solutions that help streamline business operations, increase employee productivity, and simplify mission-critical business processes

1.2.3 CONVOLUTIONAL NEURAL NETWORK

CNN image classifications take an input image, process it and classify it under certain categories (E.g., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels. Based on the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension). E.g. An image of $6 \times 6 \times 3$ array of matrix of RGB (3 refers to RGB values) and an image of $4 \times 4 \times 1$ array of matrix of grayscale image where 3 and 1 are the number of colour values required to represent an image pixel. The followings are the type of layers which as commonly found in the CNNs:

1. **Convolution** – The main building block of CNN is the convolutional layer which a mathematical operation to merge two sets of information. The convolution is applied on the input data using a convolution filter to produce a feature map. The first layer to extract features from an input image is convolution.
2. **Striding** – The number of shifts over the input image is called stride. Example When stride is 1 then we move the filters to 1 pixel at a time.

3. **Non-Linearity** – The Activation function generally used is ReLU stands for Rectified Linear Unit for a non-linear operation. The output is:

$$\{F(x) = \max(0, x)\}$$

4. **Pooling Layer** – After a convolution operation we usually perform pooling to reduce the dimensionality. This enables us to reduce the number of parameters, which in turn shortens the training time and reduces chances of overfitting. Pooling layers down sample each feature map independently, reducing the height and width, keeping the depth intact. Contrary to the convolution operation, pooling has no parameters. It slides a window over its input, and simply takes the maximum value in the window. The largest element from the feature map is selected by using this operation.

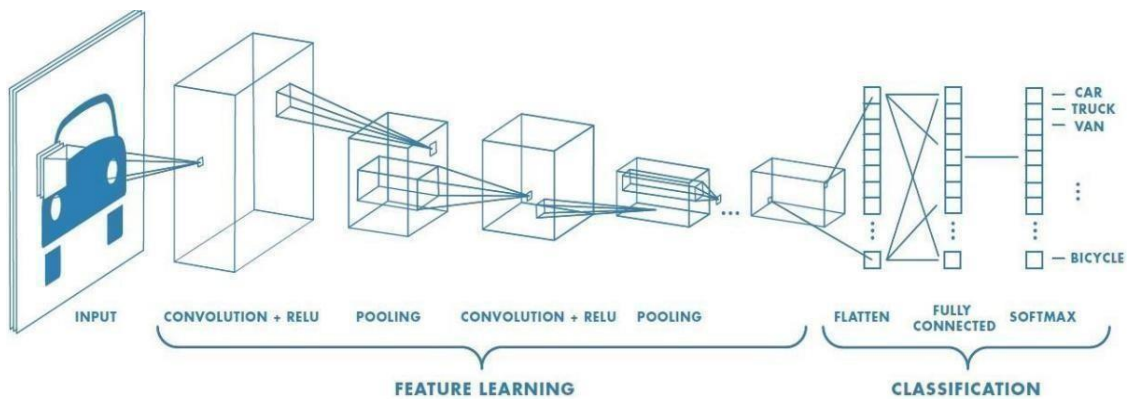


Fig 1.3: Convolutional Neural Network Architecture

2.LITERATURE SURVEY

[1] Normalization Ensemble for Handwritten Character Recognition. - Cheng-Lin Liu and Katsumi Marukawa (2004)

This paper proposes a multiple classifier approach, called normalization ensemble, for handwritten character recognition by combining multiple normalization methods. By varying the coordinate mapping mode, we have devised 14 normalization functions, and switching on/off slant correction results in 28 instantiated classifiers. We would show that the classifiers with different normalization methods are complementary and the combination of them can significantly improve the recognition accuracy. In experiments of handwritten digit recognition on the NIST special database 19, the normalization ensemble was shown to reduce the error rate by factors from 10.6% to 26.9% and achieved the best error rate 0.43%. We also show that the complexity of normalization ensemble can be reduced by selecting seven classifiers from 28 with little loss of accuracy.

[2] Handwriting Recognition using LSTM Networks. - Sarita Yadav, Ankur Pandey, Pulkit Aggarwal, Rachit Garg, Vishal Aggarwal (2018)

This paper proposes Recognizing digits in an optimal way is a challenging problem. Recent deep learning-based approaches have achieved great success on handwriting recognition. English characters are among the most widely adopted writing systems in the world. This paper presents a comparative evaluation of the standard LSTM RNN model with other deep models on MNIST dataset.

[3] Word Recognition Using Fuzzy Logic. - Richard Buse, Zhi-Qiang Liu, and Jim Bezdek(2002)

This paper presents an offline word-recognition system based on structural information in the unconstrained written word. Oriented features in the word are extracted with the Gabor filters. We estimate the Gabor filter parameters from the grayscale images. two-dimensional fuzzy word classification system is developed where the spatial location and shape of the membership functions are derived from the training words. The system achieves an average recognition rate of 74% for the word being correctly classified in the top position and an average of 96% for the word being correctly classified within the top five positions

[4] Handwriting Recognition using Deep Learning based Convolutional Neural Network- -Asha K, Krishnappa H K (2019)

Handwriting is a learned skill that had been an excellent means of communication and documentation for thousands of years. The simple way to communicate with the computers is through either speech or handwriting. Speech has some limitation; hence input through handwriting is recommended. It is difficult to input data for computers for Indian language scripts because of their complex character set. This paper focuses on exploring convolutional neural networks (CNN) which is deep learning based for the recognition of handwritten script. The proposed method has shown 99% for handwritten English numerals and promising recognition accuracy for Kannada numerals.

[5] Handwritten Text Recognition using Machine Learning Techniques in Application of NLP--Polaiah Bojja, Naga Sai Satya Teja Velpuri, Gautham Kumar Pandala, S D Lalitha Rao Sharma Polavarapu, Pamula Raja Kumari (2019)

Handwriting Detection is a technique or ability of a computer to receive and interpret intelligible handwritten input from source such as paper documents, touch screen, photographs etc. Handwritten Text recognition is one of are a pattern recognition. The purpose of pattern recognition is to categorizing or classification data or object of one of the classes or categories. Handwriting recognition is defined as the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. Each script has a set of icons, which are known as characters or letters, which have certain basic shapes. The goal of handwriting is to identify input characters or image correctly then analyzed to many automated process systems. This system will be applied to detect the writings of different format. The development of handwriting is more sophisticated, which is found various kinds of handwritten character such as digit, numeral, cursive script, symbols, and scripts including English and other languages. The automatic recognition of handwritten text can be extremely useful in many applications where it is necessary to process large volumes of handwritten data, such as recognition of addresses and postcodes on envelopes, interpretation of amounts on bank checks, document analysis, and verification of signatures. Therefore, computer is needed to be able to read document or data for ease of document processing.

**[6] Handwritten Recognition by using Machine Learning Approach P.Thangamariappan
Dr.J.C.Miraclin Joyce Pamila(2020)**

This paper presents the result of handwritten recognition using deep learning. Handwriting is unique to each individual. So the handwriting is differed from one person to another person. Handwritten Recognition can be done in two ways. One is Online Handwritten recognition and another one is Offline Handwritten Recognition. Online Handwritten recognition system, which takes the input at run time and Offline Handwritten Recognition which works on scanned images. Offline handwritten is the hardest to find the handwriting. MNIST data set is used for this handwritten digit recognition process and it has 70000 handwritten digits. Many Machine Learning Algorithms are developed which can be used for this digit classification. This paper performs the analysis of accuracies and performance measures of algorithm Convolutional Neural Networks (CNN). The proposed approach recognizes with overall accuracy is 93%.

**[7] Handwritten Text Recognition: with Deep Learning and Android-Shubham Sanjay
Mor, Shivam Solanki, Saransh Gupta, Sayam Dhingra, Monika Jain, Rahul Saxena
(2019)**

This research paper offers a new solution to traditional handwriting recognition techniques using concepts of Deep learning and computer vision. An extension of MNIST digits dataset called the Emnist dataset has been used. It contains 62 classes with 0-9 digits and A-Z characters in both uppercase and lowercase. An application for Android, to detect handwritten text and convert it into digital form using Convolutional Neural Networks, abbreviated as CNN, for text classification and detection, has been created. Prior to that we pre-processed the dataset and applied various filters over it. We designed an android application using Android Studio and linked our handwriting text recognition program using tensorflow libraries. The layout of the application has been kept simple for demonstration purpose. It uses a protobuf file and tensorflow interface to use the trained keras graph to predict alphanumeric characters drawn using a finger

**[8] Handwritten Text Recognition System Based on Neural Network-IAhmed Mahi
Obaid, IIAhmed M. El Bakry, IIIM.A. Eldosuky, IVA.I. Shehab (2016)**

This paper proposes an efficient approach towards the development of handwritten text recognition systems. 3-layer Artificial Neural Network (ANN) is utilized in this Paper using supervised learning approach. The choice of optimal feature vectors greatly the accuracy of any

text recognition system therefore bit map representation of input samples are utilized as feature vector. The feature vectors are first pre-processed and then applied to the ANN along with the generated target vectors; that are generated on the basis on input samples. 55 samples of each English alphabet are used as a ANN training process in order to make sure the general applicability of system towards new inputs. Two different learning algorithms are utilized in this paper. Additive image processing algorithms are also developed in order to deal with the multiple characters input in a single image, tilt image and rotated image. The trained system provides an average accuracy of more than 95 % with the unseen test image

[9] Handwriting Recognition using Artificial Intelligence Neural Network and Image

Processing-Sara Aqab¹, Muhammad Usman Tariq (2020)

This paper aims to report the development of a Handwriting character recognition system that will be used to read students and lectures Handwriting notes. The development is based on an artificial neural network, which is a field of study in artificial intelligence. Different techniques and methods are used to develop a Handwriting character recognition system. However, few of them focus on neural networks. The use of neural networks for recognizing Handwriting characters is more efficient and robust compared with other computing techniques. The paper also outlines the methodology, design, and architecture of the Handwriting character recognition system and testing and results of the system development. The aim is to demonstrate the effectiveness of neural networks for Handwriting character recognition.

