OPTICAL CHARACTER RECOGNITION SYSTEM FOR TELUGU HANDWRITTEN TEXT

A Project Report

submitted by

SAICHARAN GONTLA

EE16B137

in partial fulfilment of the requirements

for the award of the degree of

DUAL DEGREE

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

June 2021

THESIS CERTIFICATE

This is to certify that the thesis entitled **Optical Character Recognition System for Telugu Handwritten Text** submitted to **Indian Institute of Technology Madras** by **Saicharan Gontla** for the award of the degree of **Dual Degree** is a bonafide record of the project work done by him under our supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. V Srinivasa Chakravarthy

Project Guide

Professor

Department of Biotechnology

Indian Institute of Technology Madras

Chennai – 600 036.

Prof. K Sridharan

Project Co-Guide

Professor

Department of Electrical Engineering

Indian Institute of Technology Madras

Chennai – 600 036.

Date: 28^{th} June, 2021 Place: Chennai

ACKNOWLEDGEMENTS

Firstly I would like to thank my project advisor Prof. V Srinivasa Chakravarthy for his motivating support and continuous guidance throughout this project. I would also like to thank my co-advisor Prof. K Sridharan for his support during this project.

I would like to thank my friend Vigneswaran for his guidance for this Project. He was always available for clarifying my doubts whenever I required him. I would also like to thank my classmates for their direct and indirect support. Finally, I am extremely grateful to my parents for their support during this Pandemic situation, without whose support this entire project would not have been possible.

ABSTRACT

Telugu is one of the prominent languages in South India. Still, the development of OCR system for Telugu text received little attention. As a result, it is a current research topic. Character recognition systems built in the past were geared for Roman and European scripts. The success rates of OCR systems created for Chinese, Korean, and Japanese scripts are also high. The scripts of Asia and other subcontinents received little attention. Some Telugu characters are made up of multiple connected symbols. Compound letters are created by combining modifiers with consonants, resulting in thousands of different combinations. One or more related symbols can be found in a compound character. As a result, methods designed for documents written in other scripts, such as Roman, cannot be used to Telugu papers.

This report discusses the OCR model, which converts the scanned document image of a Telugu word into machine readable form. These documents can be printed or handwritten documents. This model uses a combination of CNN, LSTM and CTC to convert the image into text. Later, LSTM is replaced by Multi dimensional LSTM which decreases the character error rate further more. First the Model is trained over the Printed datasets and later fine tuned using some handwritten datasets. The character error rate for printed data is 0.94% and for handwritten data it is around 7.48%.

KEYWORDS: Convolution Neural Network(CNN), Recurrent Neural Network(RNN), Long Short-Term Memory Networks(LSTM), Connectionist temporal classification(CTC), Optical Character Recognition(OCR).

List of Tables

4.1 Various Combinations of Compound characters 14
4.2 Results on printed data with LSTM and MultiDimensional LSTM 15
4.3 Results on HW data with MultiDimensional LSTM 17

List of Figures

3.1 2D Convolution Operation with Filter 4
3.2 Max Pooling with 2x2 filter and Stride 2 4
3.3 Structure of CNN 5
3.4 Structure of LSTM Cell 8
3.5 Annotation for each horizontal position 9
3.6 Character Probability 10
3.7 Complete Architecture of OCR 11

4.1 Preprocessing of Image 12
4.2 Vowel Modifiers 13
4.3 Consonant Modifiers 13
4.4 Images with different fonts 14
4.5 Creating the Image of Word from character dataset 16
4.6 Images from Online Dataset 16
4.7 Images from CVIT Dataset 17
4.8 Printed Data Recognized by our OCR System 18
4.9 Handwritten Data Recognized by our OCR System 18

Contents

ACKNOWLEDGEMENTS **i**

ABSTRACT **ii**

List of Tables **iii**

List of Figures **iv**

1 INTRODUCTION **1**

2 RELEVANT WORK **2**

3 ARCHITECTURE OF OCR SYSTEM **3**

3.1 Convolutional Neural Network (CNN) 3

3.1.1 Introduction 3

3.1.2 Convolution in CNN 3

3.1.3 Max Pooling 4

3.1.4 Activation Function 4

3.1.5 CNN in OCR System 5

3.2 LSTM Network 6

3.2.1 Introduction 6

3.2.2 Different Gates in LSTM 6

3.2.2.1 Forget Gate: 7

3.2.2.2 Input Gate: 7

3.2.2.3 Output Gate: 7

3.2.3 Multi Dimensional LSTM 8
3.2.4 LSTM Network in OCR System 8
3.3 CTC Decoder 9
3.3.1 Introduction 9
3.3.2 Encoding the Text 9
3.3.3 Loss Calculation 10
3.4 Complete Architecture of OCR 11
3.5 Summary 11

