# Optical Word Recognition system on smartphones for Indic Scripts

*A Project Report*

*submitted by*

**AWANISH RAJ**

*in partial fulfilment of the requirements*
*for the award of the degree of*

**MASTER OF TECHNOLOGY**

**DEPARTMENT OF ELECTRICAL ENGINEERING**
**INDIAN INSTITUTE OF TECHNOLOGY MADRAS.**

**MAY 2016**

# THESIS CERTIFICATE

This is to certify that the thesis titled **Optical Word Recognition system on smartphones for Indic Scripts**, submitted by **Awanish Raj**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

**Prof. Devendra Jalihal**
Research Guide
Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 28th May 2016

# ACKNOWLEDGEMENTS

I express my sincere gratitude to Dr. Devendra Jalihal for giving me this opportunity to work under his guidance. His guidance was instrumental in enhancing my interest on the subject. Throughout the course of my project, he has encouraged me to perform better and has shown faith in my abilities.

I would like to thank Nikhil Naphade, for helping me learn usage of the Caffe library for Neural Networks quickly, and giving me tips for preparation of training data. I would also like to thank Deepak Maurya for helping me with test data collection.

I take this opportunity to express gratitude to all of the Department faculty members for their help and support.

Finally, I would like to thank my parents for their support, trust and encouragement throughout my stay at IIT Madras.

# ABSTRACT

KEYWORDS: Optical Character Recognition; Perspective Transformation; Image Binarization; Word Localization; Convolutional Neural Networks

Optical Character Recognition systems aim at conversion of images of text printed on flat physical media to a digital form, producing machine-readable digital content [Wikipedia (2016)]. Designing an OCR system for Indic scripts is technically difficult due to nature of the scripts and require intricate knowledge of individual scripts by the engineer.

The system presented here takes a Word Recognition approach, to overcome difficulties associated with complicated Indic scripts. The system is incredibly fast, incapable of making spelling errors and has been architected to require no knowledge of the script by the engineer. This comes at a trade-off of a limited vocabulary.

The system is bundled end-to-end as an Android application which provides unique interface capabilities to the user. It also implements a novel Perspective Cropping tool for orientation corrections in the image that are not limited by just affine transformations. Further, innovative Image Binarization and Word Localization algorithms were implemented which are optimized for mobile. Finally, feature extraction for an input image through a Convolutional Neural Network was ported to Android, making the application completely offline from end-to-end.

The Convolution Neural Network has been trained with high frequency words for different scripts, to provide large recognition coverage with significantly lower number of classes. Layers have been designed to keep overall size of the model low, making it ideal for a mobile application.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

Explosive growth of the Internet has made access and distribution of digital information unimaginably fast. The capabilities and applications of digital information continue to transform knowledge into new understandings of human experience and the world we live in. Future of these possibilities are limited only by available research resources and capabilities and the imagination and creativity of those who use them. This rapid growth poses new challenges for research and development in the field of document image analysis. Optical Character recognition is seen as a primary challenge in this field.

The People's Linguistic Survey of India, a privately owned research institution in India, has recorded over 66 different scripts and more than 780 languages in India during its nationwide survey, which the organisation claims to be the biggest linguistic survey in India. Although fairly mature and commercial OCRs for Roman scripts exist, equally capable OCRs for Indian Languages are not yet available. This situation is attributed to the complexity that exists in most Indic scripts. According to Govindaraju and Setlur (2009), there are abugida or alphabetic-syllabic scripts, which can have 1000-1200 unique character shapes in a document of running text. The scripts generally have 50 basic characters. Additionally, there are vowel allographs and consonant compounds with distinct shapes created by the combination of two-, three-, and four-consonant compounds (there is even a five-consonant compound in a now-obsolete word), resulting in a large class space. Further increasing the shape complexity, in a script such as Bangla, some compounds and syllables are printed in more than one form (transparent and non-transparent group of fonts). With the exception of Devanagari, serious attempts at OCR development in Indian scripts by multiple research groups have also been rare.

Apart from challenges specific to Indic scripts, there are general challenges in preprocessing of input images to correct perspective transformations introduced due to camera angles. Further, binarization techniques for the input image for various indoor and outdoor conditions are key to obtaining good recognition. Following binarization, word localization and character segmentation techniques play a major role in design of the recognizer and its training data.

The difficulties mentioned above are major hurdles for most OCR systems that are being developed. But, the proposed approach quickly discards many of the mentioned issues. Instead of the recognition at a character level, we conduct recognition at a word level. This eliminates complexities associated with character combinations (conjunct consonants). It also makes the architecture of the system agnostic to the intricacies of the script, thereby reducing engineering effort in scaling up for multiple Indian Languages.

# CHAPTER 2

# Character Recognition vs. Word Recognition

## 2.1 Issues with Character Recognition

Tesseract is considered one of the most accurate open source Optical Character Recognition engines currently available [Smith (2007)]. It was initially developed for English, but has now been extended to recognize French, Italian, Catalan, Czech, Danish, Polish, Bulgarian, Russian, Greek, Korean, Spanish, Japanese, Dutch, Chinese, Indonesian, Swedish, German, Thai, Arabic, and Hindi etc.

Tesseract is the go-to engine for most Indic OCR research and development. Training the Tesseract OCR Engine for Hindi language requires in-depth knowledge of Devanagari script in order to prepare the character dataset.

Apart from training of this character dataset, training the engine also needs to tackle character segmentation challenges, which are very specific to the script and font being used for training. Different fonts tend to render conjunct consonants in a way, that they don't even seem related when examined by an outsider.

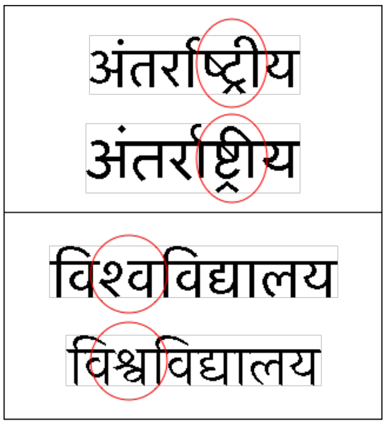


Figure 2.1: Conjunct consonants of Devanagari script complicate character segmentation algorithms

Since the recognition is carried out at the character level, scope for error is significantly higher. This poses a new requirement for a dictionary to identify closest match for a recognized combination of characters. This adds to the computation complexity.

## 2.2 Intuition for Word Recognition

Spelling error in the following passage is deliberate, but is believed to be readable by most people.

Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

Most people read through the above passage without any difficulty. Although no such research was ever carried out at Cambridge University, this example does reveal an important aspect of the way human brain processes words. The brain predicts the entire word based on some feature in the structure of the word, rather than individual characters. The proposed solution uses this insight as the premise and has the following advantages:

- Construction of the word recognition framework is independent of the script being used. Many scripts can therefore be accommodated without extra engineering efforts.
- Eliminates the need for further research in character segmentation techniques.
- Discarding the character segmentation step makes pre-processing significantly faster.
- Inherent dictionary matching significantly improves word level accuracy

# CHAPTER 3

# Android Application Development

The entire engine was implemented end-to-end for Android smartphones.

