EXTRACTION OF TEXT FROM AN IMAGE USING MACHINE LEARNING

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of,

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Ch Suresh Reddy	**321710303009**
C Charan Kumar	**321710303007**
K Bhavya	**321710303025**
Jaya Krishna	**321710303048**

Under the esteemed guidance of

Dr.Komarasamy G

Assistant Professor

Department of Computer Science & Engineering,

GANDHI SCHOOL OF TECHNOLOGY AND MANAGEMENT

(Deemed to be University)

Bengaluru Campus.

May 2021.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

We, hereby declare that the project report entitled **"EXTRACTION OF TEXT FROM AN IMAGE USING MACHINE LEARNING"** is an original work done in the **Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University)** submitted in partial fulfillment of the requirements for the award of the degree of **B.Tech.** in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree.

Date:

Registration No(s).	Name(s)	Signature(s)
321710303009	**Ch Suresh Reddy**	**Suresh**
321710303007	**C Charan Kumar**	**Charan**
321710303025	**K Bhavya**	**Bhavya**
321710303048	**Jaya Krishna**	**Jayakrishna**

ACKNOWLEDGEMENT

We had been able to complete our project successfully. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them.

Thanksgiving will not end without thanking the parents where even the word Thanks is not enough for the sacrifices they made. I would like to thank my parents for everything they have given and have made my path easier and to reach this point.

We are highly indebted to GITAM (Deemed to be University), Bangalore for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

Would like to express our gratitude towards Prof. S Dinesh (Director of GST), Prof. Vamsidhar Y (HOD of CSE, GST), Dr. Komarasamy G (Assistant Professor, CSE, GST) and also to all the other supporting faculty and staff for their kind co-operation and encouragement which helped us in the completion of this project.

Student Name’s	Registration No.
Ch Suresh Reddy	321710303009
C Charan Kumar	321710303007
K Bhavya	321710303025
Jaya Krishna	321710303048

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled **“EXTRACTION OF TEXT FROM AN IMAGE USING MACHINE LEARNING”** is a bonafide record of work carried out by **Suresh Chilukuri(321710303009), Charan Kumar(321710303007), K Bhavya(321710303025), Jaya Krishna(321710303048)** students submitted in partial fulfillment of requirement for the award of degree of **Bachelors of Technology in Computer Science and Engineering**.

Project Guide.	**Head of the Department.**
Dr. Komarasamy G,	**Dr. Vamsidhar Yendapalli,**
Assistant Professor	Professor.

ABSTRACT

Handwritten character recognition is a branch of artificial intelligence, computer vision, and pattern recognition that studies the recognition of handwritten characters. A computer capable of handwriting recognition is said to be able to acquire and detect characters in paper documents, images, touch-screen devices, and other sources, and transform them into machine-encoded form. Its use can be seen in optical character recognition, transcription, and other fields.

Since handwritten documents are so common in human transactions, optical character recognition (OCR) of documents is extremely useful. Optical character recognition is a science that allows different types of documents or images to be translated into searchable, editable, and analyzable data.Researchers have used artificial intelligence/machine learning software to process handwritten and typed documents in order to convert them to electronic format over the last decade.

Image recognition can be thought of as a subset of handwritten character recognition. The algorithm basically takes an image (a handwritten digit) as input and outputs the probability that the image belongs to various groups (the machine-encoded digits, 1–9).

TABLE OF CONTENTS

Title **Page No.**

LIST OF FIGURES

INTRODUCTION

CHAPTER-1

Machine Learning has become one of the mainstays of information technology over the last two decades, and with it, a very important, but mostly secret, part of our lives. With the ever-increasing amount of data available, there's reason to believe that smart data analysis will become much more prevalent as a necessary component of technological advancement.

Human designers also create devices that do not perform as well as they can in their environments. In reality, some aspects of the working environment could not be fully understood at the time of design. Machine learning techniques may be used to improve current machine designs on the fly.The amount of information available about certain activities can be too big for humans to encode explicitly. Machines that eventually gain this information might be able to capture more than humans would like to record.

The environment evolves over time. Machines that can adjust to their surroundings will eliminate the need for constant redesign.Humans are actively discovering new information about tasks. The vocabulary evolves. In the universe, there is a never-ending stream of new events. It's impractical to keep redesigning AI systems to adapt to new information, but machine learning methods may be able to track a lot of it.

1.1.WORKING OF OCR

The identification of printed or written text characters by a machine is known as OCR (optical character recognition). This entails character-by-character photo scanning of the text, review of the scanned-in image, and then translation of the character image into character codes, such as ASCII, which are widely used in data processing.

Many modern OCR systems are designed using conventional approaches to image processing, and although they work well with printed text, they can produce unexpected results with poor recognition quality when used for handwritten text recognition in images.

The Below F igure1.1 shows the working of OCR where it takes image as input and gives the text as Output.

Figure1.1:Working Of OCR

1.2.APPLICATIONS OF OCR

Many types of domain-specific OCR applications have been created using OCR engines, including receipt OCR, invoice OCR, search OCR, and legal billing document OCR.

They can be used for a variety of purposes, including:

- Data entry for business records such as checks, passports, invoices, bank statements, and receipts
- Automatic insurance documents main information extraction[citation needed] in airports for passport recognition and information extraction
- Recognizing traffic signs and extracting contact details from business cards into a contact list.
- Make textual copies of printed documents more easily, for example, by scanning books for Project Gutenberg. Make electronic images of printed documents searchable, for example, by Google Books.
- Using real-time handwriting to power a computer (pen computing) converting scanned documents to searchable PDFs

LITERATURE STUDY

CHAPTER-2

Sl. No.	Titleof the Paper	Journal Name, Publisher Name, Year of Publication and Volume & Issue Number (Only SCI)	Author Name	Problem Addressed / Problem Statement	Methods/ Technologies Used	Author Contribution	Shortcomings/ Deficiency / Assumption Made (Research Gap)
1	Based on improved edge detection algorithm for English text extraction and restoration from color images	Year 2020 , Volume: 20,	Jianbo Xu, Wenhan Ding, Hanbing Zhao	The algorithm needs to split and merge the image for each image, and most of the generated connected elements are not text, so the algorithm complexity is too high and the system efficiency is low.	Therefore,(1) it is necessary to use image restoration technology to repair the text target as a damaged area, restore the background area covered by it, and add new text, (2) Determine the best match module.(3) Update priority value	Author defined , extracting the text from the image and translating it will facilitate communication between different countries and different cultures	Accuracy and recall are two measures used in the field of information retrieval and statistical classification to evaluate the quality of results. The accuracy is the number of correctly extracted text areas divided by the total number of extracted text areas. Recall is the number of correctly extracted text areas divided by the number of text areas pr

							esent in the image.
2	Extraction Algorithm of English Text Information From Color Images Based on Radial Wavelet Transform	Year: 2020 |	Yaqin wang	various signal extraction techniques based on radial wavelet transform modulus Maximal are analyzed, and it is found that these techniques have poor ability to extract weak signals, and the higher requirements for directionality lead to pseudo-boundary phenomena in two-dimensional image extraction results.	The number of samples extracted from S VT-char and ICDAR 2003 data sets is relatively small, which leads to the imbalance of samples during extractor training,A large number of heterogeneous base extractors are trained for each extraction sub-problem, including KNN, ANN, TREE and radial wavelet entropy extractors	Author said , a large number of images and videos often contain certain English text information, and the English text information in these images reflects part of the important content of the image or video to a certain extent.	he radial wavelet transform coefficients represents the intensity of the gray change of the original signal at this resolution, and the points with larger local energy values represent the obvious characteristics of the original signals. Therefore, the energy value of each point can be calculated by the value of radial wavelet transform coefficients
3	A New Deep Learning-Based Handwritten Character Recognition System on Mobile Computing Devices	Mobule network and applications , 25 402-411(2020)	Yu Weng &Chunlei Xia	use the mobile to collect data, process the data, and construct the data set	CNN	Author contributed that by using advanced information processing methods, such as machine learning as well as big data collection and analysis, we can change the traditional document preservation methods and urgently establish meaningful digital pre	The amount of training data and the computational requirements of a conventional neural network are extremely large, and mobile devices are over burdened by the amount of training calculations.

						servation.	
4	On developing handwritten character image database for Malayalam language script	Engineering Science and Technology, an International JournalVolume 22, Issue 2, April 2019,	K.Manjusha, M. AnandKumar, K.P. Soman	Considerable research efforts for handwritten Malayalam character recognition are present in literature. Still, no public domain handwritten image database is available for the Malayalam language.	Malayalam language, Handwritten character recognition, Handwritten character image database,Active contour minimization,Optical character recognition	In this paper the Malayalamcharacter classes are decided based on the unique orthographicstructures in Malayalam language script. 85 Malayalam characterclasses representing vowels, consonants, half-consonants, vowelmodifiers, consonant modifier and conjunct characters that arefrequently used while writing are considered for database crea	The misclassifications happened for ReducedScatCN recognizer on testing dataset are analyzed with the help of confusion matrix.
5	Graphological Analysis and Identification of Handwritten Texts	27 October 2017, volume 136	Alexander V. Nikitin Nina N. ReshetnikovaNikolay V. Solowiev	The firstthing is to state clearly what is solved and what remains. The second thing isto compare the two approaches with each other, i.e., the statistical and structural approaches, in terms of both theoretical and pract	The two representative approaches i.e., statistical and structural approaches for character recognitionwere compared with each other focusing on their inner structures in the historical order.	The tremendous work to solve these problems is looked at from the angle of "space versus description."	In case of thedescriptive approach, the ideal object of one template per category seems tobe feasible, in the sense that each template set corresponding to a category isconstructed easily by one ideal template shape. On the contrary, in the case of the space approach how to select the spanning vectors is a big problem.