[10] Handwritten Character Recognition using Deep Learning in Android Phones Athira

M Nair¹, Chrissie Aldo¹, Blessil Bose¹, Alex Joseph¹, Praseetha V.M², Sr. Elizabeth M J (2021)

There are many things that humans have in common, yet there are other things that are very unique to every individual and one of them is handwriting. Handwriting is a skill that is personal to individuals. It has continued as a means of communication and recording information in day-to-day life. Because each person's handwriting is unique, it is sometimes hard to interpret the information they try to convey. As computerization is becoming more prominent these days, Handwriting Recognition is gaining importance in various fields. The major focus is to understand the handwriting and convert it into readable text. Deep learning, an ability of Artificial Intelligence (AI), is used for the system to learn the input automatically and convert the handwritten text to printed text.

3. EXISTING SYSTEM

Handwritten character recognition (HCR) is the detection of characters from images, documents and other sources and changes them in machine-readable shape for further processing. The accurate recognition of intricate-shaped compound handwritten characters is still a great challenge. Recent advances in convolutional neural network (CNN) have made great progress in HCR by learning discriminatory characteristics from large amounts of raw data. In this paper, CNN is implemented to recognize the characters from a test dataset. The main focus of this work is to investigate CNN capability to recognize the characters from the image dataset and the accuracy of recognition with training and testing. CNN recognizes the characters by considering the forms and contrasting the features that differentiate among characters.

DRAWBACKS:

- Take more processing Time
- Less Accurate
- Less Training Examples

4. PROPOSED SYSTEM

4.1 PROBLEM STATEMENT

The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image.

4.2 SCOPE:

The scope of the project include :

- **Digit recognition:** CNN and RNN techniques can be used to recognize handwritten digits in various applications, such as postal services, bank cheque processing, and automated form filling.
- **Optical character recognition (OCR):** OCR technology is used to convert handwritten documents into digital format. CNN and RNN techniques can be used to improve the accuracy of OCR systems, making it easier to extract information from documents.
- **Signature verification:** CNN and RNN techniques can be used to verify signatures, such as those on bank cheques or legal documents. This can help prevent fraud and ensure the authenticity of documents.
- **Medical records:** Handwritten medical records can be difficult to read and interpret. CNN and RNN techniques can be used to automatically transcribe and digitize medical records, making it easier for healthcare professionals to access and analyze patient information.
- **Historical documents:** CNN and RNN techniques can be used to digitize and preserve historical documents that may be deteriorating or difficult to read due to age or damage. This can help make these documents more accessible to researchers and the general public.

4.3 OBJECTIVE

Handwritten Text recognition is one of the areas of pattern recognition. The purpose of pattern recognition is to categorize or classify data or object of one of the classes or categories. Handwriting recognition is defined as the task of transforming a language represented in its

spatial form of graphical marks into its symbolic representation. Each script has a set of icons, which are known as characters or letters, which have certain basic shapes. The goal of handwriting is to identify input characters or image correctly then analyzed to many automated process systems. This system will be applied to detect the writings of different format. The development of handwriting is more sophisticated, which is found various kinds of handwritten character such as digit, numeral, cursive script, symbols, and scripts including English and other languages. The automatic recognition of handwritten text can be extremely useful in many applications where it is necessary to process large volumes of handwritten data, such as recognition of addresses and postcodes on envelopes, interpretation of amounts on bank checks, document analysis, and verification of signatures. Therefore, computer is needed to be able to read document or data for ease of document processing.

4.4 PROPOSED SYSTEM

Handwritten Text Recognition (HTR) systems consist of handwritten text in the form of scanned images as shown in figure 1. We are going to build a Neural Network (NN) which is trained on word-images from the IAM dataset. Because the input layer (and therefore also all the opposite layers) are often kept small for word-images, NN training is possible on the CPU (of course, a GPU would be better). For the implementation of HTR, the minimum requirement is TF.



Fig 4.1: Image of word taken from IAM Dataset **Advantages:**

- Less process time
- Better accuracy
- Easy to use

The block diagram of the proposed E handwritten character classification is shown in Figure. The proposed recognition technique relies on a convolutional neural network model (CNN) with a feature mapped output layer. Our proposed model will classify the given input out of 10 classes using CNN while classifying the Urdu numeral. Similarly, while classifying the Urdu

character, the same model will classify the given character out of 12 classes (see Figure 2). The detail about the different phases of our proposed model will come in the following subsections.

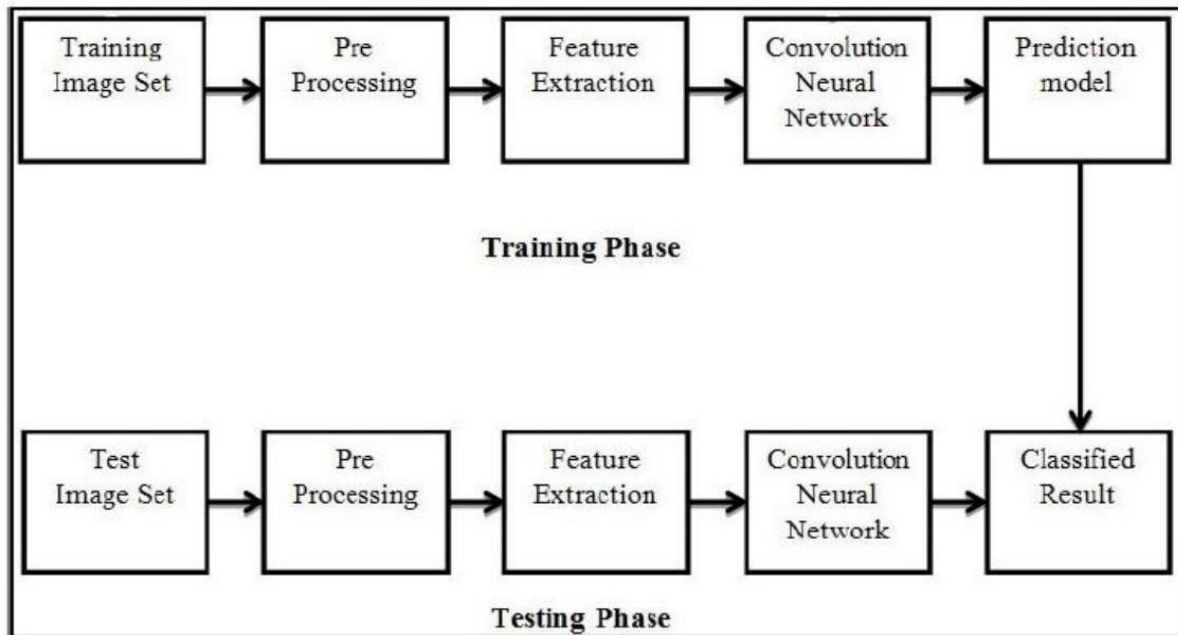


Fig4.2: Block diagram of proposed handwritten character classification system

5. REQUIREMENT ANALYSIS

5.1 INTRODUCTION

The process of requirement analysis involves the precise identification, definition, and documentation of various requirements that pertain to a specific business objective. Gathering requirements helps to gain a clear understanding of the customer's needs, define the project's scope, and determine the necessary resources required to achieve the objectives.

5.2 FUNCTIONAL REQUIREMENTS

the needs of its users. These requirements typically detail how the system or product should behave under various circumstances, what inputs and outputs are required, and what performance expectations must be met.

The system should capture input images and provide resulting output The system should be able to identify cursive letters, symbols and digits.

5.3 NON-FUNCTIONAL REQUIREMENTS

- The system should have better accuracy in recognizing the words.
- The system should have fast processing speed and should be able to process input data with better accuracy.

5.4 TECHNICAL REQUIREMENTS

The technical requirements for this project are mentioned below.

5.4.1 Minimum System Requirements:

A. For End User

5.4.1.1 Hardware

- I3 Processor Based Computer or higher
- Memory: 4 GB RAM
- Hard Drive: 50 GB
- Monitor
- Internet Connection

5.4.1.2 Software

- Windows 7 or Higher
- Any Browser

B. For Development & Training

5.4.1.3 Hardware

- I3 Processor Based Computer or higher
- Memory: 4 GB RAM
- Hard Drive: 50 GB
- Monitor
- Internet Connection

5.4.1.4 Software

- Windows 10 or Higher
- Any Browser for Testing
- Python dependencies
- Deep Learning libraries

5.4.2 Development Technologies:

- **Front End:** Flask, HTML, CSS

For building the UI and UX for the app/website that one could use while using the service.

- **Back End:** Python, Deployment Services, Deep Learning Libraries.

We built and trained the model by using Python and then trained and tested the model to hyper-tune the parameters and achieve high level of accuracy.

6.DESIGN AND METHODOLOGY

6.1 PROCESS MODEL:

Process Model are processes of the same nature that are classified together into a model. Thus, a process model is a description of a process at the type of level. Since the process model is at the type of level, a process is an instantiation of it. The same process model is used repeatedly for the development of many applications and thus, has many instantiations. One possible use of a process model is to prescribe how things must/should/could be done in contrast to the process itself which is really what happens. A process model is roughly anticipation of what the process will look like. What the process shall be determined during actual system development. The goal of a process model is to be:

1. Descriptive: • Track what happens during a process.
 - Take the point of view of an external observer who looks at the way a process has been performed and determines the improvements that must be made to make it perform more effectively or efficiently.
2. Prescriptive: • Define the desired processes and how they should/could/might be performed.
 - Establish rules, guidelines, and behavior patterns which, if followed, would lead to the desired process performance. They can range from strict enforcement to flexible guidance.
3. Explanatory:
 - Provide explanations about the rationale of processes.
 - Explore and evaluate the several possible courses of action based on rational arguments.
 - Establish an explicit link between processes and the requirements that the model needs to fulfil.
4. Pre-defines points at which data can be extracted for reporting purposes.