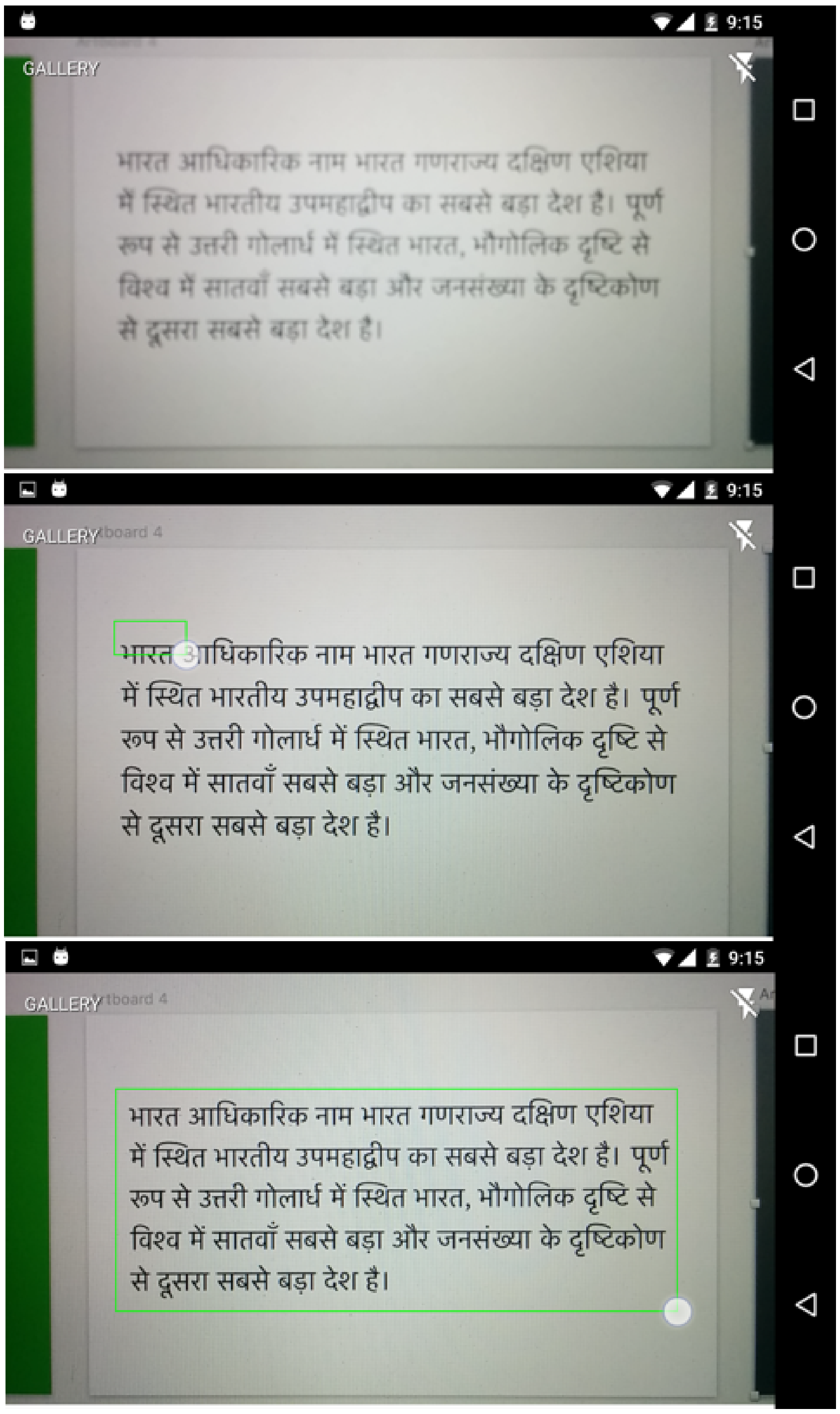
## 3.1 Camera with Feed cropper

Upon opening the app, the user lands directly on a camera feed. In majority of the use cases, the region of interest(region containing text) in the camera feed is a smaller portion compared to the full image. To take advantage of this situation, a unique feed cropping interface was implemented. In this interface, the user performs the following action on a live camera feed:

1. First the user presses on the left top corner of his region of interest. Performing this action triggers an auto-focus call to the camera sensor.
2. Now the user drags his finger to the right bottom corner of his region of interest. A green rectangle tracks the users touch to give a visual feedback for this cropping action.
3. The user now releases his touch and image capture is triggered. The image data received from the camera sensor is the full image. A cropping pass is done to capture only the region of interest from the image.

## 3.2 Perspective cropping

For proper functioning of the recognition engine, the input image is required to have correct orientation with minimal distortion. In real world situations, it is not usually possible to capture a road side sign-board or a poster from the perfect angle. Therefore, most captures are warped by a projective transformation. The correction of these captures are not possible by simple affine transformation on the image. But, correction of such images can greatly increase usability of the application.



भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

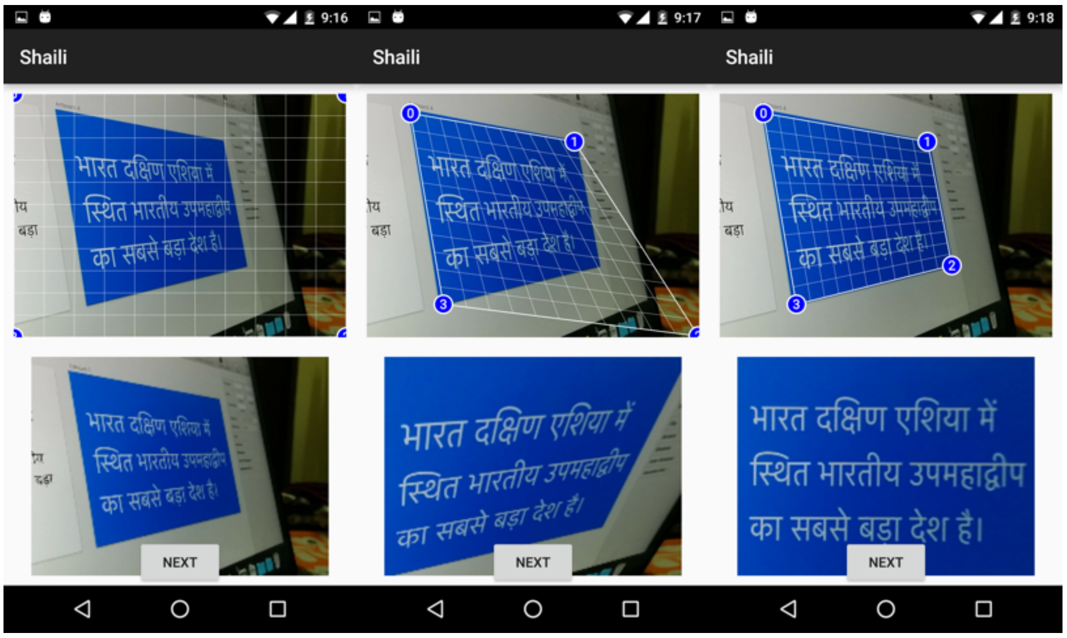
भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

Figure 3.1: The feed is initially out of focus. The user touches the screen and autofocus is triggered. The users touch is tracked by a green rectangle.

A novel perspective cropping interface was implemented for this purpose. The user is provided with a floating quadrilateral over the base image. The corners of this quadrilateral can be moved around by the user. There are no constraints on the position of these corners. The user can now choose a warped region of interest. A transformation is applied on the source bitmap such that these 4 points are stretched out to form a rectangle. The result of this transformation is shown on the screen in real-time. Once the user is satisfied with the perspective correction, he can now choose to proceed with recognition.

Although the aspect ratio information of the region of interest is not recovered, it will later be shown that this information has no bearing on the recognition accuracy. Therefore, aspect ratio correction can be ignored.



9:16
9:17
9:18
Shaili
Shaili
Shaili
भारत दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है।
NEXT
NEXT
NEXT

Figure 3.2: Tool for perspective cropping. The transformation is show in real-time for visual feedback to the user