				ical points of view.			
6	Automatic tracing and extraction of text-line and word segments directly in JPEG compressed document images	Year: 2020 | Volume: 14,	Bulla Rajesh;Mohammed Javed;P. Nagabhushan	. The major challenge with these images is that its processing becomes expensive as it requires repeated decompression and recompression operations. Recently, it has been proved that developing algorithms to operate directly on the compressed data is one of the solutions in overcoming the above issue	the space penetration algorithm ,a moving window algorithm along with a shift operation.	Author siad,JPEG is one of the popular and efficient compression algorithms supported in the consumer electronics world. Excessive usage of mobile phones and e-governance applications have all resulted in a huge collection of JPEG compressed document images. The major challenge with these images is that its processing becomes expensive as it requires repeated decompression and recompression operations.	The observation is that, if it takes the immediate zero and goes forward, in the long run, the large font size and large skew will bring the window into the middle of text-line and give the wrong direction for segmentation

SOFTWARE AND HARDWARE SPECIFICATIONS

CHAPTER-3

3.1 INTRODUCTION

Software Software Requirements (SRS) is a basic document, based on a software development process It not only calculates system requirements, but also describes its main features. SRS is basically (organizational) customer understanding or customer plan requirements and reliance on a specific time (frequency) before any actual construction or development works. It is a two-tier insurance policy that ensures that the client and the organization understand each other's needs from that point of view in a given time. SRS also serves as a project to complete a project at the lowest possible cost. SRS is often referred to as the "parent" guide because all of the following project management documents, such as design specifications, job descriptions, software software design details, testing and verification programs, and documentation programs, relate to it. It is important to note that SRS covers only operational, operational and non-operational requirements; Does not provide development proposals, potential technical solutions or business problems or other information unless the development team understands what the customer should be.

3.2 PURPOSE OF SRS

This definition of the need for software provides a complete overview of all the functions and definitions of "identifying small neck prices and bottles to prolong the life of the network". It also specifies non-functional requirements such as reduced time, reliability and failure to deliver etc. It also focuses on the program's target users. Includes thinking designed to keep working; Software software and hardware are required to manage the product.

3.2.1 SPECIFIC REQUIREMENTS

This section provides details about the operation of the system. And that's obvious

Issues are taken into account in providing such operations.

- Prefers Linux, Windows 10 and on.
- The system with 16 GB RAM is compatible with 10 GB hard disk.
- Dataset data is required for a handwritten data set.
- The skills we learn in this project are machine learning, Python.

3.3 DATA SET

The IAM signature database includes English handwritten text types that can be used to train and verify signature identifiers and perform author identification and verification tests.

The website ICDAR was first published in 1999. The HMM Handwriting Recognition System was developed using this database and published in ICPR 2000. The classification system used in the second version of the information has been written and published. In ICPR 2002. I.A.M. of October 2002. Described in the database. We make extensive use of databases in our research, refer to the literature for more details.

The database contains unencrypted text types, which are scanned at 300dpi resolution and stored as 256-degree PNG images. The following figure provides a sample of the complete form, text line, and other words used.

The Figure 3.3 Shows the Sample IAM Handwritten DataSet where the IAM DataSet is a collection of Handwritten Images with variable styles.

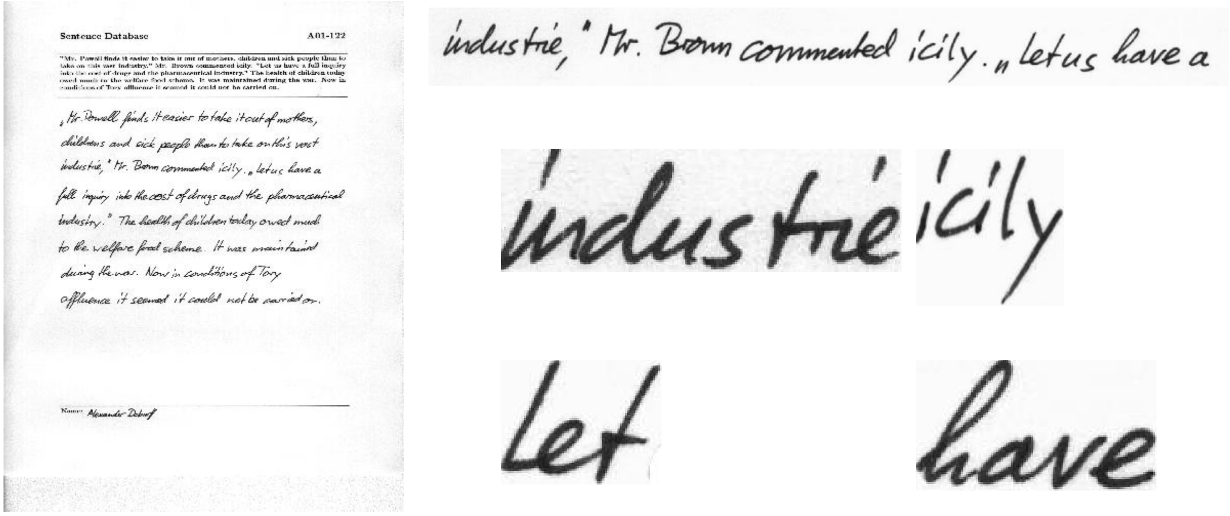

Sentence Database
A01-122
industrie," Mr. Brown commented icily. " let us have a
industrie
icily
let
have

Figure 3.3:Sample IAM HandWritten Dataset

All files and all processed text lines, names and sentences are available for download as PNG files, image files contain relevant meta-XML details. All documents in the IAM database are compiled using the sentences provided by the LOB corpus.

IAM Hand. The Hand Signature Database is configured as follows:

- 6656 authors donate handwritten specimens,
- Text1'538 pages of scanned text
- '5'689 sentences selected and labeled
- '13'363 text lines alone and labeled
- 5115'321 words are separated and labeled

These words are extracted from scanned text pages using an automated separation system and are manually verified. A separate plan has been drawn up at our facility.

All forms, lines and image images are provided as PNG files and related form label files, including partition details and various parameters measured (from the described step-by-step), image files as meta data in XML format. XML file and in XML file format (DTD).

3.3.1 ENGLISH LANGUAGE

English is the most widely used language in the world. It is the official language of 53 countries and is spoken as the first language by nearly one million people. Two languages use English as an international language. The validity of English language letters has been studied extensively for years. In this systematic review of texts, the English language has the most publications, meaning 45 publications after completing the research selection process. English-language OCR programs play an important role as a number of courses are taken in English during the period 2000-2018.

English language OCR systems have been used successfully in various trading systems. The most comprehensive study of English manuscript OCR by Plumndon and Shrihari in 2000 with over 2900 quotes. Purpose of the study by Plemon et al. Will present a detailed review of the state of the art in the field of automated signature processing. This paper describes the possibility of computer based computers and fulfills the purpose of automated electronic ink processing by copying and expanding the metaphor of the paper. Character structure, structural models and rules such as (SOFM) self-configured feature map, (TDNN) neural network delay time and (HMM) to use hidden Markov model. Extensive overview of character recognition presented by Erica et al. There are over 500 quotes. Erica et al. Concluded that characters are natural units, and respect for characters is unlikely to impose strict mathematical law on drivers' method. Structural or mathematical models cannot show any complex patterns on their own. Mathematical details and the formation of multi-character patterns can be combined with neural network (NN) or harmonic Markov (HMM) models.

3.4 SOFTWARE ENVIRONMENT AND TOOL

3.4.1 PROGRAMMING LANGUAGE

Our project Programming mustr be dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented programming and it must be easy to code.

3.4.2 PYTHON

Python is a powerful typewriter and garbage collection. Supports multiple editing patterns, including structured, object-oriented, and active applications. Python is often described as a "battery powered" language because of the general library.