6.1.1 Breakdown Process:

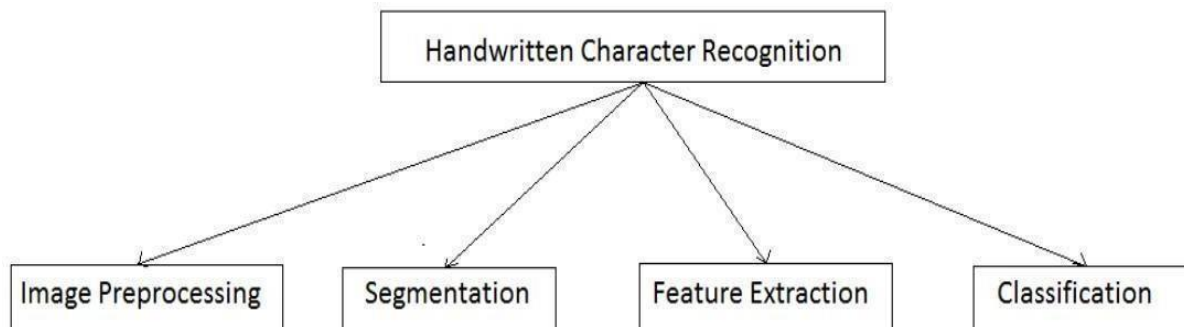


Fig 6.1 Breakdown Model 1.

Image Preprocessing

2. Segmentation

3. Feature Extraction

4. Classification

1. Image Pre processing

The image is preprocessed using different image processing algorithms like Inverting image, Gray Scale Conversion and image thinning.

2. Segmentation

After preprocessing of the image segmentation is done. This is done with the help of following steps:

1. Remove the borders
2. Divide the text into rows
3. Divide the rows(lines) into Word
4. Divide the word into letters

3. Feature Extraction

Once the character is segmented, we generate the binary glyphs and calculate the summation of each row and columns values as a feature.

4. Classification

In this phase, we are going to train and test the Neural Network, to finally identify the answer.

6.1.2 System Architecture:

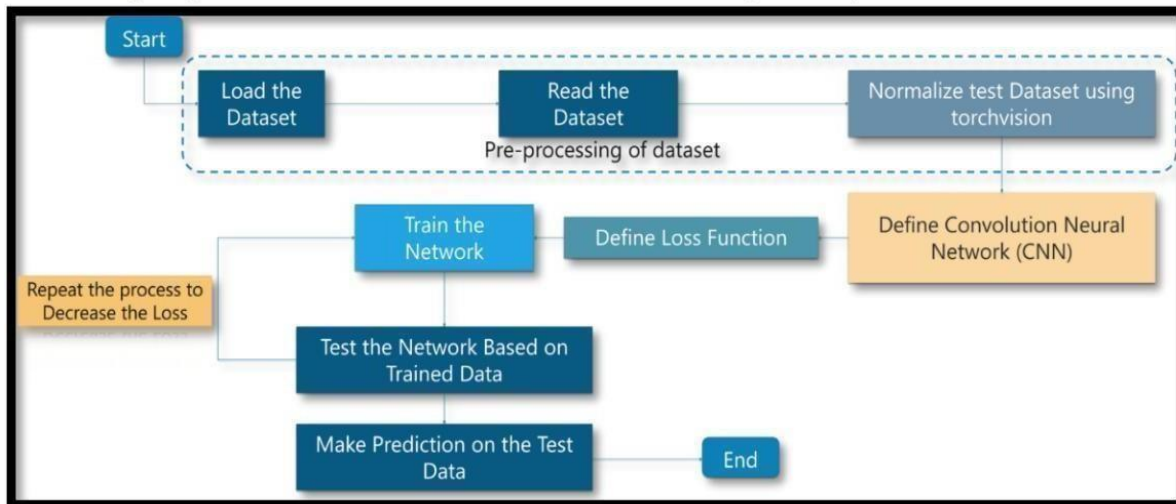


Fig 6.2: Process Flow Diagram for Model Creation

The Model Architecture proposed involves pre-processing the data and then feeding it into the CNN, after which it will be fed into the RNN network which will then finalize and give us our output.

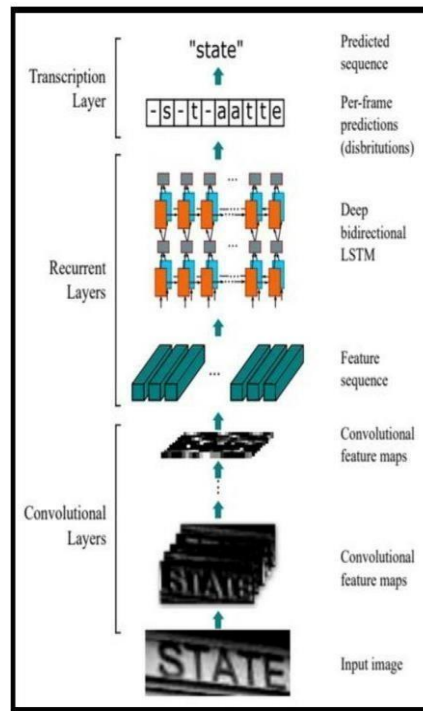


Fig 6.3: Proposed Model Architecture Training the Model:

- The relevant dataset chosen by us, MNIST dataset will be loaded and a subset of it will be used, after dividing it into training and testing set.
- Firstly, we are looking to develop a model architecture with a mixture of CNN, RNN and other desired layers, using DL libraries of Python.
- Then we are looking to first perform the pre-processing operations required to be done on the dataset, to feed the data then to the Training Mode.
- After the Model will be put under training, and we will train multiple models by changing its parameters just so slightly for us to have multiple models to test upon.

Testing the Model:

- The models trained in the earlier steps will not be tested on the data that it had not seen before.
- For, different models we will be getting different accuracies and then we will be able to have the best accurate model.

6.1.3 System Deployed:

In the system we have deployed, after training and saving the model, we deployed the model in the form of an API, or by using any kind of deploying service into the web. Then we created an interface with the model where the user will be able to access it and then use it accordingly. We developed a website kind of an interface at first where as per the user's requirement the model will be called to perform and identify the characters.

Layer(Type)	Configuration
Input	800x64 gray-scale image
Convolution	maps:64, k:5x5, s:1,p:same
Maxpooling	Window: 2x2, s:2
Convolution	Maps:128, k:5x5, s:1, p:same
Convolution	Maps:128, k:3x3, s:1, p:same
BatchNormalization	-
Maxpooling	Window:2x2, s:2
Convolution	maps:256, k:3x3, s:1, p:same
Convolution	maps:256, k:3x3, s:1, p:same
Convolution	maps:512, k:3x3, s:1, p:same
BatchNormalization	-
Convolution	maps:512, k:3x3, s:1, p:same
Maxpooling	Window:2x2, s:2
Map-to-Sequence	-
LSTM1	#hidden units:512
LSTM2	#hidden units:512
Concatenate	LSTM1, LSTM2
Decoding	-