4 EXPERIMENTATION AND RESULTS 12

4.1 Preprocessing of the Input Image 12
4.2 Training the Model over Printed Data 12
4.2.1 Introduction 12
4.2.2 Generating the Printed DataSet 14
4.2.3 Training and Results 15
4.3 Training Model over Handwritten Data 15
4.3.1 Introduction 15
4.3.2 Datasets 15
4.3.2.1 Telugu Handwritten character Dataset [4] 15
4.3.2.2 Telugu Online Text Data 16
4.3.2.3 CVIT IIIT Hyderabad Dataset [1] 17
4.3.3 Training and Results 17
4.4 Summary 18

5 CONCLUSION AND FUTURE WORKS 19

INTRODUCTION

Telugu is a Dravidian language used mostly by Telugu people in the Indian states of Andhra Pradesh and Telangana, where it is also an official language. Telugu is the fourth most widely spoken language in India, with about 82 million native speakers according to the 2011 census, and ranks 15th in the Ethnologue list of languages by the number of native speakers. Historically, Telugu is a language that evolved from the ancient Brahmi script. It also utilised some of the script creation features from the Dravidian (Pali) language. During it's formation, the text was carved with needles on palm leaves. so, it favoured rounded letter forms.

Significant research efforts have been committed to Optical Character Recognition (OCR) throughout the last few decades . The goal of OCR is to convert human-readable letters into machine-readable codes by automatically interpreting optically sensed document text contents. OCR research is attractive because of the numerous applications it could have in banks, post offices, and defence companies. Reading aids for the blind, library automation, language processing, and multi-media design are some of the other applications.

Despite the fact that Telugu is one of the India's most widely spoken language, Telugu script has only a few OCR solutions. This instilled in us the desire to tackle the situation. The digitization of thousands of printed books in Indian languages by both the private and public sectors adds to the desire to develop a Telugu language OCR. An OCR for both printed and handwritten Telugu text is urgently needed for efficient access to these scanned documents

RELEVANT WORK

The creation of an end-to-end OCR system for the Telugu script has received little attention. While the availability of a large online corpus of scanned texts necessitates the use of an OCR system, the challenge is complicated by the complex script. It's significantly more difficult to create a system that performs well on real-world documents with noise and erasure.

Most of the works in telugu OCR involves segmenting words into basic symbols and detecting it using some Machine Learning or Deep Learning techniques. Lakshmi and Patvardan[3] developed a multi font printed OCR system which gives accuracy more than 92% but it only works for single characters. Atul Negi, B.Krishna[5] gave a practical approach to Telugu OCR using connected components and fringe distance template matching which gives an accuracy of 92%. N Prameela and P Anjusha[6] proposed an OCR System which involves feature extraction and classification based on the features extracted, both support vector machine (SVM) and Quadratic discriminate Classifier (QDA) has been separately used as the classifier. This OCR gives an accuracy of 95.01%. The work presented by Sastry, Panyam Narahari[7] uses zoning features gives an accuracy of 78%.

The previous models gives good results only for an individual character. Since Telugu is a complex script with a lot of curves and complex shapes it is very hard to segment a word into individual characters. This report discusses a model which uses combination of CNN and LSTM. This model directly takes a word as an input and outputs a machine readable text which is different from the previous models which requires segmenting a word into individual characters.

ARCHITECTURE OF OCR SYSTEM

3.1 Convolutional Neural Network (CNN)

3.1.1 Introduction

The CNN model [9] is a form of neural network that allows us to extract higher representations for image input. Unlike traditional image recognition, which requires you to define the image characteristics explicitly, CNN takes the image's raw pixel data, trains the model, and then extracts the features for improved categorization automatically.

3.1.2 Convolution in CNN

A convolution is the basic process of applying a filter to an input to produce an activation. When the same filter is applied to an input multiple times, a feature map is created, displaying the positions and strength of a recognised feature in an input, such as an image.

These convolved features will always change depending on the filter values affected by the gradient descent to minimize prediction loss. The more filters used, the more features CNN will be able to extract. This enables for additional characteristics to be discovered, but at the expense of additional training time.