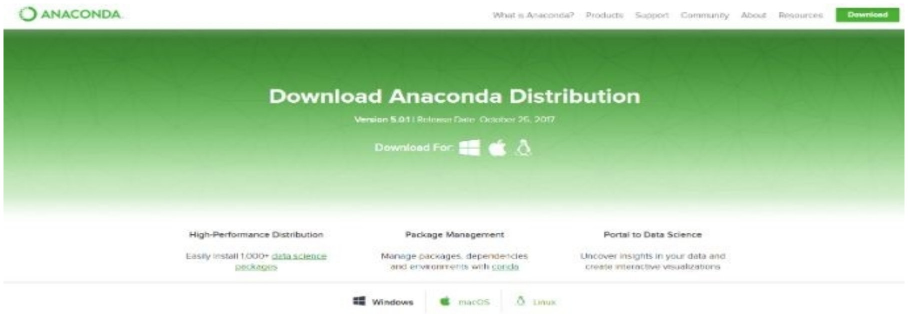
Python is a high-quality language with translated, object-oriented, dynamic economics. Its high-quality built-in data architecture, combined with powerful typing and dynamic binding, makes it extremely attractive for fast application development, as well as for use as a writing or paste language to connect existing components together. Python's simple, easy-to-read syntax emphasizes readability and therefore reduces system maintenance costs. Supports Python modules and packages, promoting system layout and code reuse. Python Interpreter and General Library is available free of charge and distributed free of charge on all major platforms in binary form.

3.4.3 PYTHON INSTALLATION WITH ANACONDA:

Anaconda is a free open source distribution of python for large scale data processing, predictive analytics and scientific computing.

- Conda is a package manager that quickly installs and manages packages. Anaconda for Windows installation:
- Go to the following link: Anaconda.com/downloads

Figure 3.4.3A shows the options to download Anaconda



Download Anaconda Distribution

Version 5.0.1 | Release Date: October 26, 2017

High-Performance Distribution

Easily install 1,000+ data science packages

Package Management

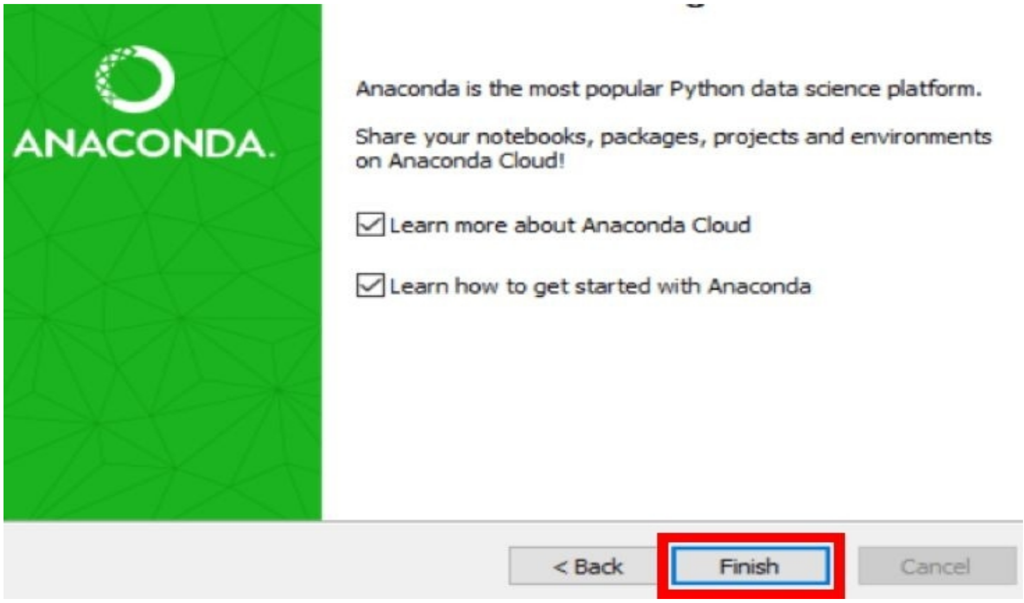
Manage packages, dependencies and environments with conda

Portal to Data Science

Uncover insights in your data and create interactive visualizations

Figure 3.4.3A:Select the option to download Anaconda

- Download python 3.4 version for (32-bit graphic installer/64 -bit graphic installer)
- Select path(i.e. add anaconda to path & register anaconda as default python 3.4)
- Figure 3.4.3B shows the after installation image
- Click finish



Anaconda is the most popular Python data science platform.

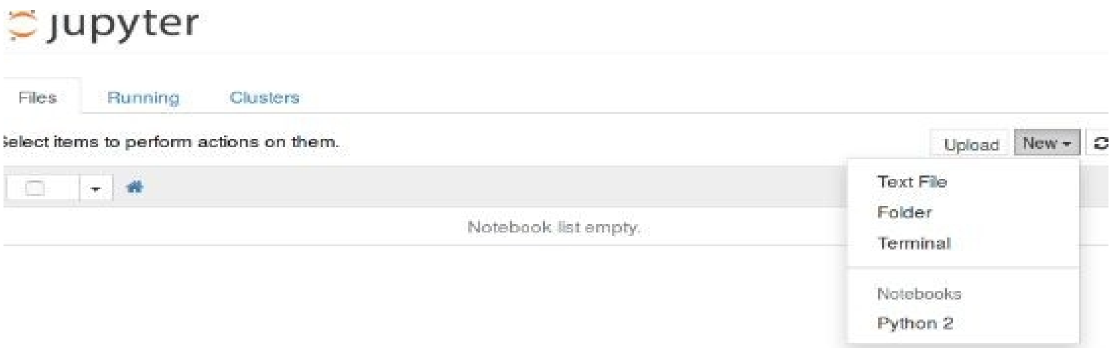
Share your notebooks, packages, projects and environments on Anaconda Cloud!

☑ Learn more about Anaconda Cloud

☑ Learn how to get started with Anaconda

Figure 3.4.3B After installation

- The Figure 3.4.3 C shows the Jupyter notebook look like
- Open jupyter notebook



Select items to perform actions on them.

Notebook list empty.

Figure 3.4.3C: Jupyter notebook

3.5 PACKAGES

Nampi is a standard inventory processing package. It provides the most effective tool for equal members and the tools to work with these members. It is a Python based computer package. In addition to its explicit scientific uses, Nampi can also be used as a multimedia container for general information.

Keras is an open and easy-to-use Python library for developing and testing in-depth learning models. It complements Theono and Tenserflow's efficient numerical libraries and lets you define and train neural network models in just a few lines of code.

Pandus is an open library built on top of the NumPy library. It is a Python package that provides various data structures and functionality for modifying numerical data and time series. It is known that importing and analyzing data is very easy. Panda is fast and has high performance and user productivity.

Python-Tesseract is the cover for Google's Teserect-OCR engine. It is useful as a standalone text to process, as it can read all kinds of images supported by the pillow and leptonica libraries, including JPAG, PNG, GIP, BMP, TIFF and others.

Python-Tesseract is a Python (OCR) recognition tool. That is, it will see and "read" the text embedded in the images. Python-Tesseract is the cover for Google's Teserect-OCR engine.

PROBLEM STATEMENT

CHAPTER-4

The OCR software software system was developed and distributed to educational institutions, adopting OCR census and metal stamps. In the early 2000's, glass-cutting techniques were introduced to preserve digital-level historical documents and give researchers access to them. In the mid-2000s, a number of applications were created to help people with different backgrounds. This application has improved the literacy skills of these people. Over the past decade, researchers have worked on a variety of machine learning methods, including support vector machines (SVMs), random forests (RF), or nearest neighbors (KNNs), decision trees (DTs), neural networks, and more. Techniques in machine learning and image processing techniques to increase the accuracy of the visual character system. Recently, researchers have focused on developing techniques for computerized handwriting, especially in the field of in-depth reading. This pattern has also emerged as a result of better performance, recurrent neural network (RNN), convulsive neural network (CNN), long short-term memory (LSTM) network, etc. through cluster computing and GPU adoption and deep infrastructure.

1. In our daily lives, people face many language problems. For example, if people move from one state to another without understanding their language, then this mobile app will help them. The existing system, with a unique application for each camera process, a Google Translator and a text scanner for Optical Character Recognition (OCR). However, people expect the app to keep these three components together. So this proposed application gives people a new perspective to translate other text into English. The program consists of three steps.

2. Take a picture of the English handwritten text you want to translate (either handwritten or printed).

3.Tessaract is an open source ical Practical Character Recognition (OCR) technology, used to extract text from an image and translate the Google API and Bing API.

4Translated text is our result.

4.1 OBJECTIVES

- After taking the image highlight the characters that are persent in the image.
- We need to give Hand Written Text as a Input Image.
- Cleaning the image by removing the waste present in the image

DESIGNIG

CHAPTER-5

5.1 ALGORITHM USED:

Analysis is a way to find the best solution to a problem. System analysis is the process by which we learn about existing problems, define resources and needs and evaluate solutions. It is a way of thinking about the organization and the problem involved, a collection of technologies that help solve these problems. Ongoing research plays an important role in system analysis that provides the goal of construction and development. System Design is a creative process, good design is the key to a functional plan. A system design is defined as the process of applying various techniques and principles in order to define a process or system with sufficient detail to allow for its physical discovery. Various construction features are followed to improve the system. Design specification describes the features of a program, components or elements of a program and its appearance that eliminates users.

Here algorithm used is:

CNN LTSM : A short-term memory network is, in short, an LSTM structure designed specifically for problems predicting sequence by location input, such as photos or videos. Local-based inputs, such as photos, cannot be easily created with standard vanilla LSTM.