Fig6.4 Model Architecture

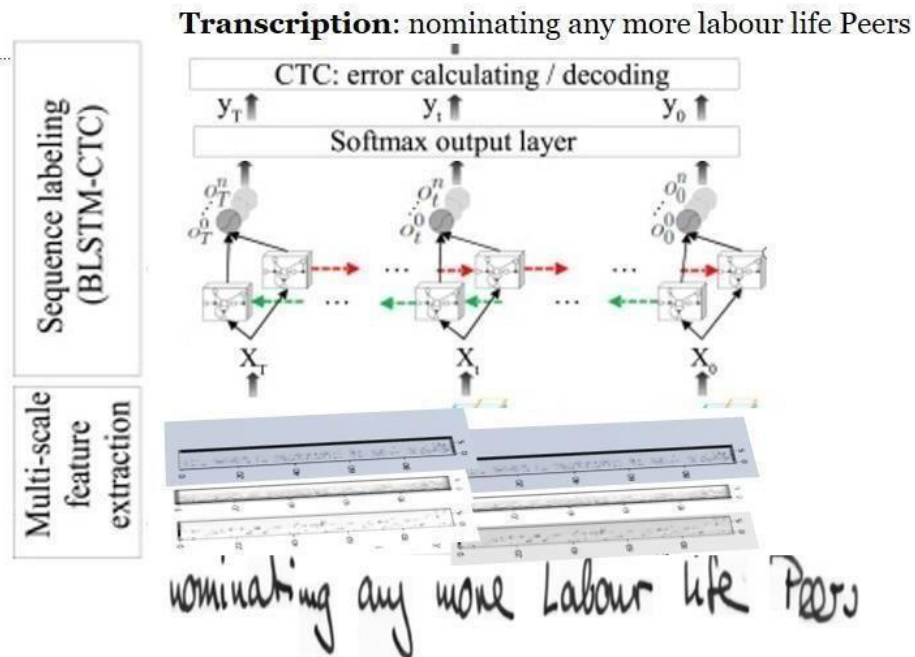


Fig 6.6: Sample Proposed Performance

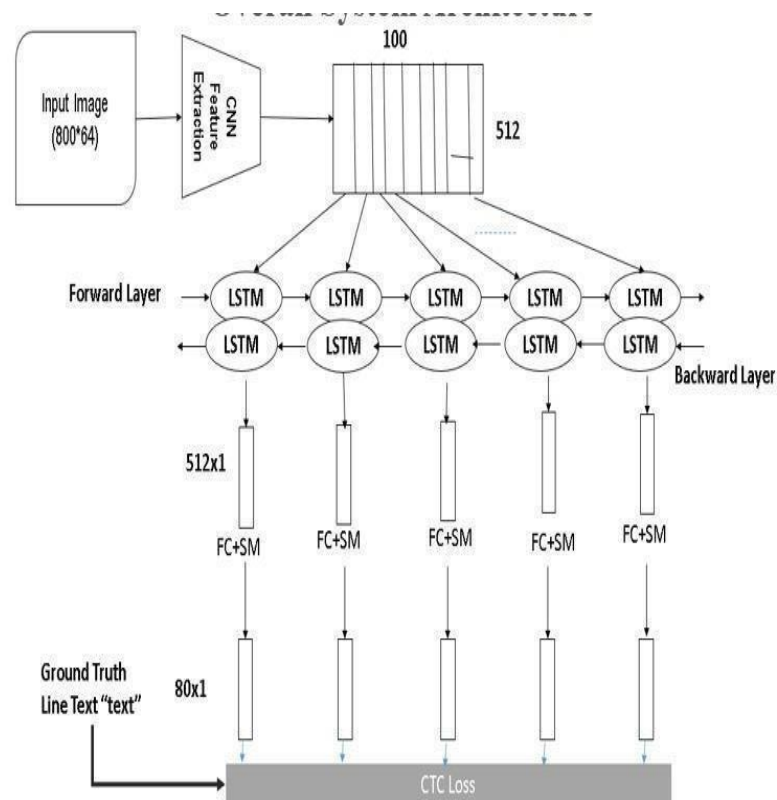


Fig 6.7: LSTM Architecture

6.2 UML DIAGRAMS:

The unified modelling language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows

6.2.1 Control Flow Diagram:

The large class of applications having following characteristics requires control flow modelling:

- The applications that are driven by events rather than data.
- The applications that produce control flow information rather than reports or displays.
- The application that process information in specific time.

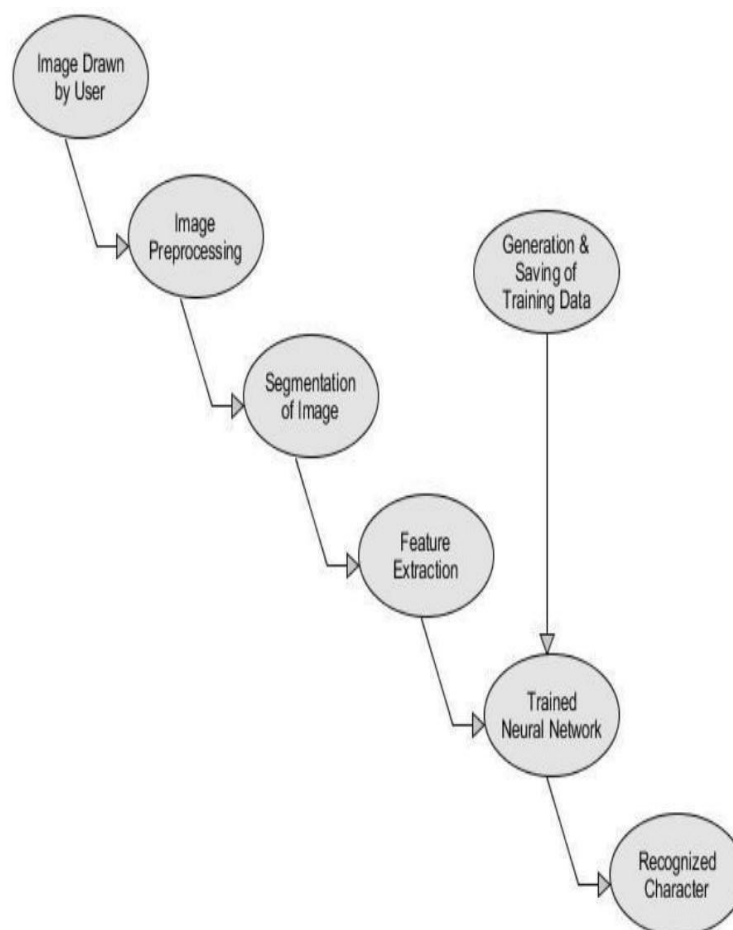


Fig 6.8: Control Flow Diagram for HCR System

6.2.2 Deployment Diagram:

A deployment diagram shows the allocation of processes to processors in the physical design of a system. A deployment diagram may represent all or part of the process architecture of a system.

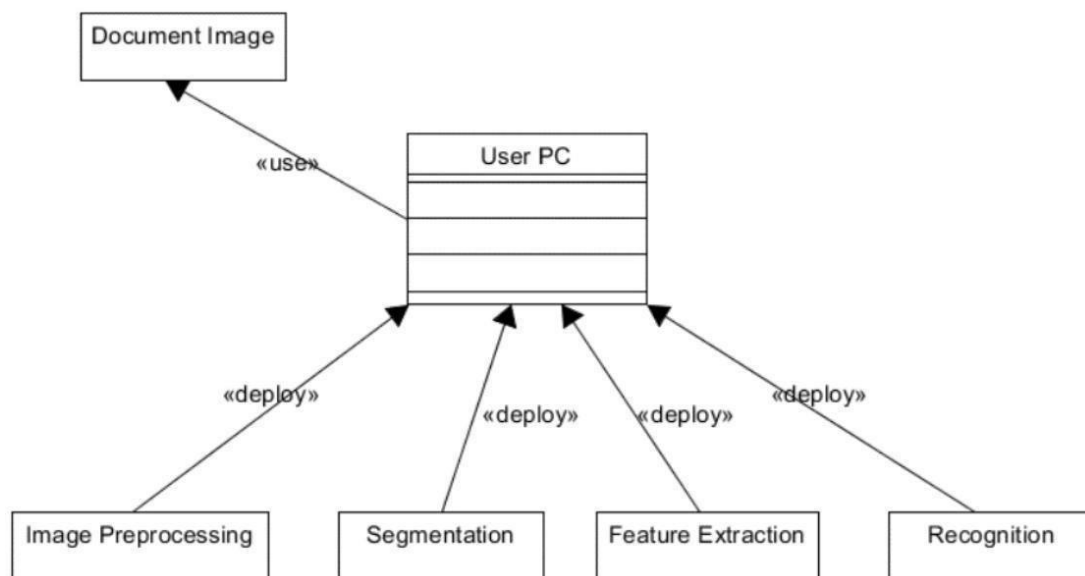


Fig 6.9 Deployment Diagram

6.2.3 Class Diagram:

The class diagram is the main building block of object-oriented modelling. It is used for general conceptual modelling of the structure of the application, and for detailed modelling, translating the models into programming code. Class diagrams can also be used for data modelling.

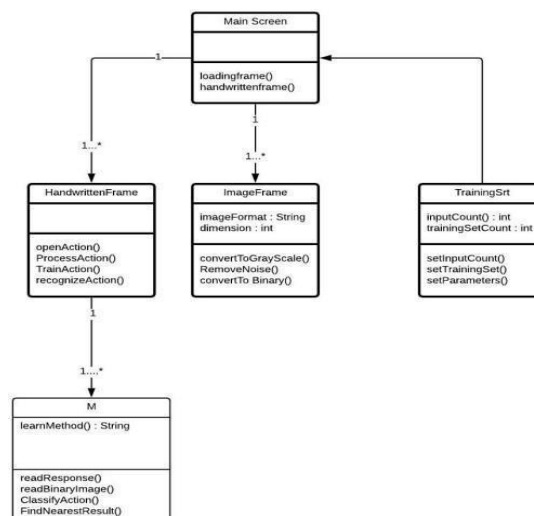


Fig 6.10 Class Diagram

7.IMPLEMENTATION

7.1 MODULES:

1. Image Pre processing
2. Segmentation
- 3.Feature Extraction
- 4.Classification

1. Image Pre processing

The image is pre processed using different image processing algorithms like Inverting image, Gray Scale Conversion and image thinning.

2. Segmentation

After pre processing of the image segmentation is done. This is done with the help of following steps: 1.Remove the borders 2.Divide the text into rows

3.Divide the rows (lines) into words

4.Divide the word into letters

3. Feature Extraction

Once the character is segmented we generate the binary glyphs and calculate the summation of each rows and columns values as an features.

4. Classification

In this phase, we are going to train and test the Neural Network, to finally identify the answer.

7.2 TECHNOLOGIES DESCRIPTION:

Introduction to Deep Learning:

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behaviour of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well.

Deep learning neural networks, or artificial neural networks, attempt to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.

Deep learning methods:

Various methods can be used to create strong deep learning models. These techniques include learning rate decay, transfer learning, training from scratch and dropout.

Learning rate decay. The learning rate is a hyperparameter a factor that defines the system or set conditions for its operation prior to the learning process that controls how much change the model experiences in response to the estimated error every time the model weights are altered. Learning rates that are too high may result in unstable training processes or the learning of a suboptimal set of weights. Learning rates that are too small may produce a lengthy training process that has the potential to get stuck.

Transfer learning. This process involves perfecting a previously trained model; it requires an interface to the internals of a pre-existing network. First, users feed the existing network new data containing previously unknown classifications. Once adjustments are made to the network, new tasks can be performed with more specific categorizing abilities. This method has the

advantage of requiring much less data than others, thus reducing computation time to minutes or hours.

Training from scratch. This method requires a developer to collect a large labelled data set and configure a network architecture that can learn the features and model. This technique is especially useful for new applications, as well as applications with a large number of output categories. However, overall, it is a less common approach, as it requires inordinate amounts of data, causing training to take days or weeks.

Dropout. This method attempts to solve the problem of overfitting in networks with large amounts of parameters by randomly dropping units and their connections from the neural network during training. It has been proven that the dropout method can improve the performance of neural networks on supervised learning tasks in areas such as speech recognition, document classification and computational biology.

7.3 LIBRARIES:

PyTesseract: PyTesseract is a Python wrapper for Google's Tesseract-OCR Engine. Tesseract is an open-source OCR (Optical Character Recognition) engine maintained by Google and it's widely used for recognizing text from images. PyTesseract makes it easy to use Tesseract in Python applications.