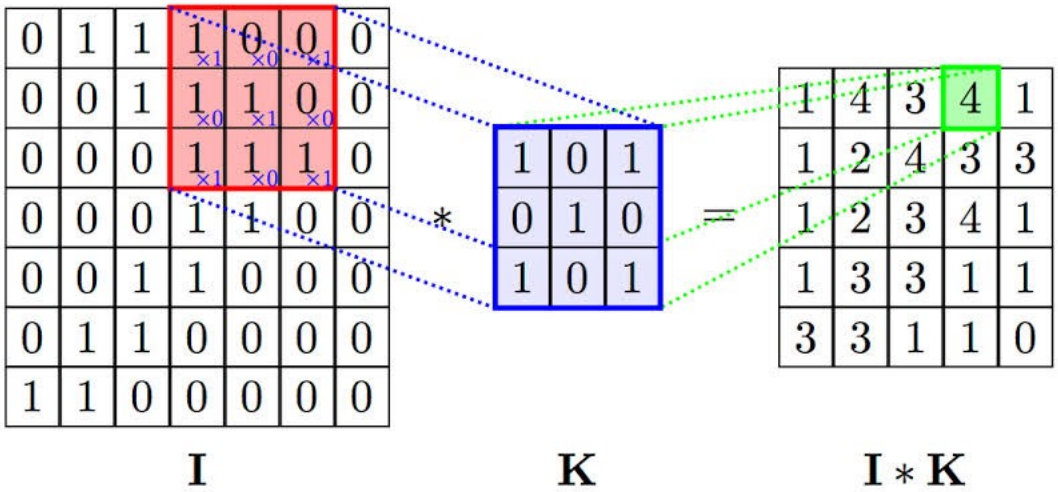

Figure 3.1: 2D Convolution Operation with Filter

3.1.3 Max Pooling

To minimise data space and processing time, CNN employs max pooling to replace output with a max summary. This enables you to identify the features that have the greatest influence while also reducing the risk of overfitting. The Max Pooling layer has two hyper parameters, Stride and Size. The stride determines the value pool skip, whereas the size determines the size of the value pools in each skip.

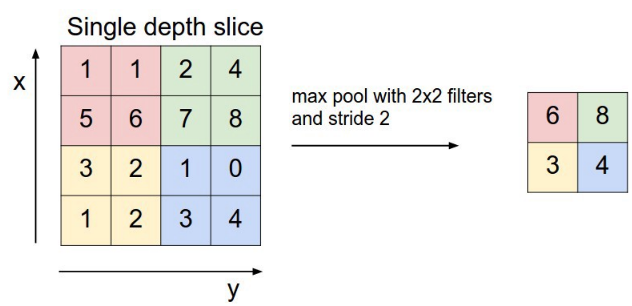

Figure 3.2: Max Pooling with 2x2 filter and Stride 2

3.1.4 Activation Function

After each convolutional and max pooling operation, Rectified Linear Unit(ReLU) is applied on the output. The Relu function mimics the neuron only when the input is greater than zero.

$$\text{RELU}(x) = \text{Max}(0, x) \tag{3.1}$$

3.1.5 CNN in OCR System

The Preprocessed Image is given as an input to the CNN Layers. The CNN Architecture consists of Eight Convolutional layers. In the first layer, the kernel is of the size 7x7 and kernel of size 3x3 in the remaining layers. The Non-linear function Relu is used as an Activation Function. Finally, every Convolutional layer is followed by a Pooling layer which gives a downsized version of input. The Final Output feature map of the CNN has the size of 32x512. This final output is fed to the next part of OCR System.

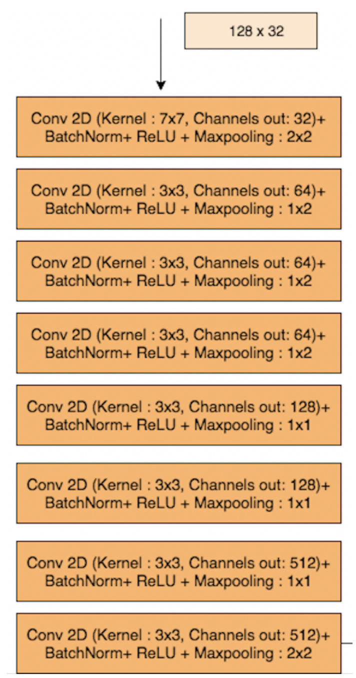

Figure 3.3: Structure of CNN

3.2 LSTM Network

3.2.1 Introduction

Long Short-Term Memory (LSTM) Networks [2] were created to solve the time series problems faced by the by recurrent neural networks RNNs (Vanishing Gradient Problem). LSTMs have feedback connection which make them different from the standard neural networks. This trait allows LSTMs to process whole sequences of data, such as time series, without having to handle each point in the sequence separately, instead preserving important information from prior data in the sequence to aid in the processing of new data points. As a result, LSTMs excel at processing data sequences such as text, audio, and time series in general.

The output of an LSTM Cell is dependant on three things,

- The input data at current time step
- The output of LSTM at previous time step, known as previous Hidden State
- The memory of the cell, known as Cell State

3.2.2 Different Gates in LSTM

LSTMs employ a set of 'gates' that regulate how information in a sequence of data enters, is stored, and exits the network. A typical LSTM has three gates: a forget gate, an input gate, and an output gate. These gates are each their own neural network and can be thought of as filters.

Forget Gate:

Here we will decide which data is useful given the input and previous hidden state. The prior hidden state and the fresh input data are fed into a neural network to accomplish this. This network creates a vector with each element in the [0,1] range (ensured by using the sigmoid activation). This network (inside the forget gate) is taught to output near 0 when a component of the input is regarded irrelevant, and near 1 when it is deemed significant.

Input Gate:

The objective of this stage is to determine what new information is to be added to Cell state given the current input and the previous hidden state. The new memory network is a tanh activated neural network that has learned how to construct a 'new memory update vector' by combining the prior hidden state and incoming input data. The 'input gate' is a sigmoid activated network that functions as a filter, determining which components of the new memory vector should be kept. Because of the sigmoid activation, this network will output a vector of values in the range [0,1], allowing it to operate as a filter via pointwise multiplication. An output approaching zero, just like the forget gate, indicates that we do not wish to update that element of the cell state.