5.2 MODEL BUILDING AND EVALUATION

New LSTM:

Duplicate layers, such as LSTM, but the internal iteration of the matrix is replaced by the convolution function. As a result, the data flowing in the ConviLSTM cells retains the size of the input (3D for us) instead of the 1D vector with features. With features received. When you repeat this process for all images at the scheduled time, the result is a set of features over time, and this is an LSTM level installation.

ConvilSTM Level Installation:

LSTM cell input into a set of data over time, i.e., a 3D tensor with conditions (samples, time measures, features). The Convolution Layer Input is a set of images as standard 4D tensors (templates, channels, rows, cumns). A new LSTM input into a set of images over time as a 5D status tensor (templates, time steps, channels, rows, etc.).

ConvLSTM layer output:

LSTM cell output is based on return_sequences. If set to true, output sequence over time (single output per input). In this case, the output is a single 3D tensor (samples, time measures, features). When the return_follow is set to false (default default), the output is the last order value, i.e., 2D tensor with position (samples, features).

ConviLSTM is a combination of layer effect convolution and LSTM output. Like LSTM, if return_squences = true, then return the order as standard 5D tensor (samples, time measures, filters, rows, cumns). On the other hand, if return_sequences = false, then it only returns the final value of the sequence as a standard 4D tensor (samples, filters, rows, cumns).

5.2.1 SYSTEM ARCITECTURE

This section provides a high-level overview of how the functionality and the responsibilities of the system were portioned and then assigned to each phase in the figure appropriately.

The Figure 5.2.1 shows the Architecture Diagram of our project.

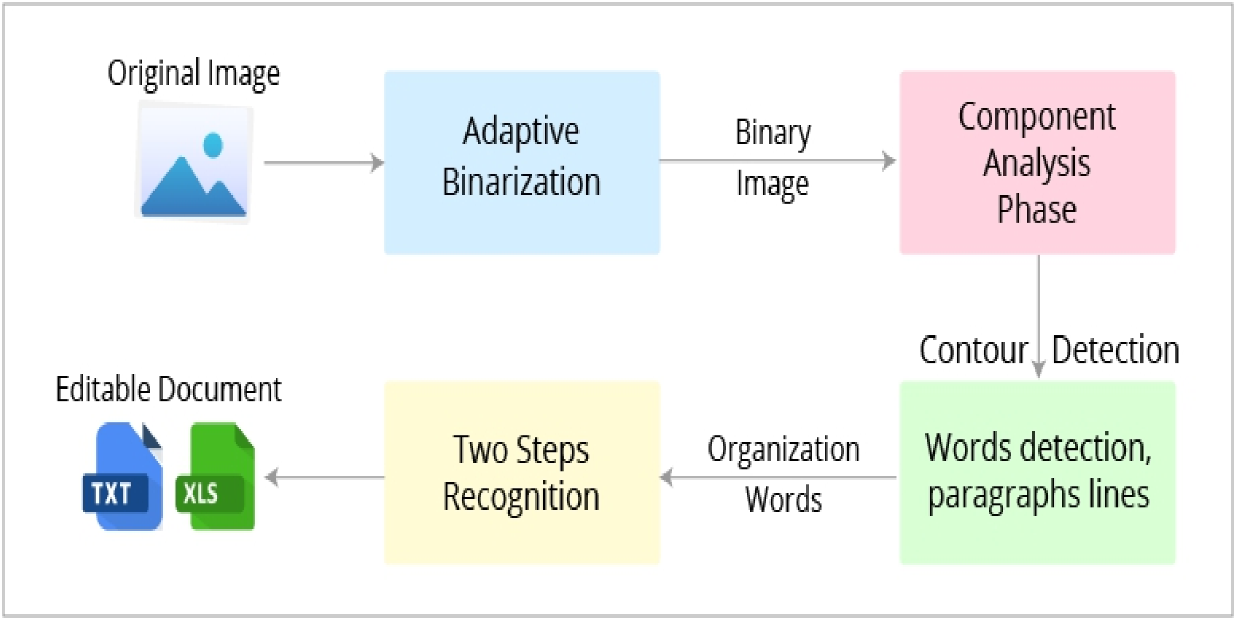


Figure 5.2.1: Architecture Diagram

5.3 METHODOLOGY

5.3.1.TERMS

Before we have a tendency to conceive to justify the assorted techniques employed in TIE, it's necessary to outline the ordinarily used terms and summarize the characteristics of text which will be used for TIE algorithms. Text in pictures will exhibit several variations with relevancy the subsequent properties.

Geometry Size: though the text size will vary heaps, assumptions may be created counting on the appliance domain.

Alignment: The characters within the caption text seem in clusters and typically lie horizontally, though generally they'll seem as non- planate texts as a results of computer graphics. This doesn't apply to scene text, which might have varied perspective distortions. Scene text may be aligned in any direction and might have geometric distortions

Inter-character distance: characters in an exceedingly text line have a homogenous distance between them.

Colour :The characters in an exceedingly text line tend to own a similar or similar colors. This property makes it potential to use a connected component-based approach for text detection. Most of the analysis reportable until date has focused on finding 'text strings of one color (monochrome).However, video pictures and alternative complicated color documents will contain 'text strings with quite 2 colors (polychrome)' for effective mental image, i.e., totally different colors among one word.

Motion:The same characters sometimes exist in consecutive frames in an exceedingly video with or while not movement.This property is employed in text pursuit and sweetening. Caption text sometimes moves in an exceedingly uniform way: horizontally or vertically Scene text will have absolute motion thanks to camera or object movement.

Edge :Most caption and scene text ar designed to be simply browse, thereby leading to sturdy edges at the boundaries of text and background

Compression: several digital pictures ar recorded, transferred, and processed in an exceedingly compressed format. Thus, a quicker TIE system may be achieved if one will extract text while not decompression .

5.3.2 PRE-PROCESSING

Before starting text recognition, it is important to analyze the text image in light and dark areas to see all the letters of the alphabet.

For that purpose, we would like to offer you first-hand image processing. Pre-processing law covers many requirements and initially looks for specific steps:

1. Image binarization

2.waste clearing

3.Text Line Detection

4 .Character Discovery

Image Binaryization: The binary method is largely based on image quality. In this case, you may find very poor quality pictures.

First, using the bar chart, let us construct the background size. The law can give the North American country the most effective results in the creation of color-coded images. Here are the images processed while applying the law.

The figure 5.3.2A is the image befor applying image binarization and the Figure 5.3.2B shows the binarized image.

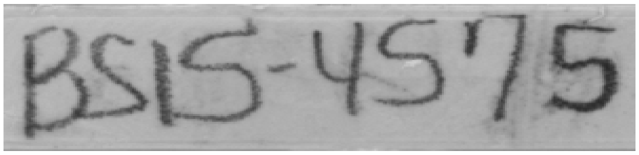


Figure 5.3.2A: Before Image Binarization

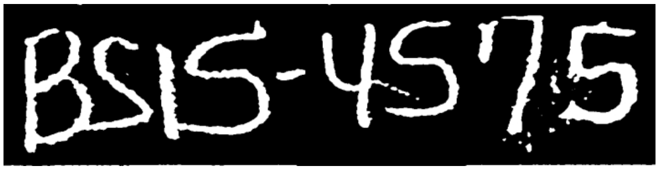


Figure5.3.2B : After Image Binarization

WASTE CLEARING: After completing the first step, we have amazing images, where groups of white pixels form characters and black pixels create a background, As you seen the below Figure 5.3.2C. Although you will see that there are large varieties of white pixels that are not part of the character, they require blobs made with sound which is not exactly desirable in the theory of character recognition.After we clearing the waste from the figure5.3.2C we get the Cleaned the Figuer 5.3.2D

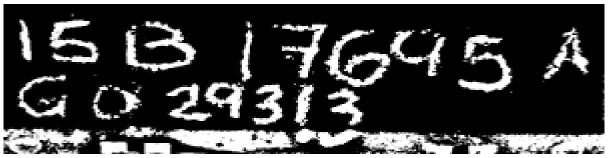


Figure5.3.2C : Before Waste Clearing

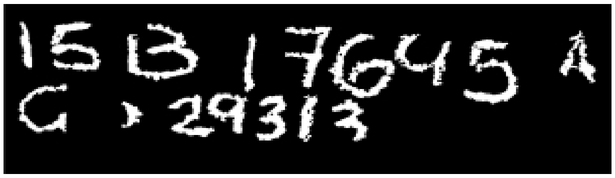
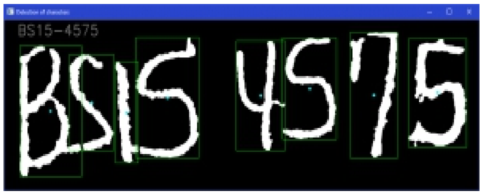


Figure 5.3.2D: After Waste Clearing

TEXT LINE DETECTION: One of the problems with text recognition is when it comes to finding the right text lines. The text line seems to be inconsistent and sometimes 2 neighboring lines will overlap, so we will apply the revised Huff rule to the modified images.