Gradio: Gradio is a Python library that allows you to create simple UIs for your machine learning models with just a few lines of code. It's particularly useful for quickly prototyping and sharing your models with others, as it provides an interface for users to interact with your model without needing to write any code themselves. With Gradio, you can create interfaces for various types of inputs (text, images, audio, etc.) and outputs (classification, regression, generation, etc.), making it versatile for a wide range of applications.

Os: The os module in Python provides a way of interacting with the operating system. It offers functions for accessing filesystem paths, manipulating environment variables, executing shell commands, and more. Here's a brief overview of some common tasks you can accomplish using the os module.

7.4 INTRODUCTION TO PYTHON:

Python:

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type

relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Drop box, Uber... etc

Advantages of Python:

Extensive Libraries:

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases , CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

1. Extensible:

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

2. Embeddable:

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

3. Improved Productivity:

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

4. IOT Opportunities:

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

5. Simple and Easy:

When working with Java, you may have to create a class to print „Hello World“. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python; they have a hard time adjusting to other more verbose languages like Java.

6. Readable:

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory.

7. Object-Oriented:

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

8. Free and Open-Source:

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It comes with an extensive collection of libraries to help you with your tasks.

9. Portable:

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it is not the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

8.MODULE DIVISION

8.1 Modules

The proposed System Contains the following modules:

- Data Loader
- Main
- Model
- Sample pre processor
- Spellchecker
- Upload

8.1.1 Data Loader

This is a Python code that loads and preprocesses data for a handwriting recognition model. The data is in the IAM format, and this code extracts the images and ground truth texts from the dataset. It also splits the data into a training and validation set and shuffles the samples in each epoch.

The File Paths class stores the paths to the different files required for the model training. The Sample class represents an image and its corresponding ground truth text, while the Batch class represents a batch of images and their corresponding ground truth texts.

The Data Loader class loads and preprocesses the data.

8.1.2 main

This Model performs Optical Character Recognition (OCR) on an image. It has three main functions: train, validate, and infer.

The train function trains the OCR model using the Data Loader and Model classes. It trains the model in batches and validates the model after each epoch. It saves the best model parameters based on validation character error rate. The validation function evaluates the model on a validation set and computes the character error rate (CER), address accuracy, and word error rate (WER). The infer function takes a file path to an image and performs OCR on that image using the trained model. There are also two additional functions: `load_different_image` and `generate_random_images`. The `load_different_image` function loads different images of the same

character to test the robustness of the model. The `generate_random_images` function generates random images to test the model's ability to handle noise.

8.1.3 Model:

This code defines a class `Model` with methods to setup a Convolutional Neural Network (CNN), a Recurrent Neural Network (RNN) and Connectionist Temporal Classification (CTC) to perform Optical Character Recognition (OCR) on images of text.

The CNN has six layers with various filters and pooling operations. The RNN is a Long Short-Term Memory (LSTM) network with 256 units. The CTC is used to calculate the loss between the predicted text and the ground truth text.

8.1.4 Sample preprocessor:

This code defines a function that implements the Wagner-Fischer algorithm to calculate the Levenshtein distance between two strings `r` and `h`. The Levenshtein distance is the minimum number of single-character insertions, deletions, or substitutions required to transform one string into another.

8.1.5 Spell checker:

The `correct_sentence` function takes a string of text as input and returns a new string with each word in the original text corrected for spelling using the `autocorrect` library. The function first splits the input text into a list of individual words using the `split()` method with a space as the delimiter. Then it uses a for loop to iterate over each word in the list, applies the `spell()` function from `autocorrect` to correct the spelling of the word, and concatenates the corrected word to a new string with a space separator. Finally, the function returns the new string with corrected spelling.

8.1.6 Upload:

This is a Python Flask web application that provides a user interface for image classification using a pre-trained model. The web application allows the user to upload an image and select a prediction option. The uploaded image is saved to a specified directory, and the specified prediction option is passed to a `predict_image` function which uses a pre-trained model to generate a prediction result for the uploaded image. The prediction result is then displayed on the web page. The `app.run` line at the end of the code starts the Flask application on the local host with port 5000 and enables debugging mode.

9.SAMPLE CODE

Step 1: Install Tesseract-OCR and pytesseract

```
!sudo apt install tesseract-ocr -y
```

```
!pip install pytesseract
```

```
!pip install gradio
```

Step 2: Import necessary libraries

```
import pytesseract
```

```
from PIL import Image
```

```
import os
```

```
import gradio as gr
```

Step 3: Define the function to extract text from images

```
def extract_text(image):
```

```
    try:
```

```
        text = pytesseract.image_to_string(image)
```

```
        return text
```

```
    except Exception as e:
```

```
        print(f"Error processing image: {e}")
```

```
    return ""
```

Step 4: Define Gradio interface

```
inputs = gr.Image(type="pil")
```

```
outputs = gr.Textbox()
```

```
gr.Interface(fn=extract_text,inputs=inputs, outputs=outputs, title="Image Text  
Extraction").launch()
```

10.TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail unacceptably. There are various types of tests. Each test type addresses a specific testing requirement.

10.1Types of Tests:

10.1.1Unit Testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit, before integration. This is structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

10.1.2 Integration Testing:

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfied, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

10.1.3 Functional Testing:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centred on the following items: Valid Input: identified classes of valid input must be accepted. 43 Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised. Output: identified classes of application outputs must be executed.

10.1.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. 10.1.5 Whitebox Testing White Box Testing is testing in which the software tester knows the inner workings, structure and language of the software, or at least its purpose. It is a purpose. It is used to test areas that cannot be reached from a black-box level.

Testing the model

Validation character error rate of saved model: 8.654728%

```

# Step 1: Install Tesseract-OCR and pytesseract
!sudo apt install tesseract-ocr -y
!pip install pytesseract
!pip install gradio

# Step 2: Import necessary libraries
import pytesseract
from PIL import Image
import os
import gradio as gr

# Step 3: Define the function to extract text from images
def extract_text(image):
    try:
        text = pytesseract.image_to_string(image)
        return text
    except Exception as e:
        print(f"Error processing image: {e}")
        return ""

# Step 4: Define Gradio interface
inputs = gr.Image(type="pil")
outputs = gr.Textbox()
gr.Interface(fn=extract_text, inputs=inputs, outputs=outputs, title="Image Text Extraction").launch()

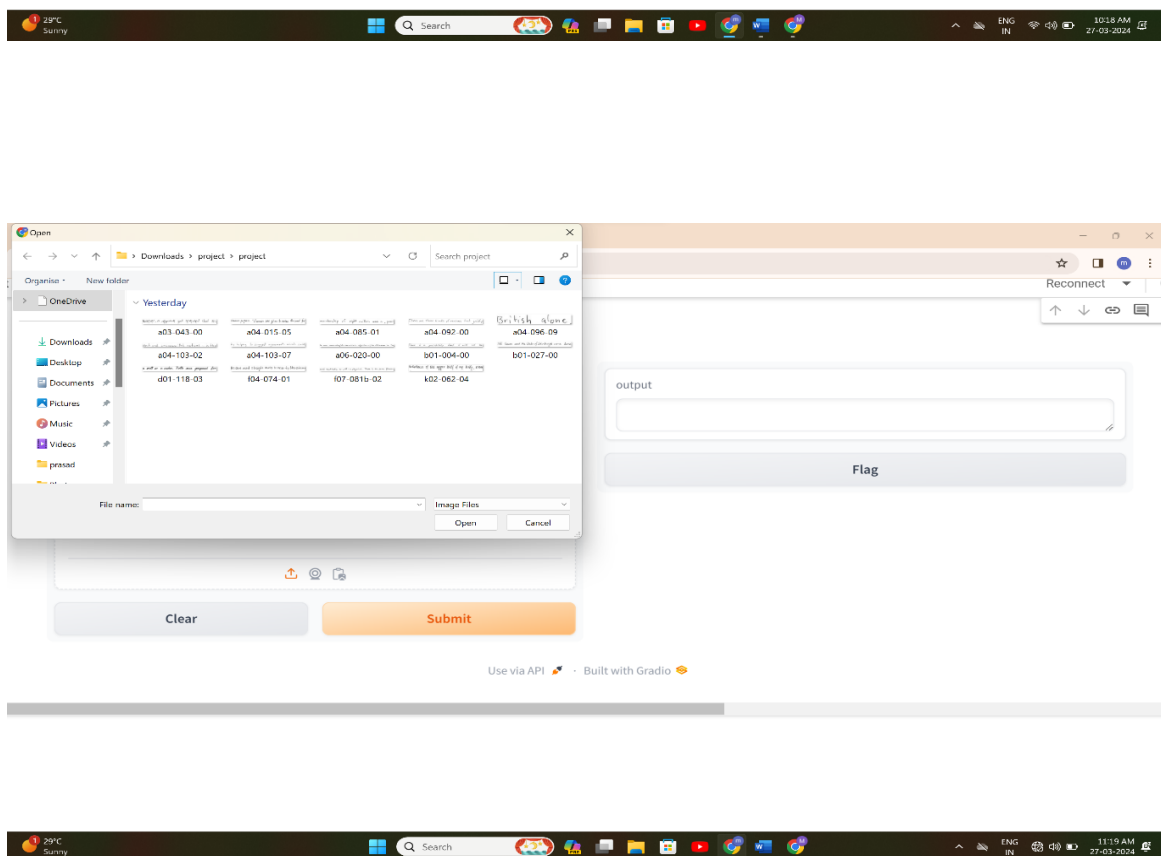
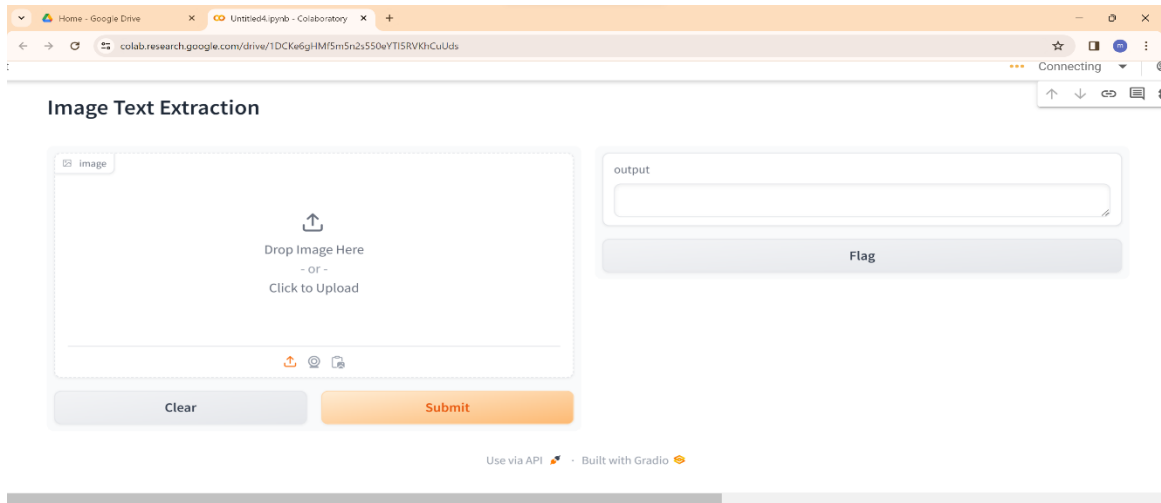
package lists... Done
dependency tree... Done
site information... Done
ing additional packages will be installed:
:-ocr-eng tesseract-ocr-osd
ing NEW packages will be installed:
:-ocr tesseract-ocr-eng tesseract-ocr-osd
. 3 newly installed, 0 to remove and 39 not upgraded.
: 4,816 kB of archives.
operation, 15.6 MB of additional disk space will be used.
//archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr-eng all 1:4.00-gtk30-7274cfa-1.1 [1,591 kB]
//archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr-osd all 1:4.00-gtk30-7274cfa-1.1 [2,990 kB]
//archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr amd64 4.1.1-2.1build1 [236 kB]
  