Output Gate:

Now we approach the final step in LSTM Cell. First, The tanh function is applied on the current cell state to get the squished cell state which lies in [-1,1]. The filter output vector is obtained by passing the current input vector and previous hidden state to sigmoid activate neural network. This filter vector is point-wise multiplied with squished cell state to get the final output vector.

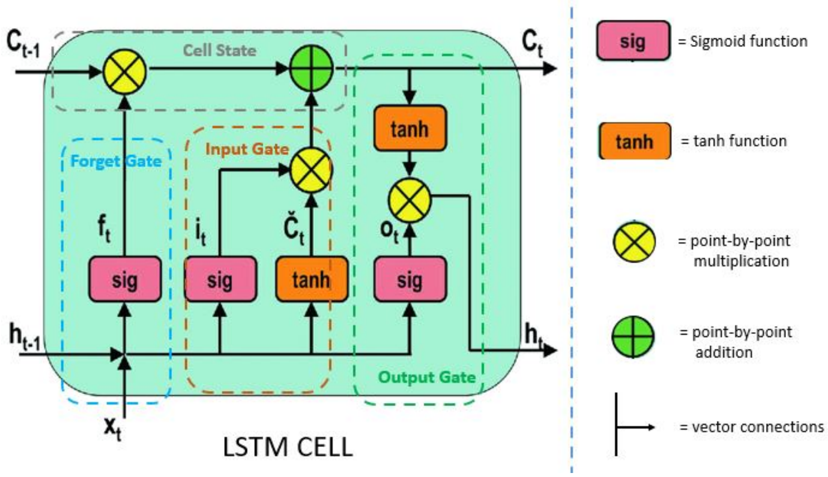

Figure 3.4: Structure of LSTM Cell

3.2.3 Multi Dimensional LSTM

The basic idea of Multi Dimensional LSTM is to replace the single recurrent connection found in standard LSTM with as many recurrent connections as there are dimensions in the data. The standard formulation of LSTM is explicitly one dimensional, since the cell contains a single self connection whose activation is controlled by a single forget gate. However we can easily extend this to 'n' dimensions by using 'n' self connections (one for each of the cell's previous states along every dimension) with 'n' forget gates.

3.2.4 LSTM Network in OCR System

The Output feature sequence of the CNN consists of 512 features per time step is given as input to the LSTM Network. The LSTM Network we used in this OCR System consists of two LSTM layers. The LSTM output sequence is mapped to a output of 32x167 matrix. The dataset constains 166 unique characters and one more character for CTC Blank Label. Thus there are 167 entries for each of 32 time steps.

3.3 CTC Decoder

3.3.1 Introduction

The output of the LSTM is the matrix containing the probabilities of the characters at each time step. From this matrix, we should extract the underlying text. The loss should also be calculated, which can be used to update the weights in neural network. Both these tasks can be acheived using a CTC decoder [8].

Instead of using a CTC decoder we can directly decode the output of LSTM at each time step. But this naive approach can lead to a problem. A Single character can span multiple horizontal positions. From the below example we can see that naive approach gives the output as 'నననటటననన' but the correct output should be 'నటన'.

Figure 3.5: Annotation for each horizontal position

3.3.2 Encoding the Text

In the above example, we can remove the duplicate characters to get the correct output. But what if the desired output consists of a duplicate character? For example, 'నటటన' then just replacing the duplicates is not enough. This issue of duplicates is solved by introducing a pseudo character called as blank('-'). To tackle the duplicate-character problem, we employ a creative coding schema: when encoding a text, we can introduce an arbitrary number of blanks at any place, which will be eliminated while decoding it. However, between duplicate characters we must include a blank. We can also repeat each character as many times as we want.

నటన --> నటటనన or న-టట-నన or న-ట-న

నటటన --> నట-టనన or న-ట-ట-నన or న-ట-ట-న but not నటట-న

3.3.3 Loss Calculation

Loss calculation is necessary to train the Neural Network. The loss calculation is explained here by using a simple example. Let the correct output for the below figure is 'న'. The following example consists of only three characters and two time steps.

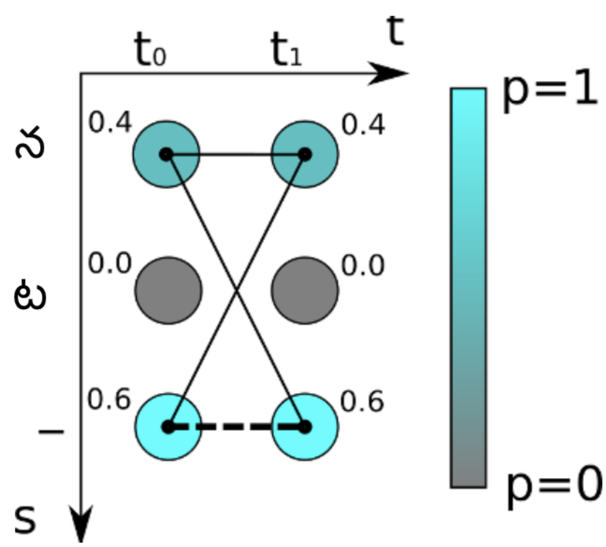

Figure 3.6: Character Probability

In the above example, 'న' can be encoded in three ways. 'న-','-న' and 'నన'. Each way is called a path. The score of each path is calculated by multiplying the corresponding character probabilities. For 'న-' the score is 0.4*0.6 = 0.24, In the same way the score of the other two paths is 0.24 and 0.16. The score of the text is the sum of the scores of all individual paths and Loss is Negative log of the Score of text. For the given example the Total score is 0.64 and the loss is 0.22.