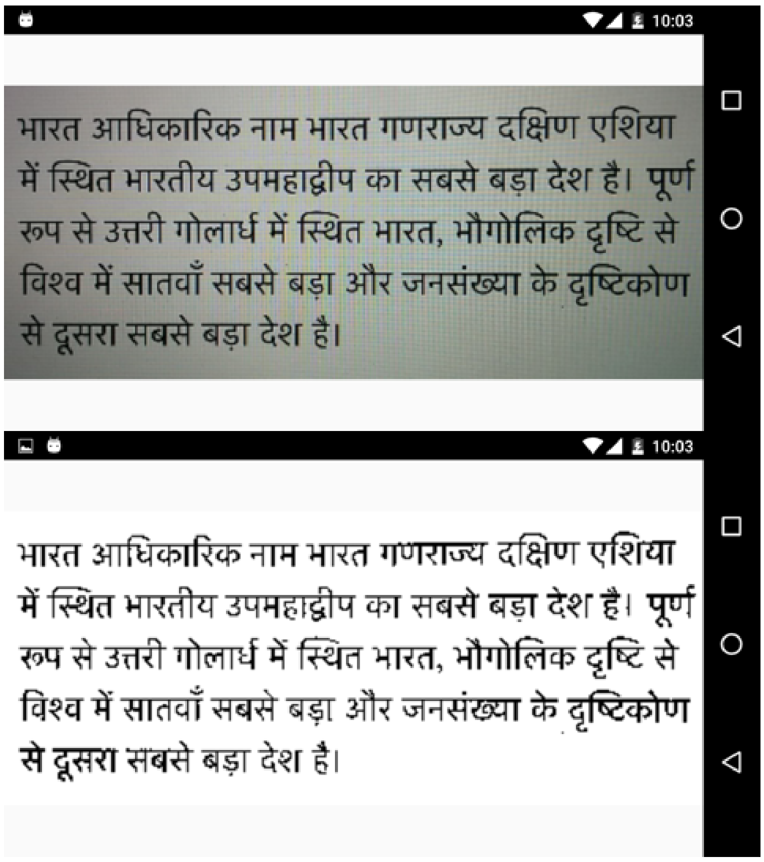
## 3.3 Image binarization

Image binarization is a crucial pre-processing step in any OCR application and has been extensively researched over the years. It according facilitates the following steps in the recognition pipeline, like text localization and word(and character) segmentation.

It is the process of converting a gray scale image into a black-and-white(binary) image. Essentially, pixels that have a gray value above a threshold are classified as white, while the rest of the pixels are classified as black. The motivation for this lies in the fact that most texts are printed to have a contrast between the text(foreground) and the background. They can either have dark text on a light background, or a light text on a dark background. This process of identifying the optimal threshold value that best classifies the foreground pixels from the background pixels is called thresholding.

### 3.3.1 Otsu's method

The algorithm assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels), it then calculates the optimum threshold separating the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal as proposed in Otsu (1975).

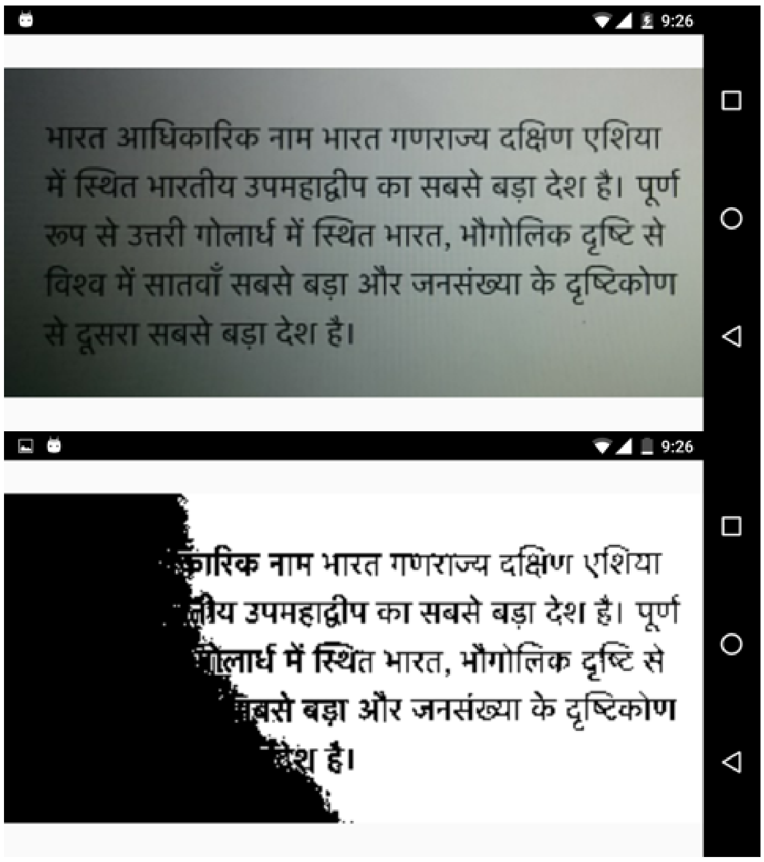


भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

Figure 3.3: Binarization using Otsu's method

This method works gives the best binarization threshold, but its basic assumption that the image contains only 2 modes is often not satisfied. This can be attributed to nonuniform lighting, which introduces a gradient in the background pixels. Because of this, the calculated threshold consumes background portions of the image which received less lighting as black pixels instead of white. This has been analyzed in [Sezgin *et al.* (2004)].



भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

कारिक नाम भारत गणराज्य दक्षिण एशिया
तीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
बसे बड़ा और जनसंख्या के दृष्टिकोण
देश है।

Figure 3.4: Otsu's binarization fails due to variation in illumination

One famous and effective improvement suggested over this method is the two-dimensional Otsu's method. In this approach, the gray-level value of each pixel as well as the average value of its immediate neighborhood is studied so that the binarization results are greatly improved. But even with dynamic programming approaches, time complexity of this modified method is large.

### 3.3.2 Adaptive thresholding using Integral Image

This method, often called Bradley's method, aims for superior performance but with significantly faster processing by limiting number of iteration over the image. The algorithms has presented extra-ordinary real-time performance as described in Bradley and Roth (2007). This algorithm is based on research on threshold selection using clustering criteria [Kittler and Illingworth (1985)].

In this method, each pixel is compared to the average of a window of pixels surrounding it. Doing this kind of adaptive thresholding, takes into account spatial variations in illumination. It fixes the problem observed in Otsu's method, with a faster and simpler algorithm.

The speed of the algorithm can be attributed to the use of integral image technique. An integral image (also known as a summed-area table) is a tool that can be used whenever we have a function from pixels to real numbers f(x,y) (for instance, pixel intensity), and we wish to compute the sum of this function over a rectangular region of the image. Without an integral image, the sum can be computed in linear time per rectangle by calculating the value of the function for each pixel individually. However, if we need to compute the sum over multiple overlapping rectangular windows, we can use an integral image and achieve a constant number of operations per rectangle with only a linear amount of preprocessing.

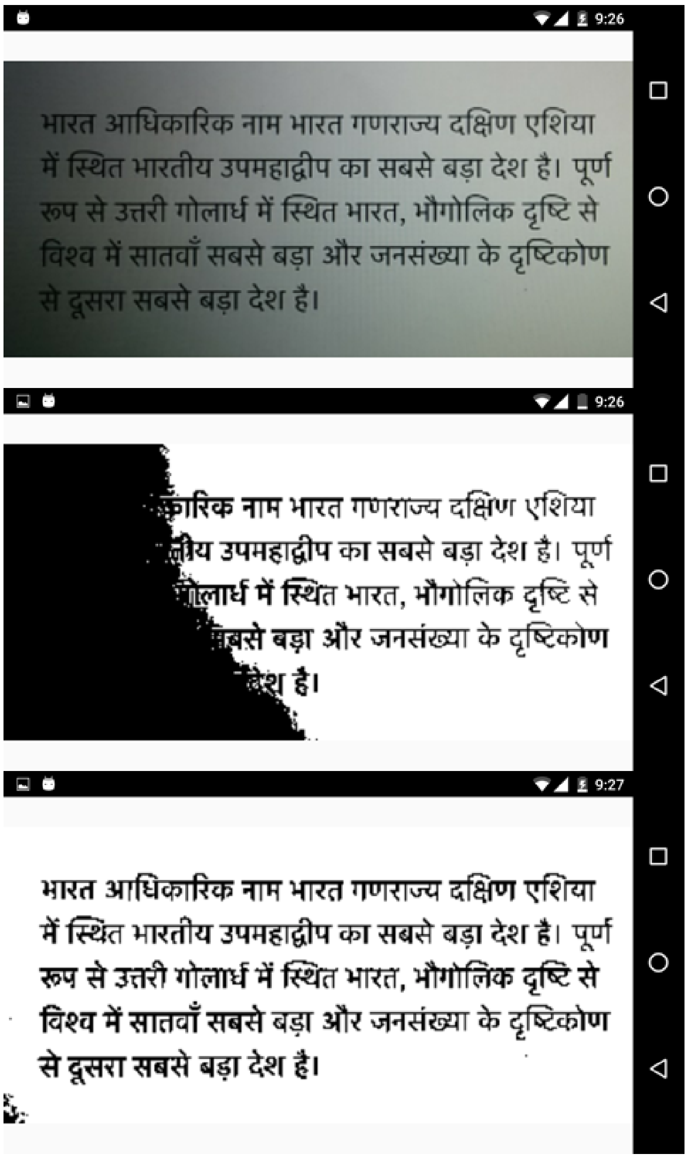
But this method also involves a threshold coefficient which is suited for either light text on dark background or dark text on light background. The algorithm doesn't provide a way to judge the kind of foreground-background combination, and thus can work for only either of these cases. An improvement to this algorithm has been made to tackle this situation.