```

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	Line Image1	Text Recognition	and assthetbic as well as physical. There S the Same finesse	And asthetic as well as physical. There is the same finesse
T02	Line Image2	Text Recognition	membership of eight was\lion WUsc a y Roe	Membership of eight millon was a „ poor
T03	Line Image4	Text Recognition	British aton ec	British alone
T04	Line Image4	Text Recognition	Paracetamol S00mg/tab #5	Paracetamol 500mg/tab #5
T05	Line Image5	Text Recognition	CWLY purposes, ”occationu way give tb fevien) the ustel py	Common purposes, ”occation was given to review the need for”
T06	Line Image6	Text Recognition	However ,a-apparak pell revealed thar ur	However ,a separate pdl revealed that ur
T07	Line Image7	Text Recognition	J here ave three kinds of veEUsons that stiff	There are three kinds of reasons that justify
T08	Line Image8	Text Recognition	West and uncommi tea nations – is the	West and uncommitted nations - is the
T09	Line Image9	Text Recognition	a woll or a codex. Kolhy pee plepared for	a roll or a codex. Rolls were prepared for
T10	Line Image10	Text Recognition	THE Queen and the Duke of colunbusgh come Aome	THE Queen and the Duke of Edinburgh come home

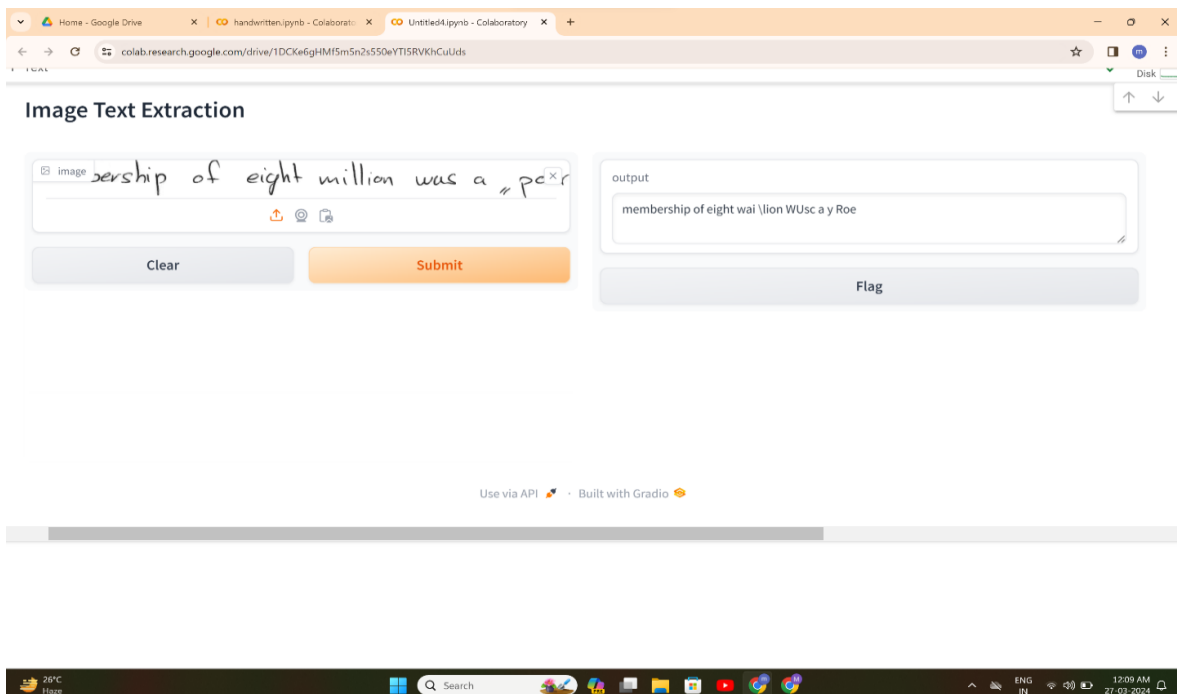
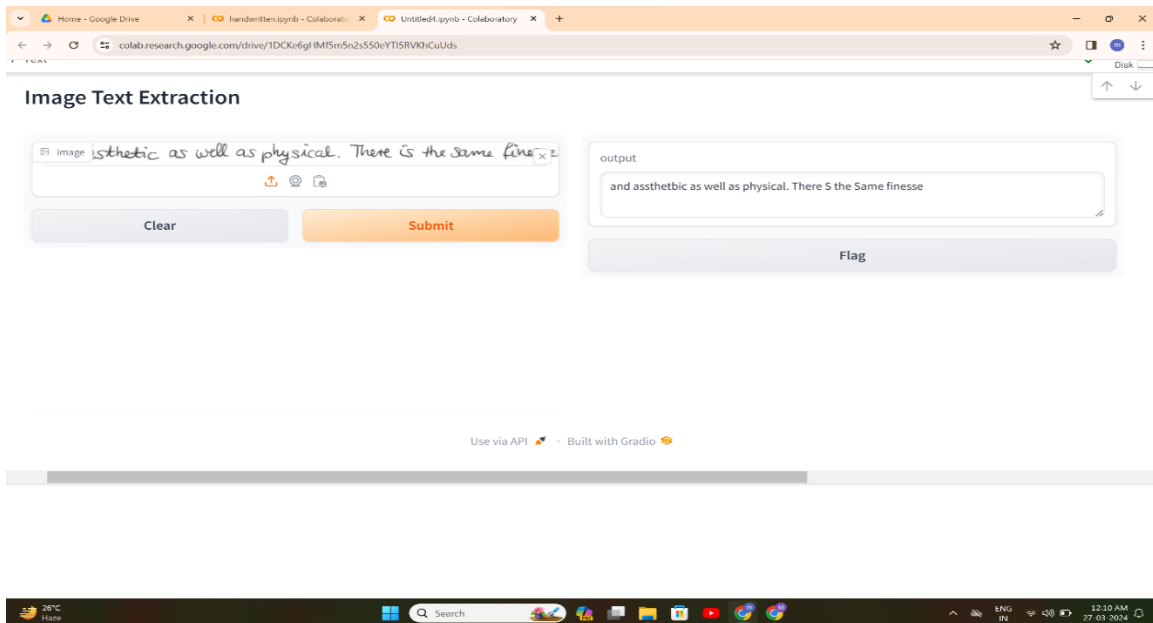
11.RESULTS & DISCUSSION