3.4 Complete Architecture of OCR

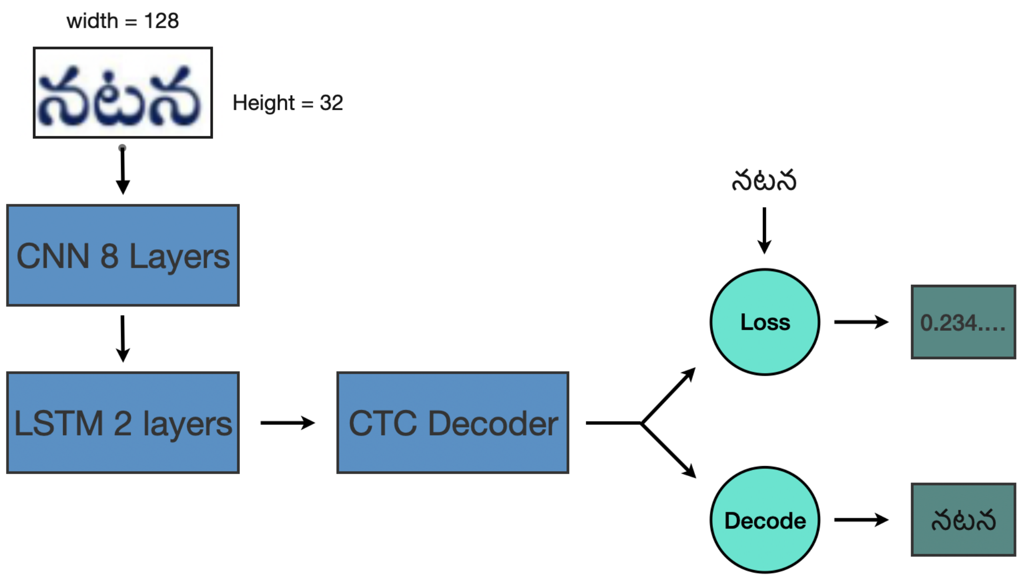

Figure 3.7: Complete Architecture of OCR

3.5 Summary

In this chapter we have seen the complete architecture of OCR. The OCR can be divided into three parts. The first one is CNN which consists of eight convolutional layers. The input to the CNN is an image of size 128x32. The output of the CNN is fed into the LSTM network which consists of two LSTM layers. The output of LSTM network is the Character Probability Matrix (CPM) which is used by the CTC Decoder either for loss calculation or to decode the output. The loss calculated by the CTC is used to update the weights in the model. The entire OCR model is implemented using Tensorflow 1.13.

EXPERIMENTATION AND RESULTS

4.1 Preprocessing of the Input Image

According to the architecture of OCR, The input is the two dimensional (grayscale Image) of size 128x32. Since the words have different number of characters, The images also have different sizes. So, we scale them (without distortion) until they are either 128 pixels wide or 32 pixels tall. The image is then copied into a (white) target image with a size of 128x32 pixels. This operation is depicted in the figure 4.1. Finally, we normalise the image's grey levels, making the task easier for the OCR.

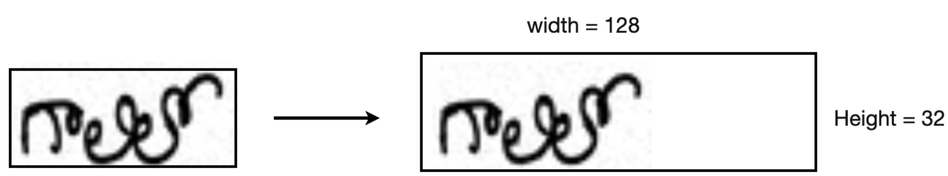

Figure 4.1: Preprocessing of Image

4.2 Training the Model over Printed Data

4.2.1 Introduction

Telugu Language consists of 18 vowels and 36 consonants. Out of 18 vowels, only 16 are in common usage. Each vowel and consonant has an associated modifier called Matra, when this matra is added to a consonant it creates a Different character. The following figures shows the different vowels, consonants and their corresponding modifier. The shaded Modifier in Figure 4.2 is rarely used. The Highlighted Modifiers in Figure 4.3 are more frequently used.