### 3.3.3 Modified adaptive thresholding

The aim is to modify the adaptive thresholding algorithm so that we achieve successful binarization in both of the following cases:

- Dark text on light background
- Light text on dark background

We need to make a decision from the input pixel data and classify the image into one of the above types, so that the corresponding threshold coefficient can be chosen. To achieve this, an average of all the pixels in the grayscale image is computed. A lower value(below 50% of max pixel intensity) of this average implies that the overall image

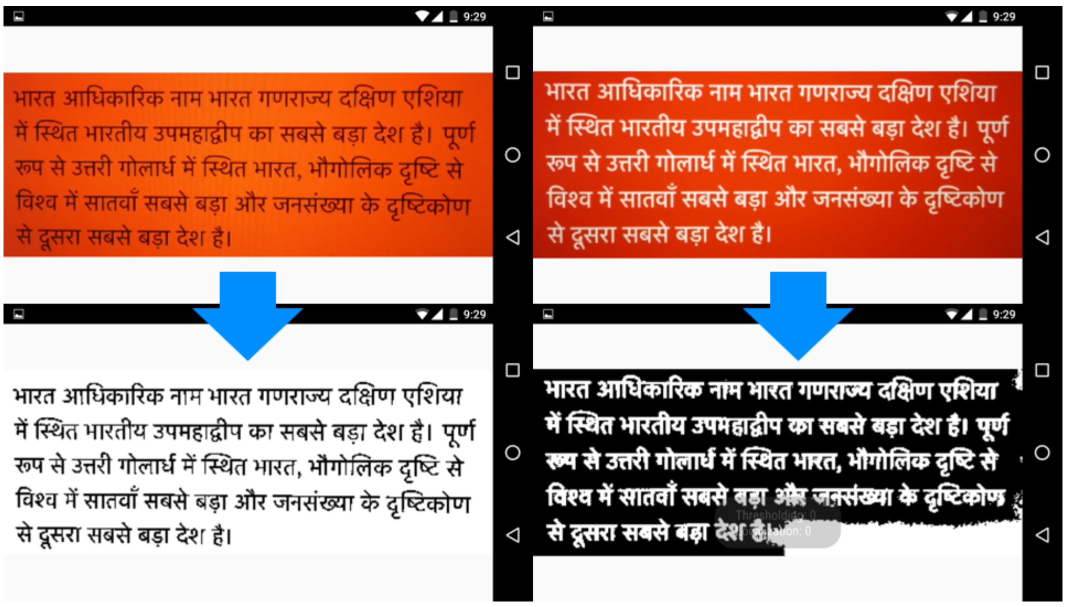


भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

कारिक नाम भारत गणराज्य दक्षिण एशिया
तीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
बसे बड़ा और जनसंख्या के दृष्टिकोण
देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

Figure 3.5: Adaptive thresholding technique compared with Otsu’s method



भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

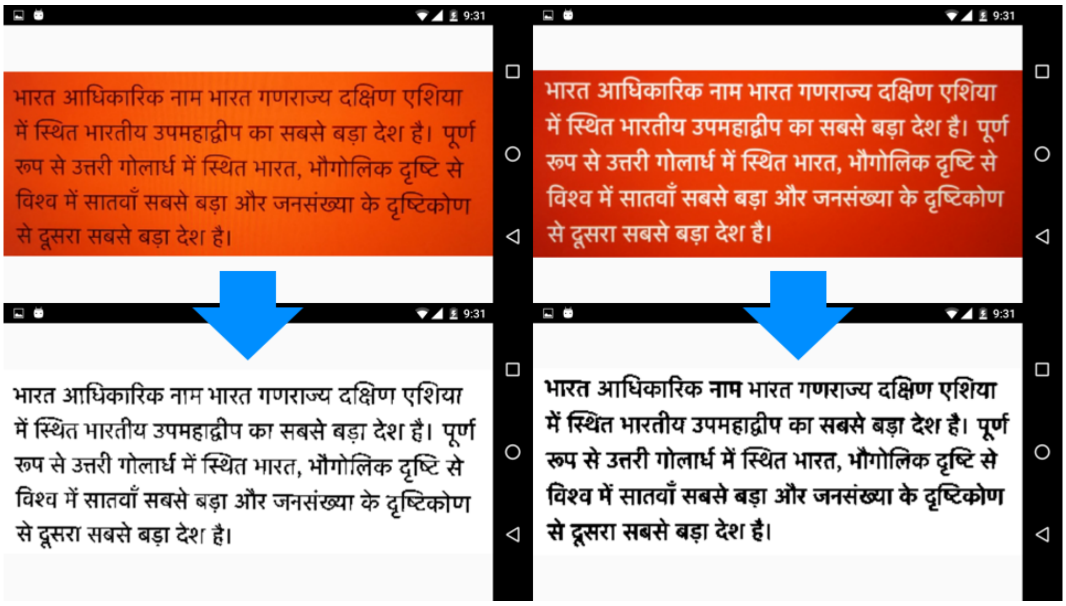
Figure 3.6: Failure of Adaptive thresholding when foreground pixels are lighter

is darker, which implies that the background pixels are darker than foreground pixels. Similarly, a higher value(above 50%) implies that the background pixels are lighter.

But, the classification at an average value 50% intensity works well only when the image histogram is normalized i.e. changing the range of the pixel intensity values to improve the contrast. This makes light pixels lighter and dark pixels darker, so as to create a high contrast image. The image can now be better classified by computing the average value of this normalized image and comparing to 50% intensity.

These 2 modifications of histogram normalization and average value based classification were integrated into Bradley's method without any additional iteration over the image. Therefore, this modification has no bearing on the speed of the algorithm.

Another less significant case of a composite image, where part of the image is of first type and another part is of the second type, was studied. It was found that computing a more local average (significantly larger than the thresholding window but smaller than the image dimensions), can identify the two types of regions and choose corresponding threholding coefficients dynamically.


भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण से दूसरा सबसे बड़ा देश है।

Figure 3.7: Modified Adaptive thresholding achieves good results with both kinds of images

## 3.4 Word localization

After a successful binarization, the only step left before recognition is word localization. This involves identifying bounding boxes around words on a binary image. After this, subimages of just the word can be cropped out and forwarded for recognition.

### 3.4.1 Connected component analysis

Connected-component analysis is an algorithmic application of graph theory, where subsets of connected components are uniquely labeled based on a given heuristic. In binary images, this helps in detecting connected regions [Samet and Tamminen (1988)]. This is specially useful for scripts like Devanagiri because of the Shiro-rekha, that connects all characters and therefore words are very easily identified by the algorithm. This method does not complete the word localization process, because in many cases, there may be some component in the word which is not connected to the major connected component. For example, in Devanagiri some conjunts that occur at the beginning of the word may not be connected directly to the word while still being a part of it. Therefore, two more steps were added after connected component analysis to achieve proper word localization.