Character Discovery: ,Finding characters is the last and most difficult step. For example, As you seen the below Figure 5.3.2E some letters can be broken or inserted into elements.



```
BS15-4575
```

Figure5.3.2E : Final Output Image.

5.4 DATA FLOW DIAGRAM

Below Figure 5.4 shows the how over project steps takes place,we provided all those in the form of Data Flow Diagram

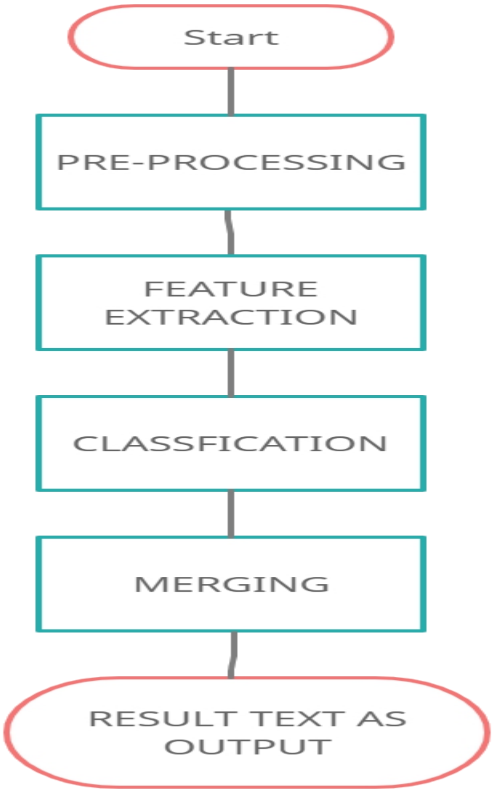


Figure 5.4: Data Flow Diagram

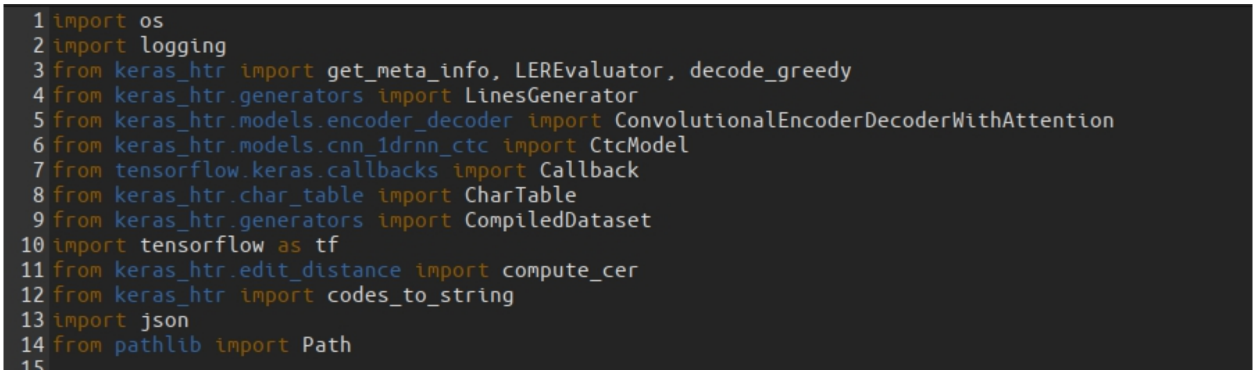
IMPLEMENTATION

CHAPTER-6

6.1. IMPORTING THE PACKAGES:

Import the necessary packages required to perform Data pre processing steps

The Figure 6.1 shows the how code how to import the packages that are using in our project



```python
import os
import logging
from keras_htr import get_meta_info, LEREvaluator, decode_greedy
from keras_htr.generators import LinesGenerator
from keras_htr.models.encoder_decoder import ConvolutionalEncoderDecoderWithAttention
from keras_htr.models.cnn_1drnn_ctc import CtcModel
from tensorflow.keras.callbacks import Callback
from keras_htr.char_table import CharTable
from keras_htr.generators import CompiledDataset
import tensorflow as tf
from keras_htr.edit_distance import compute_cer
from keras_htr import codes_to_string
import json
from pathlib import Path
```

Figure 6.1: Importing Packages

6.2. BUILDING DATASET:

We have built this data set using an IAM dataset . In this data there are characters of different alphabets, numericals of different hand writings.

To build the data set the following command is used.

python build_lines_dataset.py --source=keras_htr.data_source.synthetic.SyntheticSource' --destination=temp_ds --size=100

The Figure 6.2 is the floder of textocr contain all the below files shown in the figure 6.2A

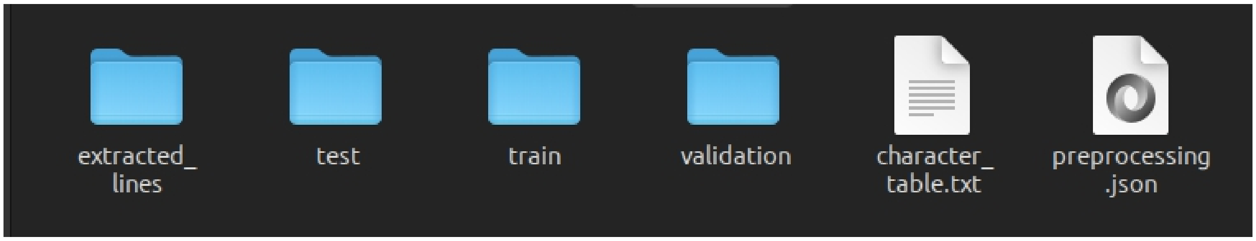


Figure 6.2:Building Data Set

It is a generator of handwritings taken from the IAM handwriting database. Before you can use this source, you have to download the actual database.

- create a directory named iam_database in the repository directory
- register an account on http://www.fki.inf.unibe.ch/databases/iam-handwriting-database
- download xml.tgz archive file (you will be prompted to enter your password)

Curl -o iam_database/xml.tgz -u <user_name>
http://www.fki.inf.unibe.ch/DBs/iamDB/data/xml/xml.tgz

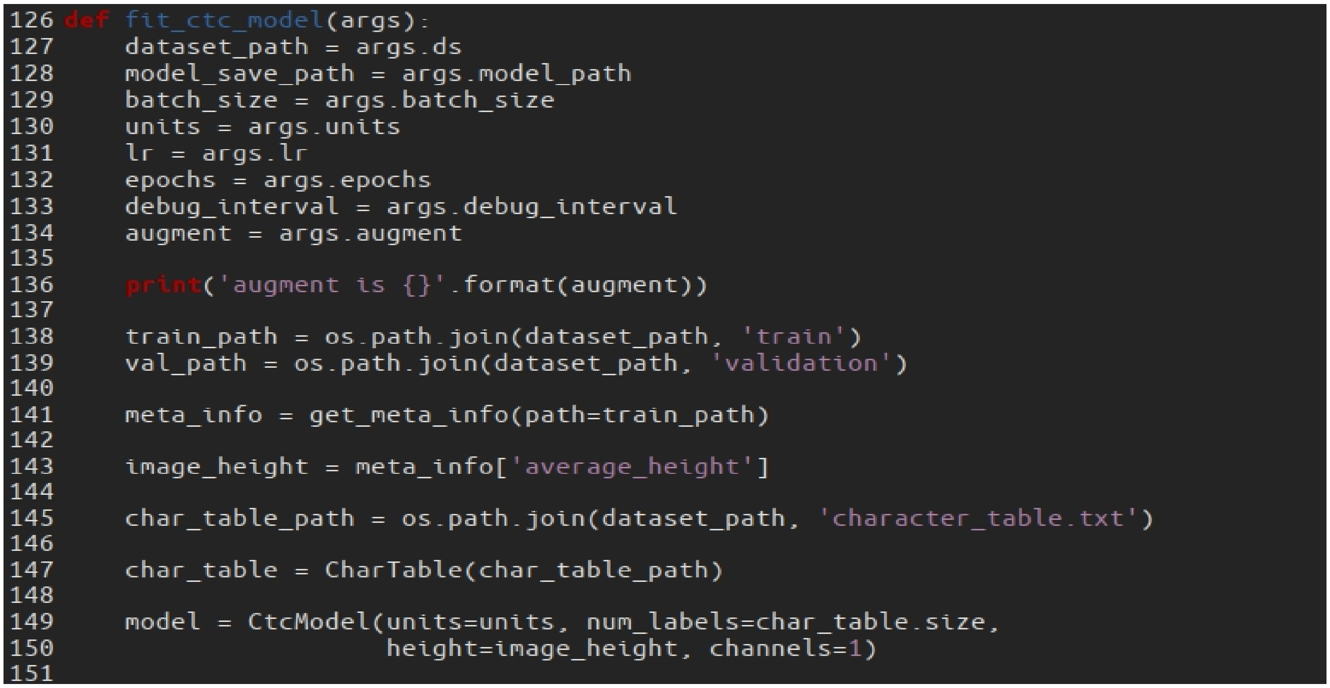
download lines.tgz archive file (you will be prompted to enter your password)

curl -o iam_database/lines.tgz -u <user_name>
http://www.fki.inf.unibe.ch/DBs/iamDB/data/lines/lines.