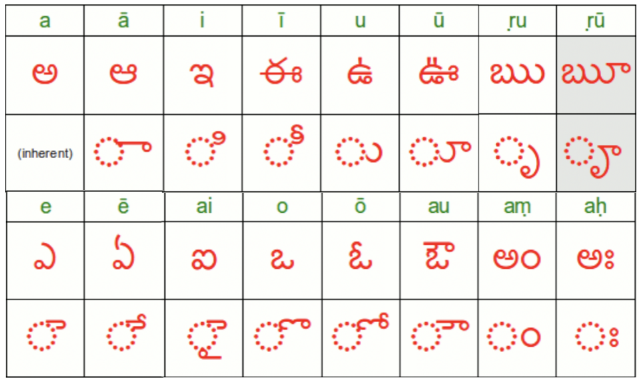

Figure 4.2: Vowel Modifiers

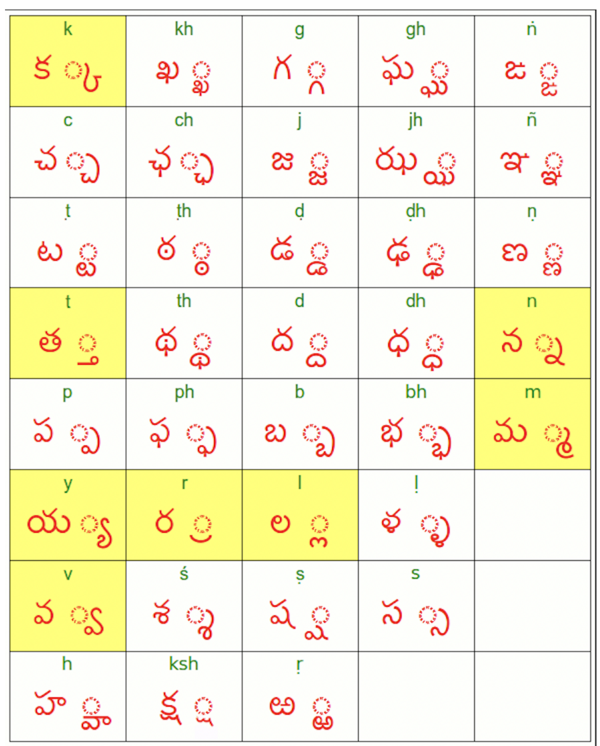

Figure 4.3: Consonant Modifiers

Using these Modifiers a lot of characters are created. The Combinations can be seen in the Table 4.1

S.No	Combination	Examples
1	Vowels	అ, ఆ, ఇ, ఈ, ఉ
2	Consonant	క, ఖ, గ, ఘ, జ
3	Consonant + vowel Modifier	ణం, బ్యా, పూ, ట్యా, యః
4	Consonant + Consonant Modifier + Vowel Modifier	హ్యో, క్షిం, త్లౌ, గ్వ, బ్లం

Table 4.1: Various Combinations of Compound characters

4.2.2 Generating the Printed DataSet

The model should be trained with each and every compound character, Otherwise when the model encounters a new compound character it cannot recognize it correctly. So, all the compound characters are grouped into words of length 4 or 5. Images with all the compound characters are generated using TrueTypeFont (TTF) files. Inorder to make the dataset more realistic, gaussian noise with zero mean and varaince 30 is added to the Images. A slight padding is also added on the four sides. The Figure 4.4 shows the images generated for the same word ('ధైర్యం') using different TTF files.

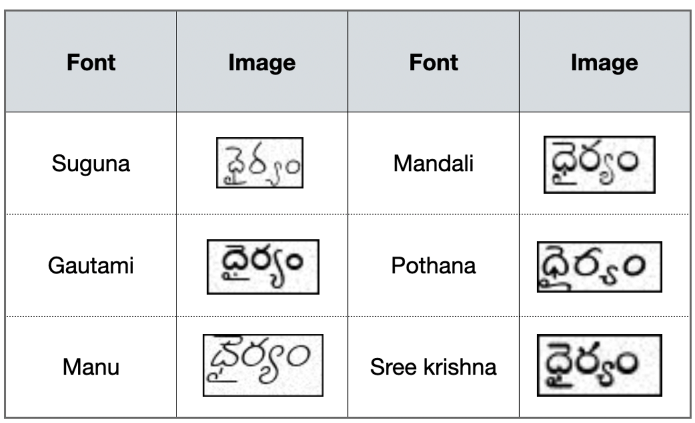

Figure 4.4: Images with different fonts

4.2.3 Training and Results

The model is trained with around 1 lakh images which are generated using the Font Files. The Leaning rate is $1e^{-3}$ and RMSProp optimizer is used. Character Error Rate (CER) is used to measure the performance of the model. The CER is based on the Levenshtein edit distance between the ground truth word and the recognized word. To get the final CER, the edit distance between each ground truth and recognized word was summed for all the words and divided by the total number of characters in all the words. The results for the Test Data are shown in the Table 4.2. The Character Error Rate and Word Accuracy are shown in percentages.