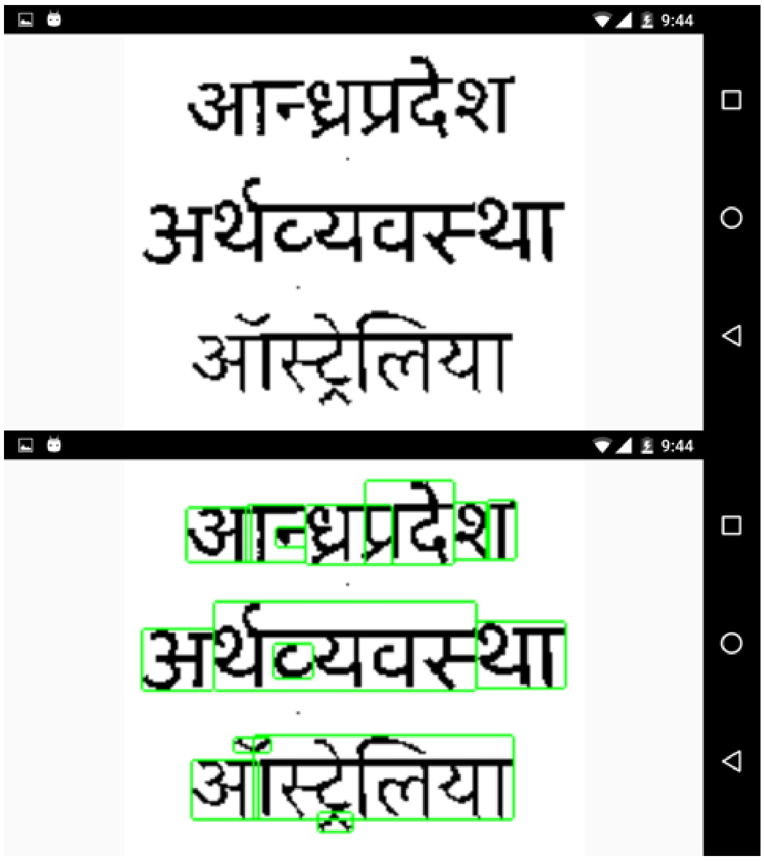


Figure 3.8: Bounding boxes obtained after Connected Component labelling

### 3.4.2 Bounding box overlap pass

One peculiar nature of words is that two disconnected components must be part of the same word if they have overlapping bounding boxes. Therefore, in this pass of the algorithm, we compute all such overlaps and merge them into larger bounding boxes containing the word. The localization is still not complete as it misses out on maatras like the chandrabindu which may not overlap with the base word, but must be merged with the base word.

### 3.4.3 Vertical and horizontal proximity pass

In this step of the algorithm, we identify boxes which are in close proximity to each other. If this distance is below a range factor, the boxes undergo a merge. The range factor has been tuned separately for both vertical and horizontal distances. This is because a large range factor for horizontal distances can lead to mergin of separate words. A slightly larger vertical range factor on the other hand, successfully merges the

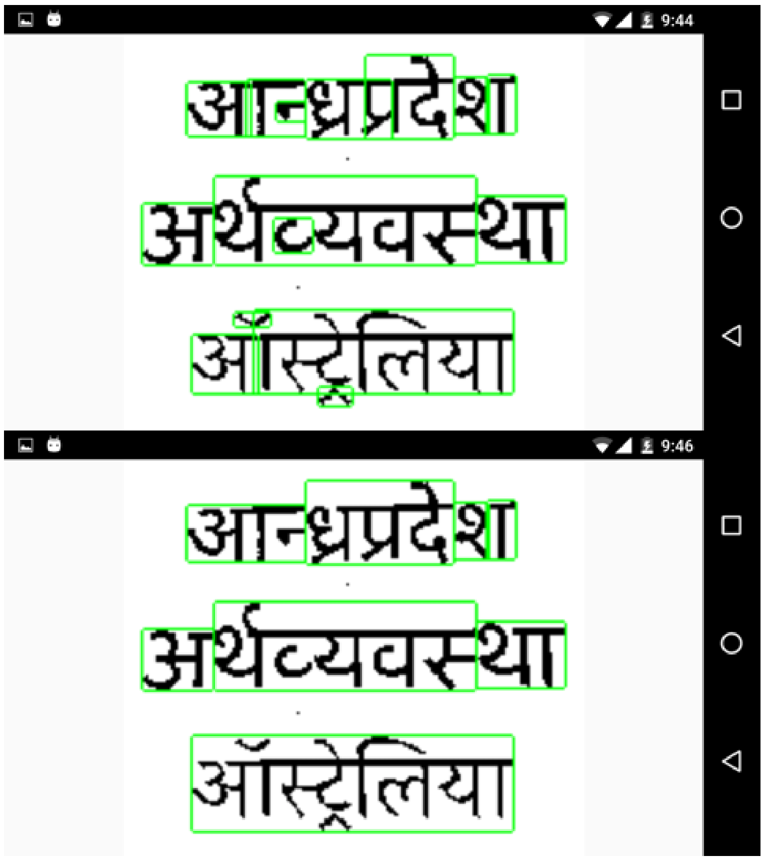
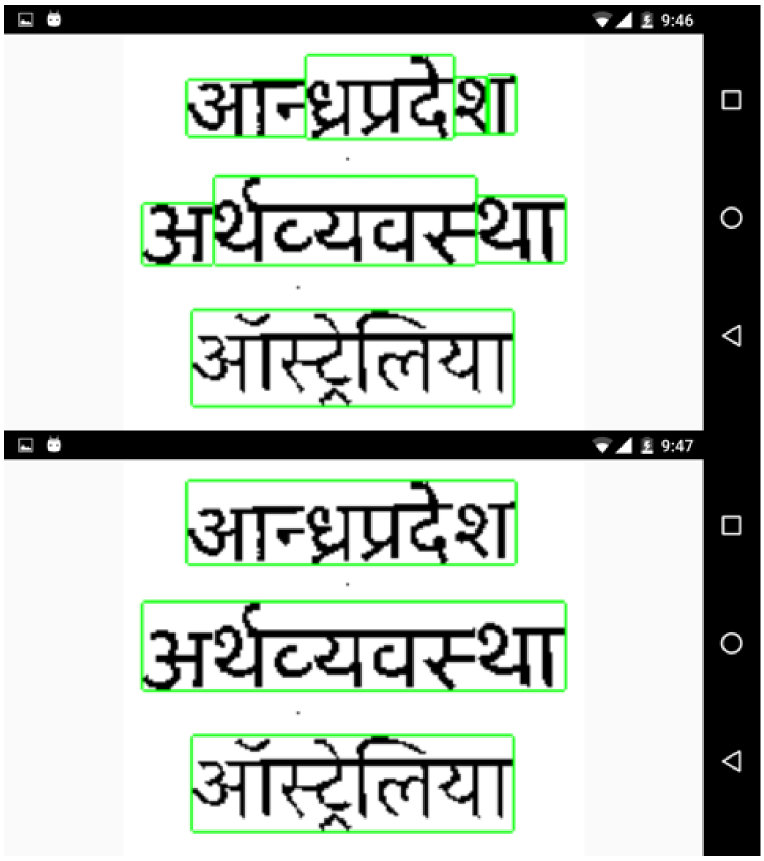


Figure 3.9: Bounding boxes after overlap merging pass

maatras like the chandrabindu with the base word.

### 3.4.4 Ordering of subimages

Now that we have cropped subimages of the word from the original image, we are ready for recognition. But, the sequence in which these subimages is crucial for the sentence to remain coherent. This is achieved by a sentence identification algorithm. It looks at the positions of the words as identified previously and indexes them with a sentence label. This is achieved by identifying overlapping horizontal projections of these words. Each set of overlapping projections represents a unique sentence. These identified sentence sets are ranked based on their distance from the top of the image. Now, second level ranking needs to occur withing each of these sentence sets. For this, distance of each word in the set from the left edge of the image is computed, and the words are ranked accordingly. By assimilating these 2 ordered lists of sentences and words in a sentence, we obtain a master list of words that are ordered in the correct sequence.