6.3 TRAIN-TEST-SPLIT

When starting a modeling project you need to first make a decision, how to use the available data. Another common process is that information is divided into two general groups as training and assessment sets. The training set is used for model and feature set development; There is a substrate for measuring the parameters, comparing the model and all other functions required to achieve the final model. To measure the final, flawless test performance of the model, tests are used only on the conclusion of these functions. It is important that the test set can be used before this point. Looking at the test set results can affect the results as the test data model will be part of the development process.

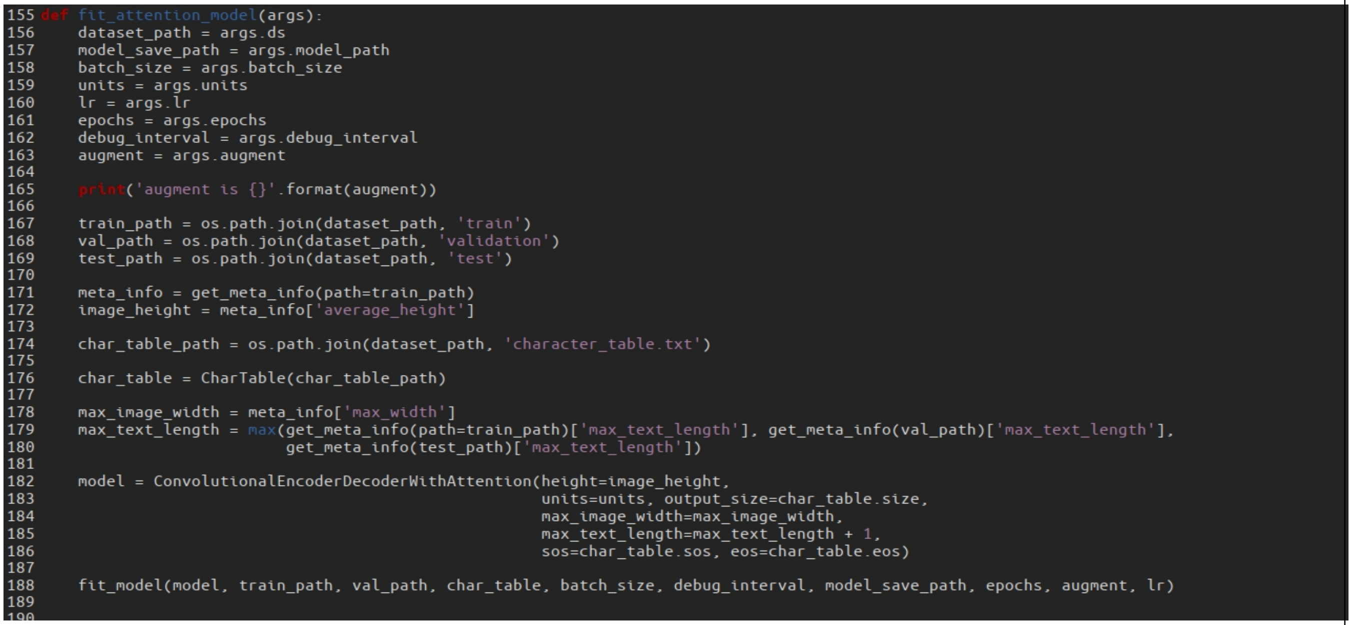
The figure 6.3A shows the how to do cnn-1drnn-ctc type model fitting



```
def fit_ctc_model(args):
    dataset_path = args.ds
    model_save_path = args.model_path
    batch_size = args.batch_size
    units = args.units
    lr = args.lr
    epochs = args.epochs
    debug_interval = args.debug_interval
    augment = args.augment

    print('augment is {}'.format(augment))

    train_path = os.path.join(dataset_path, 'train')
    val_path = os.path.join(dataset_path, 'validation')

    meta_info = get_meta_info(path=train_path)

    image_height = meta_info['average_height']

    char_table_path = os.path.join(dataset_path, 'character_table.txt')

    char_table = CharTable(char_table_path)

    model = CtcModel(units=units, num_labels=char_table.size,
                     height=image_height, channels=1)

```

Figure 6.3A:cnn-1drnn-ctc type model fitting

The Figure 6.3B tells about cnn-encoder-decoder how actualy the code looks like



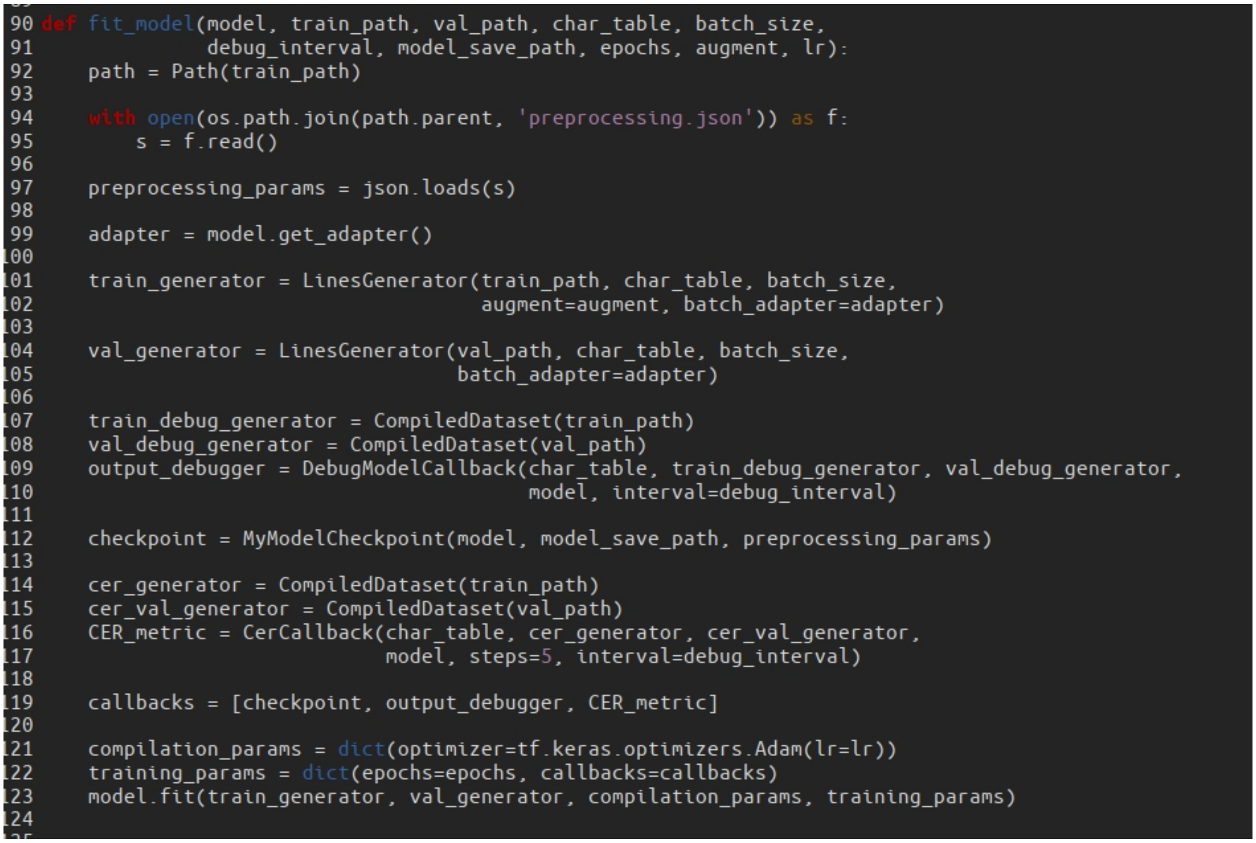
```
def fit_attention_model(args):
    dataset_path = args.ds
    model_save_path = args.model_path
    batch_size = args.batch_size
    units = args.units
    lr = args.lr
    epochs = args.epochs
    debug_interval = args.debug_interval
    augment = args.augment

    print('augment is {}'.format(augment))

    train_path = os.path.join(dataset_path, 'train')
    val_path = os.path.join(dataset_path, 'validation')
    test_path = os.path.join(dataset_path, 'test')

    meta_info = get_meta_info(path=train_path)
    image_height = meta_info['average_height']

    char_table_path = os.path.join(dataset_path, 'character_table.txt')

    char_table = CharTable(char_table_path)

    max_image_width = meta_info['max_width']
    max_text_length = max(get_meta_info(path=train_path)['max_text_length'], get_meta_info(val_path)['max_text_length'],
                          get_meta_info(test_path)['max_text_length'])

    model = ConvolutionalEncoderDecoderWithAttention(height=image_height,
                                                     units=units, output_size=char_table.size,
                                                     max_image_width=max_image_width,
                                                     max_text_length=max_text_length + 1,
                                                     sos=char_table.sos, eos=char_table.eos)

    fit_model(model, train_path, val_path, char_table, batch_size, debug_interval, model_save_path, epochs, augment, lr)
```