Model Architecture	Character Error Rate	Word Accuracy
CNN + Bidirectional LSTM + CTC	1.41	93
CNN + MultiDimensional LSTM + CTC	0.94	94

Table 4.2: Results on printed data with LSTM and MultiDimensional LSTM

4.3 Training Model over Handwritten Data

4.3.1 Introduction

The Model which is trained over printed datasets is finetuned over the different available handwritten datasets.

4.3.2 Datasets

Telugu Handwritten character Dataset [4]

This Dataset Consists of Telugu vowels, Consonants and Guninthamulu (Consonant + vowel Modifier). This Dataset is created to cover a wide range of handwritten styles. So, it is created from various distinct writers from 18 year old students to 45 year old Teachers. Using the Images of the individual

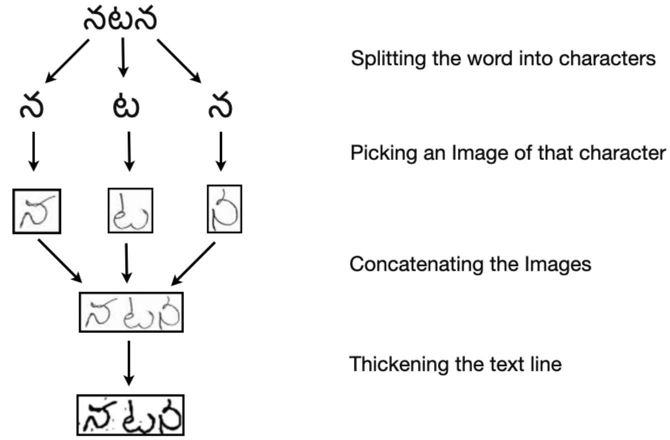

Figure 4.5: Creating the Image of Word from character dataset

characters we can create images of words. First a word is taken and it is split into individual characters using the unicode representation. An Image corresponding to each character is taken from the dataset and Concatenated to create the Image of that word. The thickness of text in the image is increased using a function called 'erode' from OpenCV module in Python. This Process is explained in the Figure 4.5 using an example.

Telugu Online Text Data

This Dataset Consists of Images of Ten thousand telugu words in the form of online data. The online data is converted into Offline. Some of the Images are shown below.

Figure 4.6: Images from Online Dataset

CVIT IIIT Hyderabad Dataset [1]

This Word level Handwritten Dataset is created by IIIT Hyderabad. This consists of more than fifty thousand handwritten words. This dataset is benchmarked using existing state-of-the art methods for HW Recognition. This Images in this dataset are little noisy, which requires a little preprocessing before using them to train the model. Some of the images from this dataset are shown below.

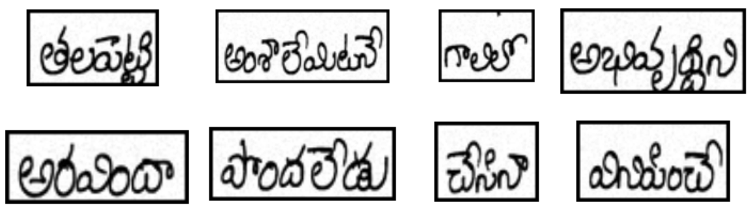

Figure 4.7: Images from CVIT Dataset

4.3.3 Training and Results

The Existing model is finetuned with all the available Handwritten image Datasets. The Leaning rate is $1e^{-3}$ and RMSProp optimizer is used. The Character Error rate for the model with MultiDimensional LSTM is **7.48%**. Figure 4.8 shows the text Recognized by our OCR system from an Image consists of Printed Telugu characters. Similarly, Figure 4.9 shows the text recognized from an image consits of Handwritten telugu characters.