आन्ध्रप्रदेश
अर्थव्यवस्था
ऑस्ट्रेलिया

आन्ध्रप्रदेश
अर्थव्यवस्था
ऑस्ट्रेलिया

Figure 3.10: Bounding boxes after merging boxes in close proximity

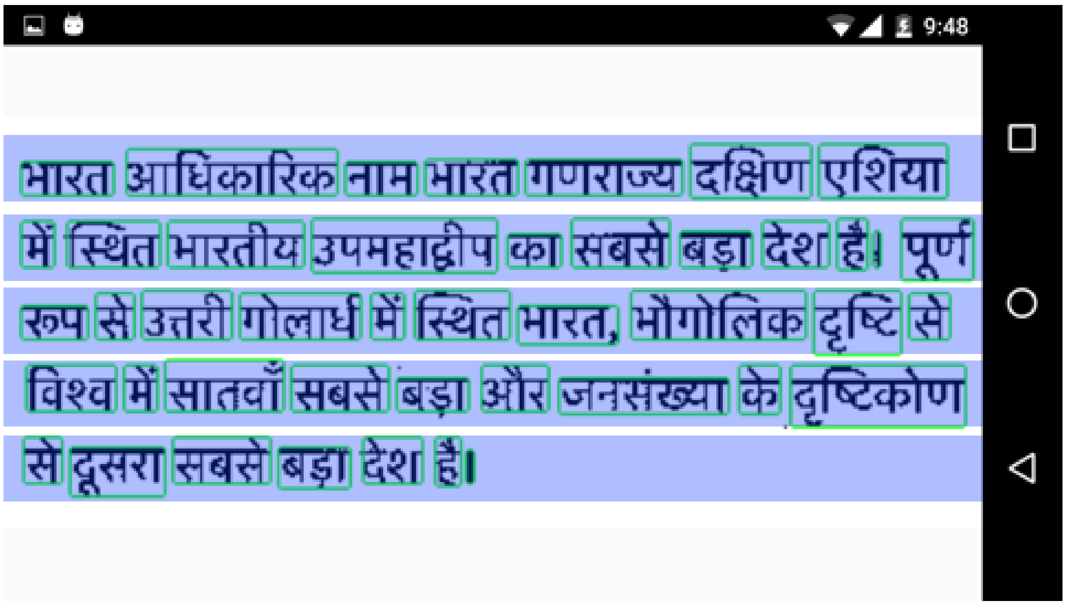

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

Figure 3.11: Identification of sentences by horizontal projection of bounding boxes

## 3.5 Image classification

Image classification is the final step in the recognition pipeline. In this step, each of the cropped subimages are forwarded through the neural network. The image classification is done entirely on the android device and takes all the credit for keeping the recognition process completely offline.

The neural network was designed and trained on a powerful GPU server with the help of the Caffe library. Caffe is a deep learning framework with speed and modularity at its core. The library has been ported to Android in open-source efforts.

There are two aspects of this library, namely training and execution. Training has been discussed in detail in a later chapter. The second aspect of execution is carried out on an Android device and has been discussed below.

### 3.5.1 Requirements for execution

The android library requires three files to prepare itself for the image classification task on Android. These files are products of the training stage. They are described below:

1. Deployment ProtoTxt file
   This is protocol buffer definition file that defines the model architecture of our neural network. It consists of information of each layer and the connections between these layers.
2. Weights (CaffeModel) file
   This is a binary protocol buffer file that consists weights for all the connections in the neural network. These weights are learned by the caffe framework in the training phase.
3. Mapping (Classes) file
   This file consists of all the labels that we have assigned to each class. The classification process results in a class index for the given input. This class index is mapped to a more desirable and human-readable label with the help of this mapping file.

### 3.5.2 Network initialization

When the recognition program is executed, network initialization is the primary step taken by the library. In this step, the library inflates the neural network as described in the ProtoTxt file. After a successful inflation, all edges of the network are initialized with the weights stored in the Weights file. The network is now ready to accept input images and perform classification. Additionally, the library also creates a lookup table of the class indices and their labels. This allows it to return a the word in machine encoded format from the detected class index.

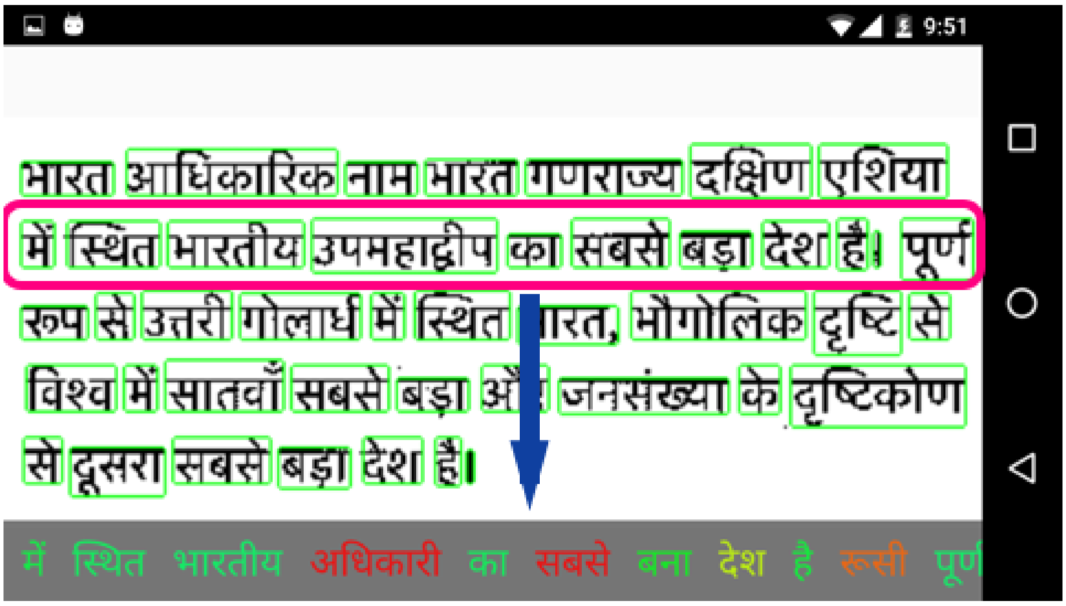
### 3.5.3 Input forwarding

In this step each of the cropped subimages are forwarded through the neural network in the correct sequence. The library accepts one image as input and returns an array of confidence scores corresponding to each class. Further wrapper methods were written to identify the class with the highest confidence score and appropriate label is picked up from the mapping of classe indices and labels. The result is returned to the calling method.

## 3.6 Result visualization

At the end of the recognition pipeline, a list of result objects are obtained. Each result object consists of a label, in this case the 'word', and associated confidence score. To present this result, a colour coded formatting was applied. Words rendered in green have high confidence score, whereas those rendered in red have low confidence scores.

A gradient of colours from green to red through yellow was selected. Words with confidence score of 100% were mapped to green, and those with score of 50 or lower were mapped to red colour. Compared to plain text format, this new representation is more insightful without any clutter.



9:51

भारत आधिकारिक नाम भारत गणराज्य दक्षिण एशिया
में स्थित भारतीय उपमहाद्वीप का सबसे बड़ा देश है। पूर्ण
रूप से उत्तरी गोलार्ध में स्थित भारत, भौगोलिक दृष्टि से
विश्व में सातवाँ सबसे बड़ा और जनसंख्या के दृष्टिकोण
से दूसरा सबसे बड़ा देश है।

में स्थित भारतीय अधिकारी का सबसे बना देश है रूसी पूर्ण

Figure 3.12: Output in colour coded format

# CHAPTER 4

# Convolutional Neural Network training

## 4.1 Convolutional Neural Networks

Convolutional Neural Networks are a type of feed-forward artificial neural network. Their design is inspired from biology. Research was done on the visual cortex of animals, which is the most powerful visual processing system known to us. The visual cortex consists of complex arrangement of cells which respond to small sub-regions in the visual field. These sub-regions are called receptive fields. These sub regions are tiled to cover the entire visual field.

From a training point of view, the current frameworks for implementing CNNs don't require the engineer to design the feature detectors. Training algorithms require a training dataset which then processes these images through an approximately initialized network. As the training algorithm forwards an image through the network, it uses backpropagation techniques to modify the weights of this network such that classification becomes more accurate. It does test the network periodically through the training process. If the training converges successfully on the given training and test dataset, the weights of the latest iteration are dumped into a weights file. The network can now be inflated on any device with these weights, and it will be ready to classify input images.