Figure 6.3B cnn-encoder-decoder

6.4 Train the models:

6.4.1 Training the data either with ‘cnn-encoder-decoder’ or ‘cnn-1drnn-ctc’.

The Figure 6.4.1 shows the Traning DataSet from the training part code as we seen in the folder figure 6.2.



```python
def fit_model(model, train_path, val_path, char_table, batch_size,
              debug_interval, model_save_path, epochs, augment, lr):
    path = Path(train_path)

    with open(os.path.join(path.parent, 'preprocessing.json')) as f:
        s = f.read()

    preprocessing_params = json.loads(s)

    adapter = model.get_adapter()

    train_generator = LinesGenerator(train_path, char_table, batch_size,
                                     augment=augment, batch_adapter=adapter)

    val_generator = LinesGenerator(val_path, char_table, batch_size,
                                   batch_adapter=adapter)

    train_debug_generator = CompiledDataset(train_path)
    val_debug_generator = CompiledDataset(val_path)
    output_debugger = DebugModelCallback(char_table, train_debug_generator, val_debug_generator,
                                         model, interval=debug_interval)

    checkpoint = MyModelCheckpoint(model, model_save_path, preprocessing_params)

    cer_generator = CompiledDataset(train_path)
    cer_val_generator = CompiledDataset(val_path)
    CER_metric = CerCallback(char_table, cer_generator, cer_val_generator,
                             model, steps=5, interval=debug_interval)

    callbacks = [checkpoint, output_debugger, CER_metric]

    compilation_params = dict(optimizer=tf.keras.optimizers.Adam(lr=lr))
    training_params = dict(epochs=epochs, callbacks=callbacks)
    model.fit(train_generator, val_generator, compilation_params, training_params)
```

Figure 6.4.1 Training data set

TESTING

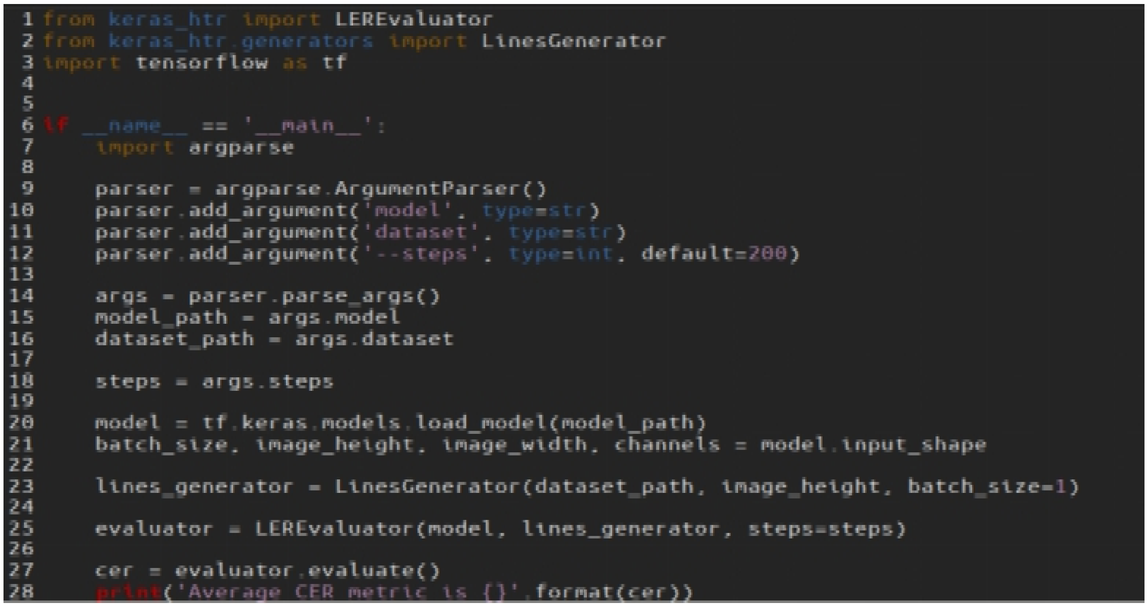
CHAPTER-7

The purpose of the test platform is to identify defects / errors by examining the components of each program. These components can be tasks, or budgets, or modules. During system testing, these components are combined to form a complete system. At this stage, the test should focus on finding out if the system meets its operational requirements, and does not behave unexpectedly. Input test data is designed to test the system while test cases are a way to test the system and output from these inputs are predicted when the system operates according to its specifications. This is a test of the effectiveness of the compatible system. Test cases have been selected to ensure that all possible combinations of system behavior can be investigated. It is often impossible to find different ways to fail software. Software testing is used in conjunction with testing and validation:Verification: Did we create the software exactly (i.e., the same as specified)?

7.1 Testing the models:

To test the model we have calculated the accuracy of different classifiers

The Figure7.1A shows the code to predict the accuracy of our project.

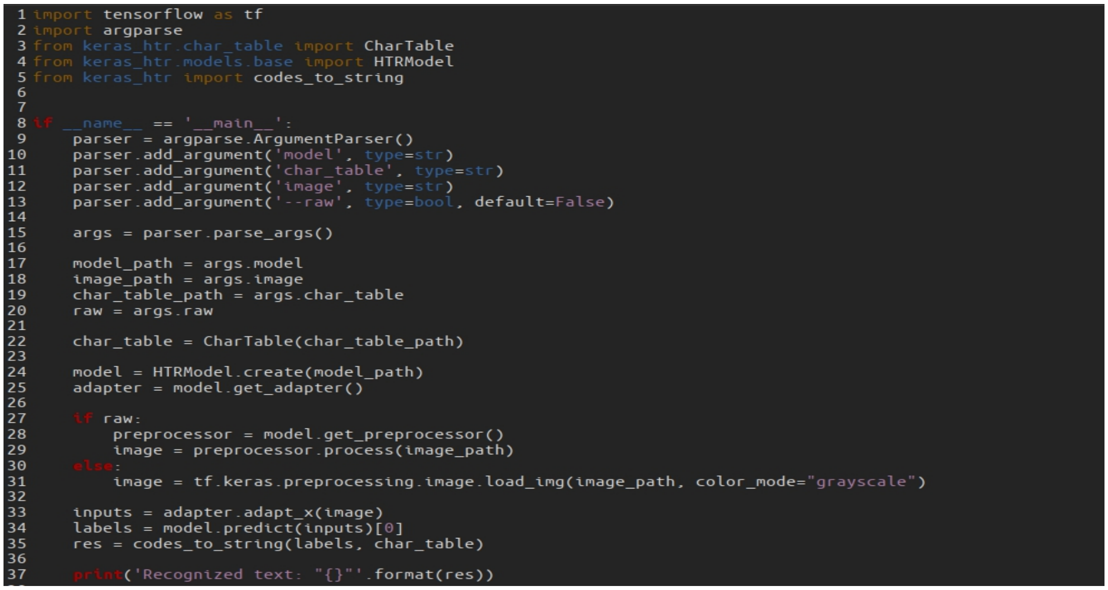


```python
from keras_htr import LEREvaluator
from keras_htr.generators import LinesGenerator
import tensorflow as tf


if __name__ == '__main__':
    import argparse

    parser = argparse.ArgumentParser()
    parser.add_argument('model', type=str)
    parser.add_argument('dataset', type=str)
    parser.add_argument('--steps', type=int, default=200)

    args = parser.parse_args()
    model_path = args.model
    dataset_path = args.dataset

    steps = args.steps

    model = tf.keras.models.load_model(model_path)
    batch_size, image_height, image_width, channels = model.input_shape

    lines_generator = LinesGenerator(dataset_path, image_height, batch_size=1)

    evaluator = LEREvaluator(model, lines_generator, steps=steps)

    cer = evaluator.evaluate()
    print('Average CER metric is {}'.format(cer))
```

Figure 7.1A Predicting Accuracy

Many researches said that they are getting only 60-64% accuracy on their IEEE published papers.At now we are getting about 60% accuracy.