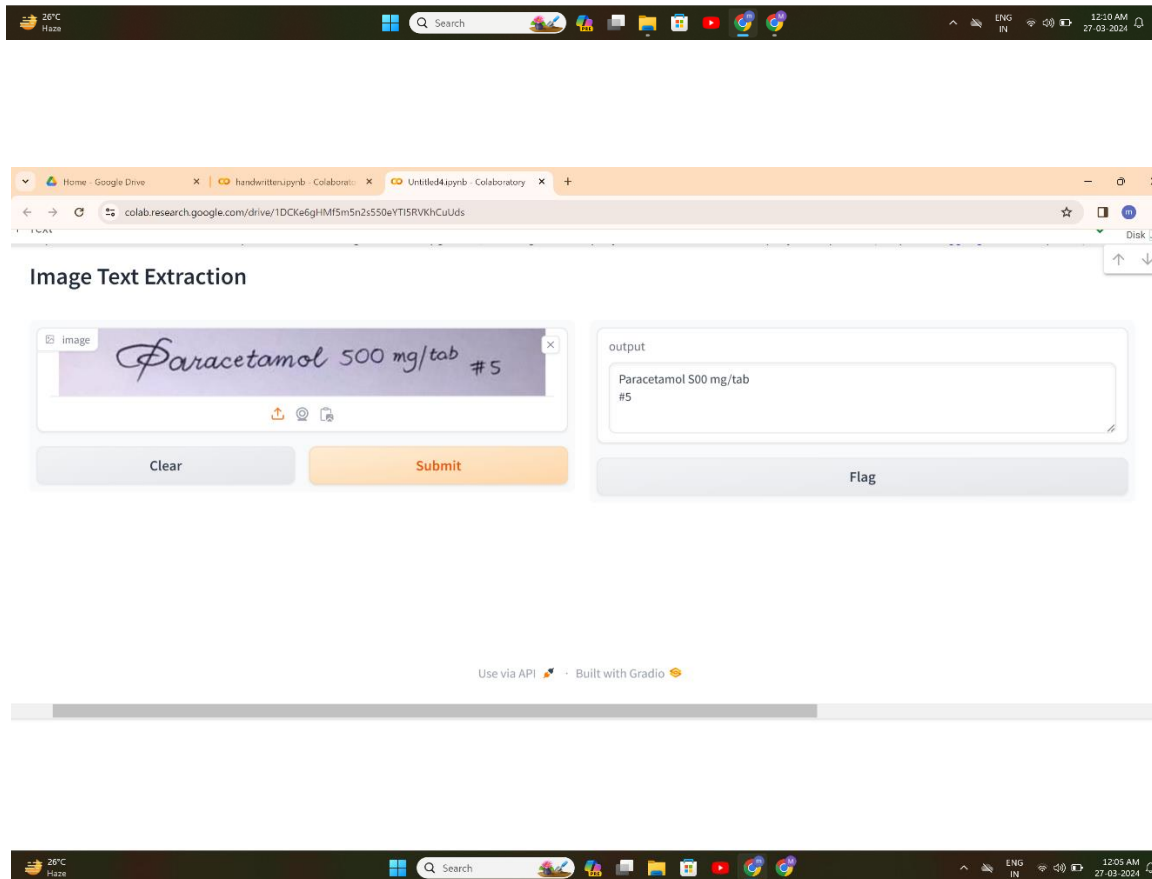
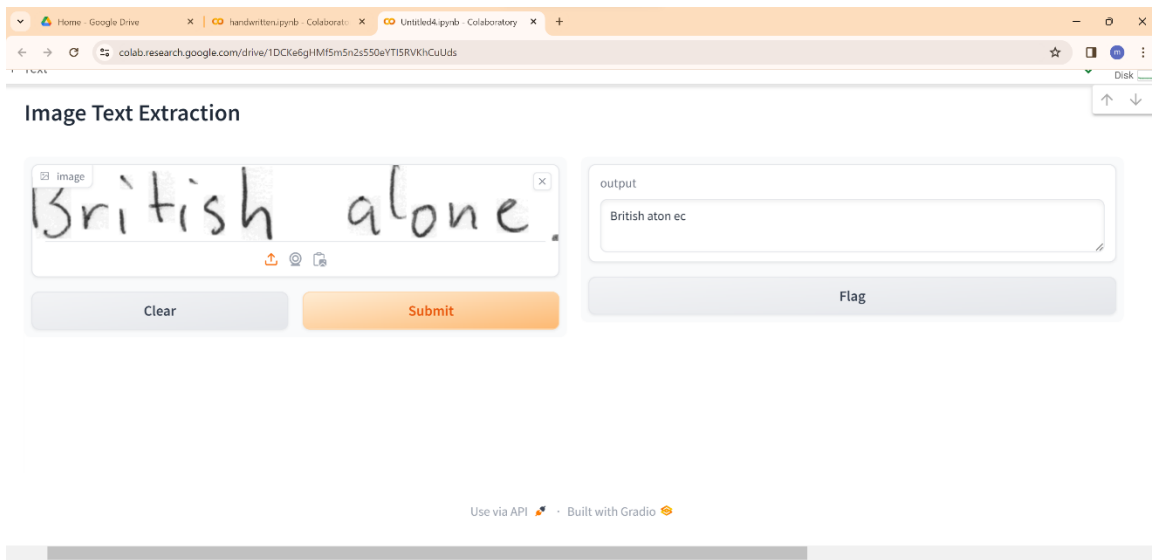
USER INTERFACE AND OUTPUT SCREENS

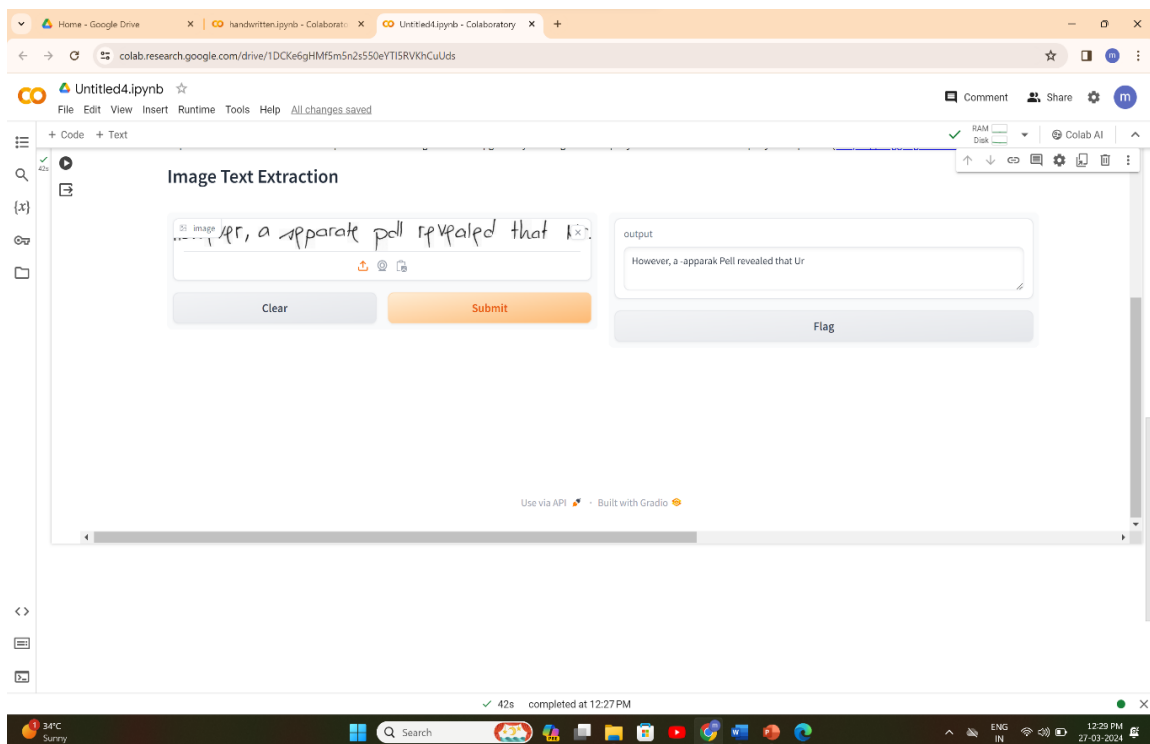
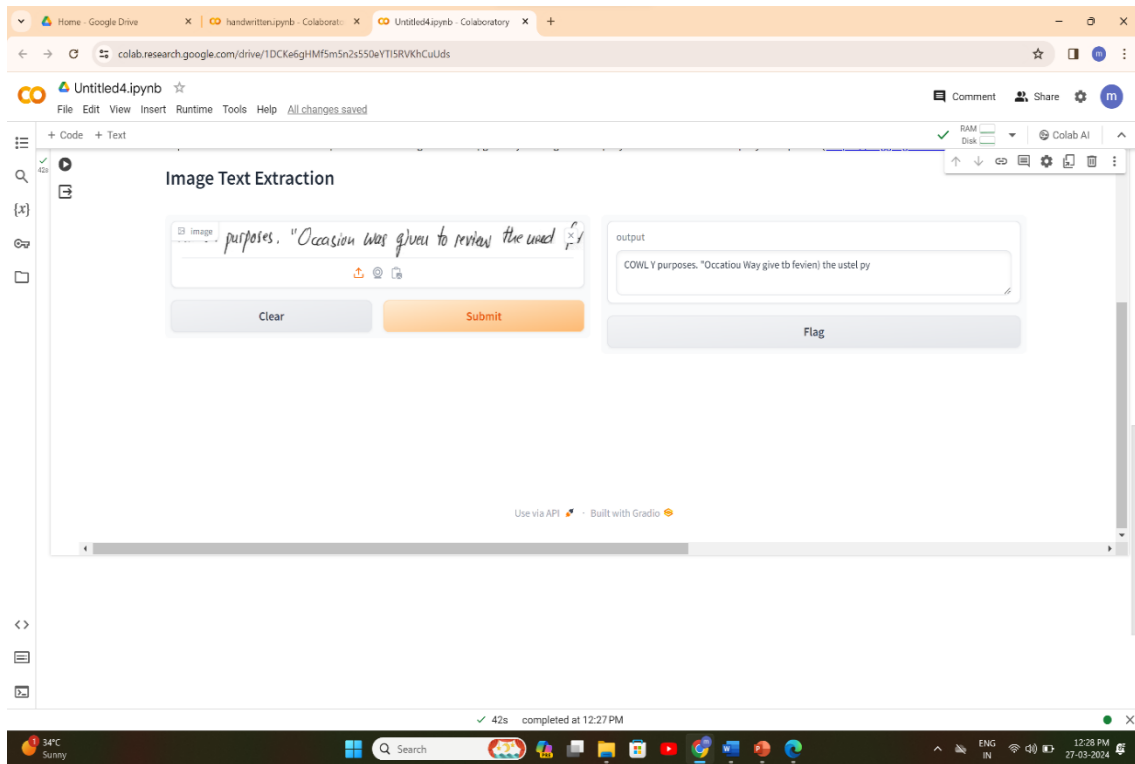
11.1.1 Input screens