In this application, we are looking at classifying images of words. Each word in our desired vocabulary will be a class for this network. The training dataset must consist of images for each word with variations, such that the network correctly learns only the important features across these variations.

## 4.2 Caffe library

Caffe is a deep learning framework which has speed and modularity at its core. It is developed by the Berkeley Vision and Learning Center (BVLC) and community contributors. It is one of the most popular libraries for Convolutional Neural Networks.

Its clean architecture attracts many enthusiasts to this field. The framework has been designed to work with GPUs which is especially crucial to fast-track the training process. NVIDIA's CUDA is a parallel computing platform and programming model that has become very popular especially for this purpose. Same training on a CPU would be much slower in comparison.

First task is to setup this framework and its requirements on a system. It is best to use a server system for this case for 2 reasons. There are server instances available with much superior GPUs, and since these servers dont have a monitor display, there is no GPU penalty to slow down the training process. Amazon EC2 Server (EC2 - Amazon Web Services) is a very good example of such a server which is affordable. It is Linux GPU instance which provides access to NVIDIA GPUs with up to 1,536 CUDA cores and 4 GB of video memory. Further this server must be installed with requirements of Caffe namely - NVIDIA Drivers for CUDA, OpenCV, Lightning Memory-Mapped Database (LMDB), and the Caffe library itself.

Once the requirements are satisfied, a sample network was trained and tested to ensure proper functioning of the framework. We can now begin to prepare our own training data and train the network.

## 4.3 Corpus analysis

Now that we are ready to prepare the training dataset, we must make a choice of vocabulary for training. Since each word is a separate class for the neural network, a smaller set of words can result in smaller models that can then be executed on even mobile devices which are low on resources. For this purpose, we analyze the corpus for Hindi to obtain this set of words which best represent the language.

While there a few corpus available for Hindi, many of them aren't very extensive. Some of the good ones are biased, because they are regarding certain topics. They are also prone to many errors which hinders proper analysis of the corpus. A more reliable and ever-growing corpus is Wikipedia's database of Hindi articles. They are managed by a community which ensures correctness and topics vary across fields. Wikimedia makes datadase dumps of all articles for a specific language available for download as a single dump file. These dumps are updated from time to time as more articles are added.

Once downloaded, the dump file which is xml format can be extracted as article files. Further these article files are parsed into words. Analysis of the database dump from 1st May 2016 reveals that there are 2,20,20,425 total words in the database. In this list, many words are repeated several times. Further analysis to identify unique words shows that there are only 5,53,973 words that make up all the Hindi articles on Wikipedia. This list of words was sorted based on frequency. We now pick up the top most frequently occurring words to ensure maximum coverage of the articles. It was found that mere 2,101 words cover 75% of all these articles. Such high coverage with such small set of words is an interesting nature for all languages, where a small set of words, make up for the bulk of the literature.

The choice of number of words was made based on a trade off between coverage of literature and size of the model that will be trained. The model size can be approximately calculated as follows:

Size(in MB) = 22 + (0.002 x NO_OF_CLASSES)

This equation depends on the network architecture chosen and are applicable only for this particular network that was used. The following table illustrates how model sizes would vary with the number of words that we select.

Table 4.1: Table of model sizes for different sizes of the vocabulary

| Number of words | Model Size (MB) |
|:---:|---:|
| 10 | 22.02 |
| 100 | 22.2 |
| 1000 | 24 |
| 10000 | 42 |
| 100000 | 222 |
| 1000000 | 2022 |

It was realized that 2000 words results in a model size of 26 MB which is small enough for execution on an Android device. Apart from these high frequency words, we also need more classes to represent the 10 digits of Hindi, and 8 punctuation marks. Finally, the 2,101 words plus the 18 extra classes were chosen, setting the number of classes at 2,119. Training with this resulted in a model size of 26.4MB which provides 75% coverage of the corpus.

## 4.4 Training images' generation

With this list of 2119 words decided, generation of training images is the next task. Preparation of the training set is crucial for the CNN to learn appropriate features and is responsible for improving confidence of the network in cases of slight variations for real world inputs. Therefore, variations are introduced while generating these synthetic images. The following sections discuss the variations that were introduced and the rationale behind them.

### 4.4.1 Rotation

Most real world images that are captured on a mobile device have slight rotations in them. Lot of research has been done on straightening of such rotated images, but no rotation can be accurately corrected. Therefore we need to teach the network to ignore slight rotations that may appear in the image of the word. This was done by generating 5 different rotations of the training image. Each was rotated at -4°, -2°, 0°, 2° and 4°. After the rotation pass of the image generation, our number of images becomes 5 times.

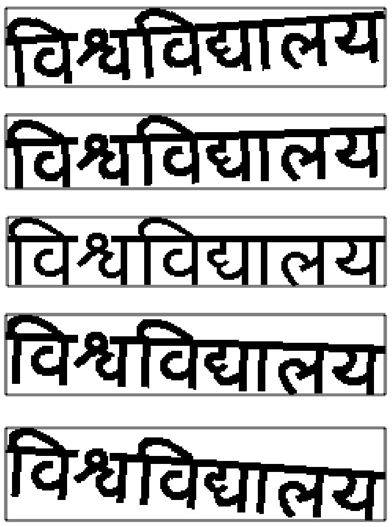


Figure 4.1: Introducing rotations in the training set

### 4.4.2 Salt and Pepper noise

Most of the captured images may have noise that don't get filtered, and the binarization performed on such images can result in some salt and pepper noise in the binarized image. This basically appears as scattered black or white pixels on the image. To teach the network to ignore such pixels, we introduce salt and pepper noise of varying densities on the training image. For this application, we chose 6 different noise probabilities equally spaced between 0 and 0.01. After the noise introduction pass, number of images is increases by a factor of 6.

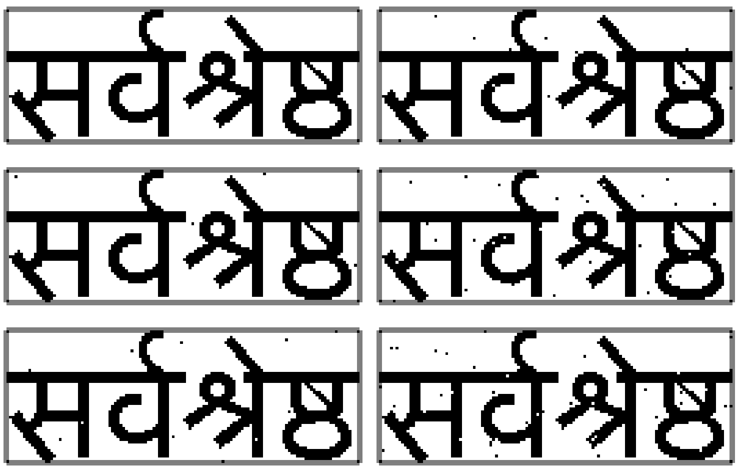


Figure 4.2: Introducing salt and pepper noise in the training set

### 4.4.3 Font variations

Different fonts render characters and conjunct consonants in different ways. But the overall structure of the word can still be recognized. To teach the network about those structures in the word that are irrelevant, we vary the font used for the training set. For Hindi, 4 different fonts were chosen, namely - Mangal, ITFDevanagari, NotoSansDevanagari and Kohinoor, for each word in the vocabulary. Although more fonts exist, these fonts have distinguishable effects on the structure of the image and the few popular ones in use. After this Font variation pass, size of our training set increases by a factor of 4.

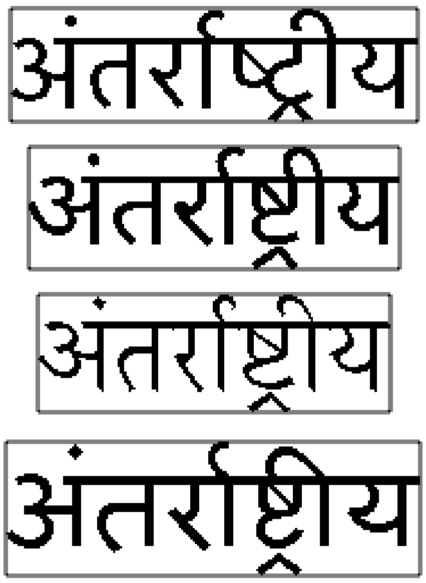


Figure 4.3: Introducing font variations in the training set

## 4.5 Compilation of training images

After applying all the variations discussed previously, the size of the training set stands at

2119*5*6*4 = 2,54,280

The Caffe library requires the training images to be compiled in one of the database formats. One such format is LMDB which stands for Lightning Memory-Mapped DataBase. It uses memory-mapped database to provide much faster I/O performance during the training process. It is works really well with larger datasets compared to other formats.