The Figure 7.1B shows the code to test the images with their respective code from the Dataset.



```
import tensorflow as tf
import argparse
from keras_htr.char_table import CharTable
from keras_htr.models.base import HTRModel
from keras_htr import codes_to_string


if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument('model', type=str)
    parser.add_argument('char_table', type=str)
    parser.add_argument('image', type=str)
    parser.add_argument('--raw', type=bool, default=False)

    args = parser.parse_args()

    model_path = args.model
    image_path = args.image
    char_table_path = args.char_table
    raw = args.raw

    char_table = CharTable(char_table_path)

    model = HTRModel.create(model_path)
    adapter = model.get_adapter()

    if raw:
        preprocessor = model.get_preprocessor()
        image = preprocessor.process(image_path)
    else:
        image = tf.keras.preprocessing.image.load_img(image_path, color_mode="grayscale")

    inputs = adapter.adapt_x(image)
    labels = model.predict(inputs)[0]
    res = codes_to_string(labels, char_table)

    print('Recognized text: "{}"'.format(res))
```

Figure 7.1B Testing the images

Recognize an image taken from a test dataset after necessary preprocessing was already applied

python htr.py conv_lstm_model temp_ds/character_table.txt temp_ds/test/0.png

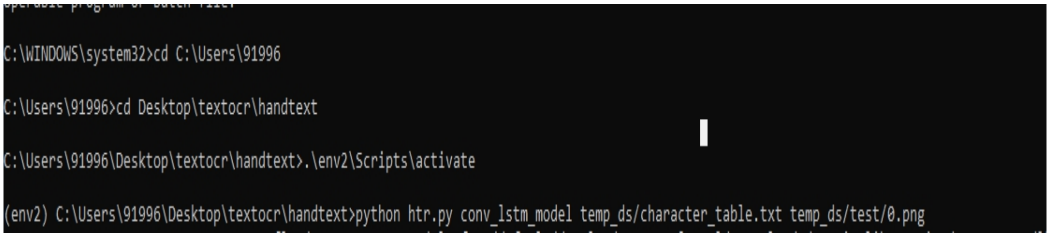
To recognize an arbitrary raw image, pass an argument --raw=True (this will ensure that all necessary preprocessing steps will be applied such as binarization, resizing, etc.):

python htr.py conv_lstm_model temp_ds/character_table.txt /path/to/unseen_image.png --raw=True

EXPERIMENTAL RESULTS

CHAPTER-8

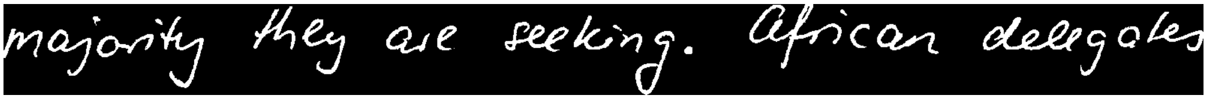
The figure 8A is the screenshot or is a photo taken of computers screen.In order to open it and run we need to open the command promt and then navigate to the path where the project files are build.Then provide the commands that contain the image name and gives it as input and execute and to get the ouputThe project executed successfully for the given set of possible inputs.



```
operable program or batch file.

C:\WINDOWS\system32>cd C:\Users\91996

C:\Users\91996>cd Desktop\textocr\handtext

C:\Users\91996\Desktop\textocr\handtext>.\env2\Scripts\activate

(env2) C:\Users\91996\Desktop\textocr\handtext>python htr.py conv_lstm_model temp_ds/character_table.txt temp_ds/test/0.png
```

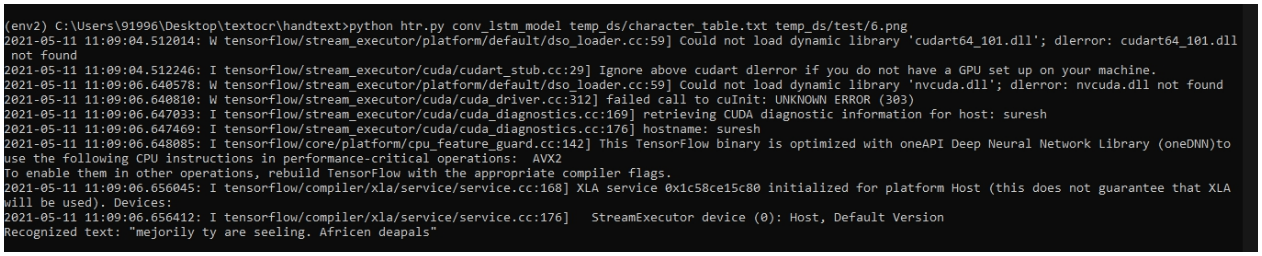
Figure 8A:Commond Prompt Commands

The below figure 8B is we are giving as input through commands in the command prompt with the figure name with .png extension.



majority they are seeking. African delegates

Figure 8B:Input Image



```
(env2) C:\Users\91996\Desktop\textocr\handtext>python htr.py conv_lstm_model temp_ds/character_table.txt temp_ds/test/6.png
2021-05-11 11:09:04.512014: W tensorflow/stream_executor/platform/default/dso_loader.cc:59] Could not load dynamic library 'cudart64_101.dll'; dlerror: cudart64_101.dll
 not found
2021-05-11 11:09:04.512246: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2021-05-11 11:09:06.640578: W tensorflow/stream_executor/platform/default/dso_loader.cc:59] Could not load dynamic library 'nvcuda.dll'; dlerror: nvcuda.dll not found
2021-05-11 11:09:06.640810: W tensorflow/stream_executor/cuda/cuda_driver.cc:312] failed call to cuInit: UNKNOWN ERROR (303)
2021-05-11 11:09:06.647033: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: suresh
2021-05-11 11:09:06.647469: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: suresh
2021-05-11 11:09:06.648085: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:  AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-05-11 11:09:06.656045: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x1c58ce15c80 initialized for platform Host (this does not guarantee that XLA
will be used). Devices:
2021-05-11 11:09:06.656412: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
Recognized text: "mejorily ty are seeling. Africen deapals"
```

Figure 8C:Output as Text

The Above Figure 8C shows the Output as a Text Format in the command prompt.

CONCLUSION

CHAPTER-9

Character recognition was present eighty years ago. However, initially, the sales that allow for visible characters were mainly created by large tech companies. Advances in machine learning and in-depth learning have enabled individual researchers to develop algorithms and techniques that can validate manuscripts with greater accuracy.

Letters written by completely different people produce great intra-class diversity, making it difficult for dividers to perform well. Nevertheless, with the increasing use of advanced teaching techniques, the search for isolated teams has consistently improved mechanical complexity (especially throughout the divisional training) is great. The underlying obstacle to the development of this time, too, is a strong system of finger recognition.

REFERENCES

CHAPTER-10

[1] M. Kumar, S. R. Jindal, M. K. Jindal, and G. S. Lehal, ‘‘Improved recognition results of medieval handwritten Gurmukhi manuscripts using boosting and bagging methodologies,’’ *Neural Process. Lett.*, vol. 50, pp. 43–56, Sep. 2018.

[2] M. A. Radwan, M. I. Khalil, and H. M. Abbas, ‘‘Neural networks pipeline for offline machine printed Arabic OCR,’’ *Neural Process. Lett.*, vol. 48, no. 2, pp. 769–787, Oct. 2018.

[3] Bulla Rajesh;Mohammed Javed;P. Nagabhushan,”Automatic tracing and extraction of text-line and word segments directly in JPEG compressed document images”,Year: 2020 | Volume: 14, Issue: 9.

[4] Rupsa Chakraborty,Jaya Sil,”Handwritten Character Recognition Systems Using Image-Fusion and Fuzzy Logic”,Lecture Notes in Computer Science book series (LNCS, volume 3776)(2019).

[5] Jianbo Xu, Wenhan Ding, Hanbing Zhao,”Based on improved edge detection algorithm for English text extraction and restoration from color imag.es”, Year 2020 ,Volume: 20.

[6] Yaqin wang,”Extraction Algorithm of English Text Information From Color Images Based on Radial Wavelet Transform”,Year:2020 ,Volume: 8.

[7] K.Manjusha, M. AnandKumar, K.P.Soman,”On developing handwritten character image database for Malayalam language script”,Engineering Science and Technology, an International Journal,Volume 22, Issue 2,April 2019.

[8] Yu Weng ,Chunlei Xia,”A New Deep Learning-Based Handwritten Character Recognition System on Mobile Computing Devices”,Mobule network and applications ,25 402-411(2020)

[9] Gauri Katiyar,Shabana Mehfuz,”A hybrid recognition system for off-line handwritten characters”,SpringerPlus 5 Article number: 357 (2019) .

[10] U.Marti and H. Bunke. A full English sentence database for off-line handwriting recognition. In Proc. of the 5th Int. Conf. on Document Analysis and Recognition, pages 705 - 708, 1999.

[11] M.Zimmermann and H. Bunke. Automatic Segmentation of the IAM Off-line Database for Handwritten English Text. In Proc. of the 16th Int. Conf. on Pattern Recognition, Volume 4, pages 35 - 39, 2000.

[12] S.Johansson, G.N. Leech and H. Goodluck. Manual of Information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital Computers. Department of English, University of Oslo, Norway, 1978.

[13] I. A. Doush, F. Alkhateeb, and A. H. Gharaibeh, ‘‘A novel Arabic OCR post-processing using rule-based and word context techniques,’’ *Int. J. Document Anal. Recognit.*, vol. 21, nos. 1–2, pp. 77–89, 2018.

[14] S. S. R. Rizvi, A. Sagheer, K. Adnan, and A. Muhammad, ‘‘Optical character recognition system for nastalique Urdu-like script languages using supervised learning,’’ *Int. J. Pattern Recognit. Artif. Intell.*, vol. 33, no. 10, Sep. 2019, Art. no. 1953004.

[15] K. U. U. Rehman and Y. D. Khan, ‘‘A scale and rotation invariant Urdu Nastalique ligature recognition using cascade forward back propagation neural network,’’ *IEEE Access*, vol. 7, pp. 120648–120669, 2019.