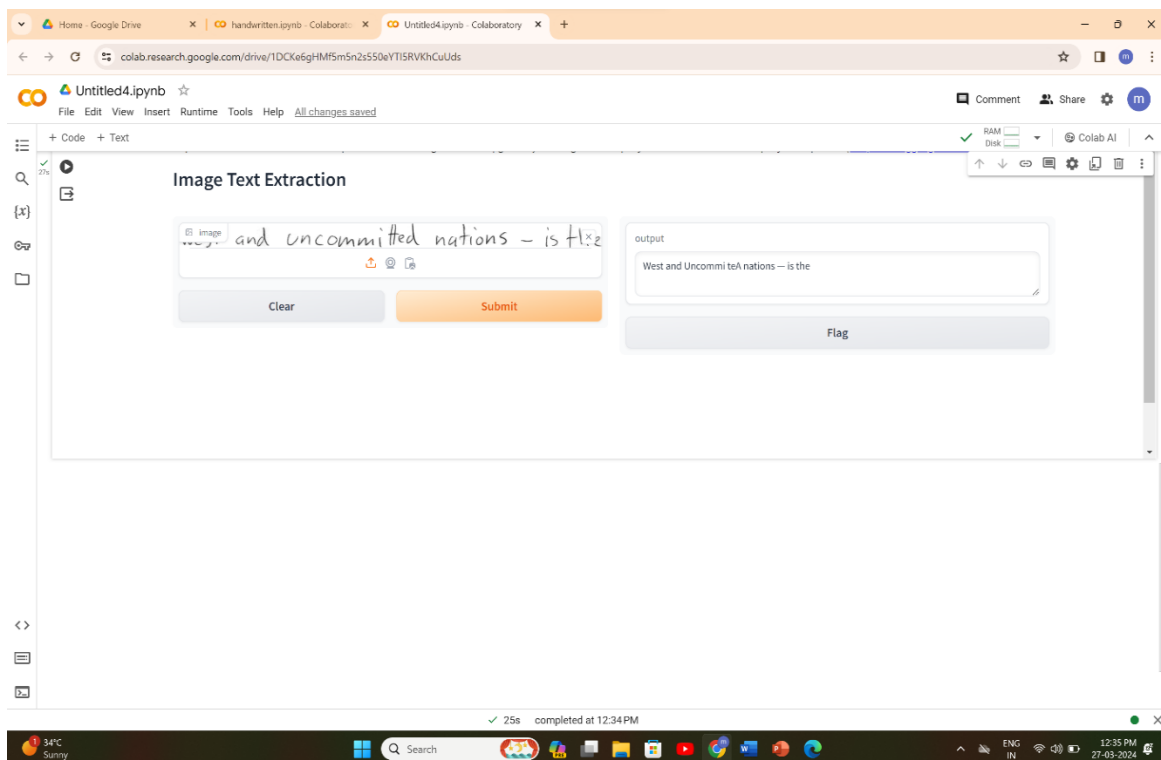
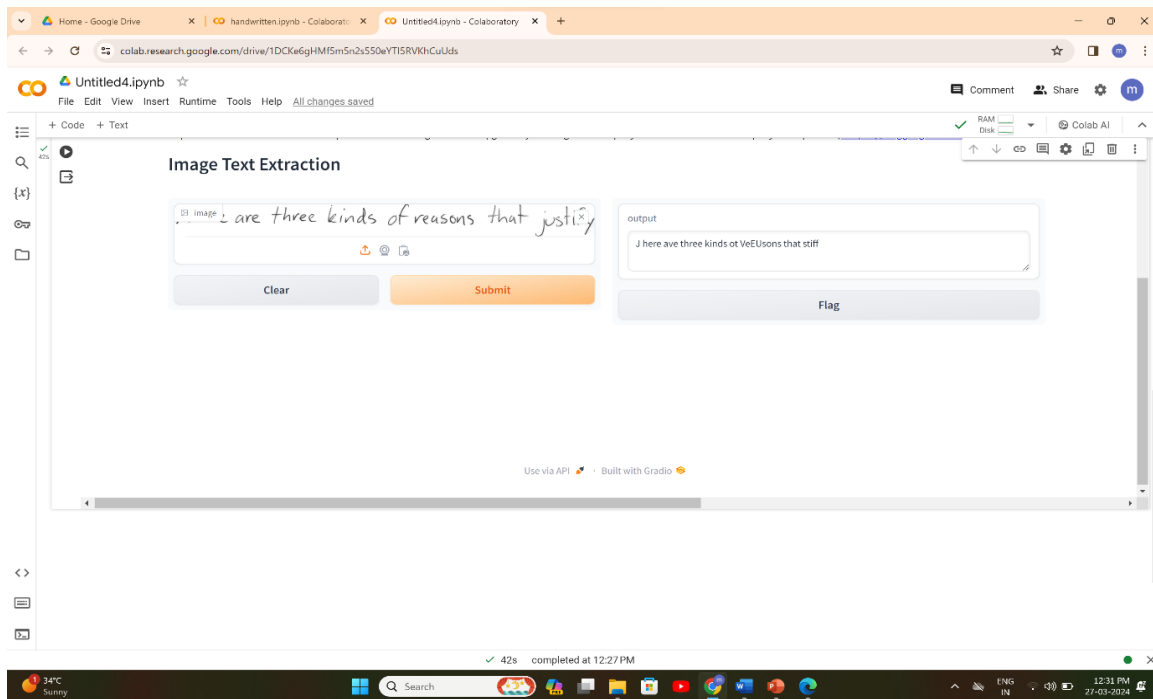


11.1.2 Output Screens:





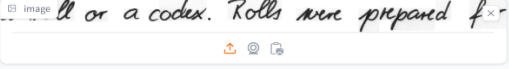




Home - Google Drive x Untitled4.ipynb - Colaboratory x +

colab.research.google.com/drive/1DCke6gHM5m5n2s550eYTISRvKhCuUds

Image Text Extraction

image 

output

a woll or a codex. Kolhy pee plepared for

Clear Submit Flag

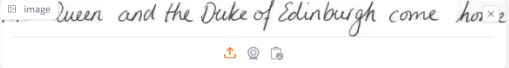
Use via API · Built with Gradio

35°C Sunny

Home - Google Drive x Untitled4.ipynb - Colaboratory x +

colab.research.google.com/drive/1DCke6gHM5m5n2s550eYTISRvKhCuUds

Image Text Extraction

image 

output

THE Queen and the Duke of colunbugsh come Aome

Clear Submit Flag

Use via API · Built with Gradio

Today's high
Near record

Home - Google Drive x Untitled4.ipynb - Colaboratory x +

colab.research.google.com/drive/1DCke6gHM5m5n2s550eYTISRvKhCuUds

Image Text Extraction

image 

output

THE Queen and the Duke of colunbugsh come Aome

Clear Submit Flag

Use via API · Built with Gradio

12.CONCLUSION&FUTURE SCOPE

12.1CONCLUSION:

Using modern day techniques like neural networks to implement deep learning to solve basic tasks which are done with a blink of an eye by any human like text recognition is just scratching the surface of the potential behind machine learning. There are infinite possibilities and application of this technology. Traditional OCR used to work similar to biometric device. Photo sensor technology was used to gather the match points of physical attributes and then convert it into database of known types. But with the help of modern-day techniques like convolution neural networks we are able to scan and understand words with an accuracy never seen before in history.

We are able to achieve good level of accuracy for English text. With proper training and further improvements, we could even make our model accurate to those of Highest Industry standards and likes of Google and Amazon.

12.2FEATURE SCOPE:

We could work on the model to make our model more diverse and expand its application to include other languages like Hindi, Marathi and other regional languages we either need to find good quality datasets that are very well labelled and that could be used directly, or we need to devise a way to train our model without having labelled dataset. We could work on improving the efficiency and accuracy by training the model more on the dataset and by using some more Deep Learning Algorithms.

13. REFERENCES

- [1] "Normalization Ensemble for Handwritten Character Recognition." -Cheng-Lin Liu and Katsumi Marukawa (2004)
- [2] "Handwriting Recognition using LSTM Networks." -Sarita Yadav, Ankur Pandey, Pulkit Aggarwal, Rachit Garg, VishalAggarwal (2018)
- [3] "Word Recognition Using Fuzzy Logic." -Richard Buse, Zhi-Qiang Liu, and Jim Bezdek (2002)
- [4] "Handwriting Recognition using Deep Learning based Convolutional Neural Network" - Asha K, Krishnappa H K (2019)
- [5] "Handwritten Text Recognition using Machine Learning Techniques in Application of NLP" -Polaiah Bojja, Naga Sai Satya Teja Velpuri, Gautham Kumar Pandala, S D Lalitha Rao SharmaPolavarapu, Pamula Raja Kumari (2019)
- [6] "Handwritten Recognition by using Machine Learning Approach" -P.Thangamariappan Dr.J.C.Miraclin Joyce Pamila(2020)
- [7] "Handwritten Text Recognition:with Deep Learning and Android" -Shubham Sanjay Mor, Shivam Solanki, Saransh Gupta, Sayam Dhingra, Monika Jain,Rahul Saxena(2019)
- [8] "Handwritten Text Recognition System Based on Neural Network" -IAhmed Mahi Obaid, IIHazem M. El Bakry, IIIM.A. Eldosuky, IVA.I. Shehab (2016)
- [9] "Handwriting Recognition using Artificial Intelligence Neural Network and Image Processing" -Sara Aqab1, Muhammad Usman Tariq (2020)
- [10] "Handwritten Character Recognition using Deep Learning in Android Phones" -Athira M Nair1, Chrissie Aldo1, Blessil Bose1, Alex Joseph1, Praseetha V.M2, Sr. Elizabeth M J (2021)