Model Architecture	Character Error Rate
CNN + MultiDimensional LSTM + CTC	7.48%

Table 4.3: Results on HW data with MultiDimensional LSTM

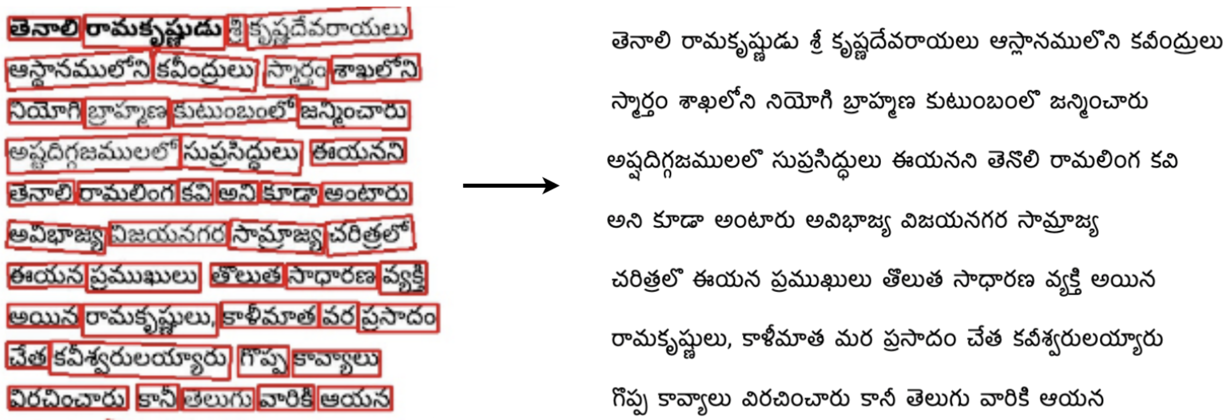

Figure 4.8: Printed Data Recognized by our OCR System

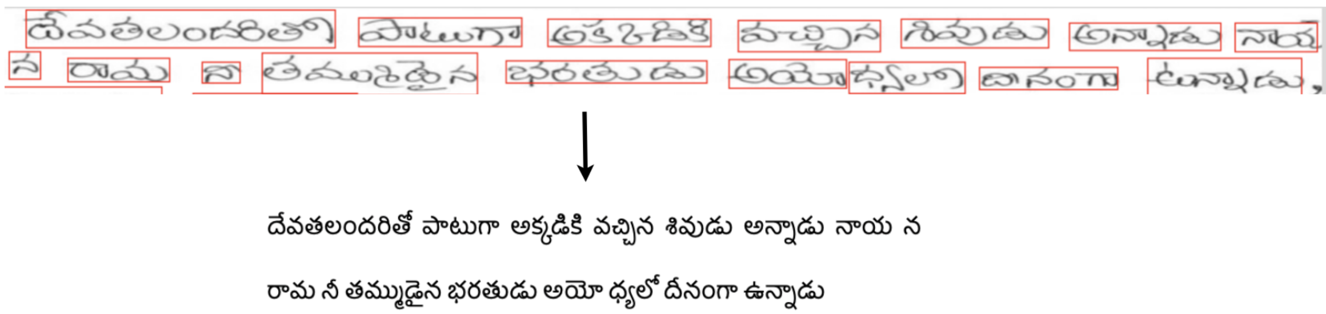

Figure 4.9: Handwritten Data Recognized by our OCR System

4.4 Summary

In this Chapter we have discussed various steps in training the OCR System. First the model is trained over the printed dataset. The results are shown in the table 4.2. Later the LSTM Network in the OCR is replaced by MultiDimensional LSTM, we can observe little decrease in the character error rate. Later, the model is fine tuned over various available Handwritten Datasets.

CONCLUSION AND FUTURE WORKS

The Main aim of this project is to develop an OCR system of Telugu handwritten text. The OCR system which we developed takes the image of a word as input and outputs the text. The OCR system can be divided into three parts. First one is the CNN network which consists of eight convolution layers. The output of the CNN is fed to the LSTM network and the output of LSTM network is used by the CTC decoder to calculate loss and to decode the output. First the OCR system is trained using the printed data which is generated using TTF files and later finetuned using all the available handwritten datasets.

The character error rate, especially for the handwritten text can be further reduced by using more deeper networks and larger handwritten datasets. Better processing techniques and a more powerful CTC decoding algorithm are two further areas of development where considerable gains can be made. Entire OCR model is implemented using Tensorflow 1.13, this model can be updated by implementing using the latest version which is Tensorflow 2.4.

References

[1] CVIT. *Word level Handwritten datasets for Indic scripts*. URL: https://cvit.iiit.ac.in/research/projects/cvit-projects/indic-hw-data.

[2] Rian Dolphin. *LSTM Networks | A Detailed Explanation*. URL: https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9.

[3] C.V. Lakshmi and C. Patvardhan. "A multi-font OCR system for printed Telugu text". In: *Language Engineering Conference, 2002. Proceedings*. 2002, pp. 7–17. DOI: 10.1109/LEC.2002.1182284.

[4] Ravi Kumar M Muni Sekhar Velpuru Tejasree G. *Telugu Handwritten Character Dataset*. URL: https://ieee-dataport.org/open-access/telugu-handwritten-character-dataset.

[5] A. Negi, C. Bhagvati, and B. Krishna. "An OCR system for Telugu". In: *Proceedings of Sixth International Conference on Document Analysis and Recognition*. 2001, pp. 1110–1114. DOI: 10.1109/ICDAR.2001.953958.

[6] N Prameela, P Anjusha, and R Karthik. "Off-line Telugu handwritten characters recognition using optical character recognition". In: *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*. Vol. 2. 2017, pp. 223–226. DOI: 10.1109/ICECA.2017.8212801.

[7] Panyam Narahari Sastry et al. "Telugu Handwritten Character Recognition Using Zoning Features". In: *2014 International Conference on IT Convergence and Security (ICITCS)*. 2014, pp. 1–4. DOI: 10.1109/ICITCS.2014.7021817.

[8] Harald Scheidl. *An Intuitive Explanation of Connectionist Temporal Classification*. URL: https://towardsdatascience.com/intuitively-understanding-connectionist-temporal-classification-3797e43a86c.

[9] Vincent Tatan. *Understanding CNN (Convolutional Neural Network)*. URL: https://towardsdatascience.com/understanding-cnn-convolutional-neural-network-69fd626ee7d4#:~:text=CNN%20is%20a%20type%20of,features%20automatically%20for%20better%20classification..