Also, the caffe library requires all input images to be of the same size. There is an important decision required at this point. Resizing all images to a specific size can be done in two ways.

- Padding the image to obtain the size
  In this case whitespace is added around the word to get a specific image dimension. This can affect training in 2 possible ways. First, the network may start learning that the whitespace is also necessary information for classification, which is clearly not the case. Also, the aspect ratio of the input image will be crucial for classification, because the network is trained with the natural aspect ratio. But, in real world cases, the text may not be of natural aspect ratio. This kind of training also poses a new pre-processing step in the android application. After the word localization pass, the application must now perform padding on

the images. It also requires the image that the application to correct the aspect ratio of the image.

- Stretching the image to obtain the size
  When the image is stretched vertically and horizontally, there are no whitespaces in the final image. Now that the word is stretched across the new canvas, the network will only learn the relevant features, and no whitespace is present to be learnt. Another advantage is on the application side. Since the aspect ratio information is lost in this kind of resizing, input images also don't need any aspect ratio correction. Whatever be the aspect ratio of the image in the crop, it is just stretched to the desired dimensions. This can be considered analogous to a normalization process with respect to the image dimensions

Looking at the advantages of stretching the image, we decide to proceed with resizing the training images to 100x50 dimension. Width has been kept higher to accomodate lengthier words.

The Dataset compilation script resizes the images to 100x50 dimension and creates memory-mapped database in LMBD format. These LMDB files can now be used by the Caffe framework for training.

## 4.6 Network architecture

Several network architectures exist in this field of Convolutional Neural Networks. One such network architecture is LeNet which was developed by Yann LeCun during the 90's [LeCun *et al.* (2015)]. Since then this architecture has been found to work very well for zipcodes and barcodes. One such architecture was developed for the MNIST database. MNIST is a large database consisting of images of handwritten dataset that is commonly used for training many image processing systems. The database consists of 60,000 training images and 10,000 testing images. The LeNet architecture for this MNIST database works really well. It also has a smaller storage footprint which results in a small model size.

Looking at the small model size and good performance for characters, this architecture was the first to be attempted. There were modifications made to the original architecture. Input image dimensions were changed to support 100x50. Further, the last fully connected layer was modified to 2119 classes.

Training with the training images always converged quickly and had very good accuracy. Seeing as how the performance was satisfactory for practical purposes, this architecture was adopted for the final version.

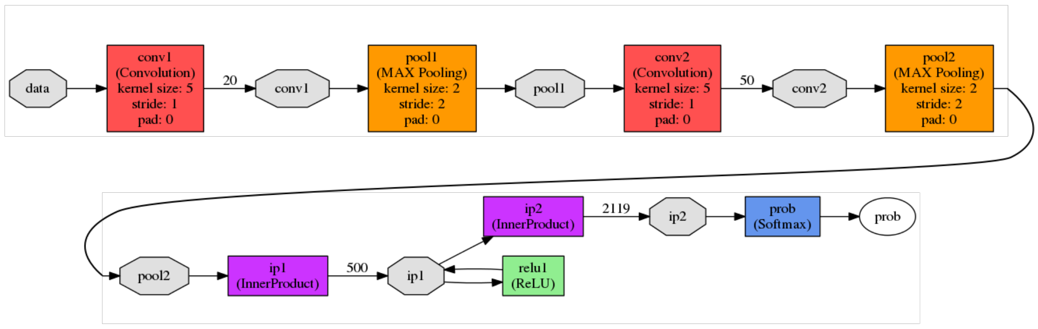


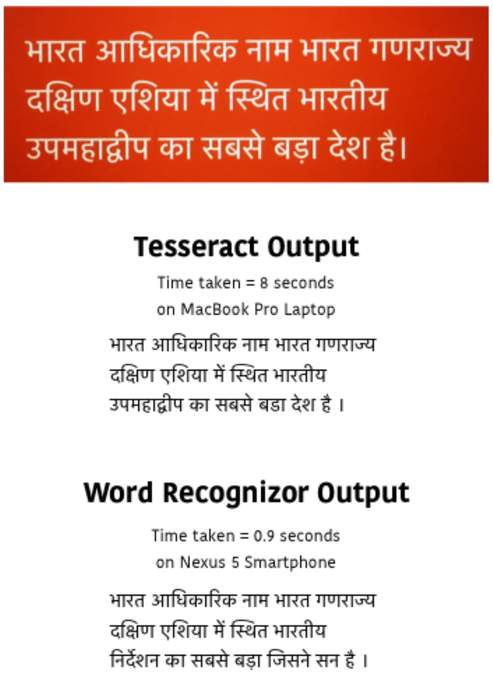
Figure 4.4: Network architecture

Training of this architecture was performed with 2,54,280 training images, with 2119 classes. The resulting model was 26.4 MB in size.

# CHAPTER 5

## Performance evaluation

The end to end application was developed encapsulating all the modules discussed so far. For benchmarking performance, comparison was done with Tesseract OCR. The Tesseract engine also claims to be an end to end solution, but the engine has not been successfully ported to Android. Demo programs work for English, but fail to execute for Hindi version of the engine. Therefore performance of the tesseract engine was only measured on a desktop machine.


भारत आधिकारिक नाम भारत गणराज्य
दक्षिण एशिया में स्थित भारतीय
उपमहाद्वीप का सबसे बड़ा देश है।

**Tesseract Output**

Time taken = 8 seconds
on MacBook Pro Laptop

भारत आधिकारिक नाम भारत गणराज्य
दक्षिण एशिया में स्थित भारतीय
उपमहाद्वीप का सबसे बडा देश है ।

**Word Recognizor Output**

Time taken = 0.9 seconds
on Nexus 5 Smartphone

भारत आधिकारिक नाम भारत गणराज्य
दक्षिण एशिया में स्थित भारतीय
निर्देशन का सबसे बड़ा जिसने सन है ।

Figure 5.1: Comparison of Tesseract OCR engine with our Word Recognizer engine

# CHAPTER 6

# Conclusion

The aim of this project was to develop an OCR system with two requirements. Firstly, it should work for Indian Languages. In this regard, the word recognition approach tackles a major challenge that arises due to complexities of these scripts. With this framework, the system can be scaled up for any Indian Language with the mere requirement of a corpus. The framework developed requires just corpus data as input. It performs analysis of the corpus, chooses the best set of words, generates sample images for training the neural network, and then successively trains the CNN and finally results in a model file. There is no intervention required by an engineer. There is also no knowledge of the script required by the engineer. This gives a major boost to scaling the system to all Indian Languages, which is reported to represent about 66 different scripts.

Secondly, the application was to be developed for a smartphone. Although Internet penetration is increasing, most users are still on slow, intermittent or in some cases no network connection. This condition is severe especially in India. Given that the system had to be designed for India and Indian languages, an offline solution carries lot of merit. The implementation of a smaller neural network model that can run on low power devices becomes a good choice. This challenge was considered seriously even by Google for their Translate application, and has been heavily research by their team.

Although the proposed solution tackles the issues stated above, it also comes with its limitations. The biggest limitation is the engine's vocabulary. Attempts were made to optimize the system for this limitation by corpus analysis, but for successful deployment, a larger more complex system is required.

# CHAPTER 7

# Limitations and proposed pipeline

The major limitation of this entire word recognition engine is the limit of its vocabulary. But, given its tremendous performance, it can be a powerful component of a larger pipeline.

To tackle this problem a larger pipeline is proposed which uses the Word Recognition engine primarily and forges ahead to complex systems in case of failure. This proposed pipeline has 4 stages as described below:

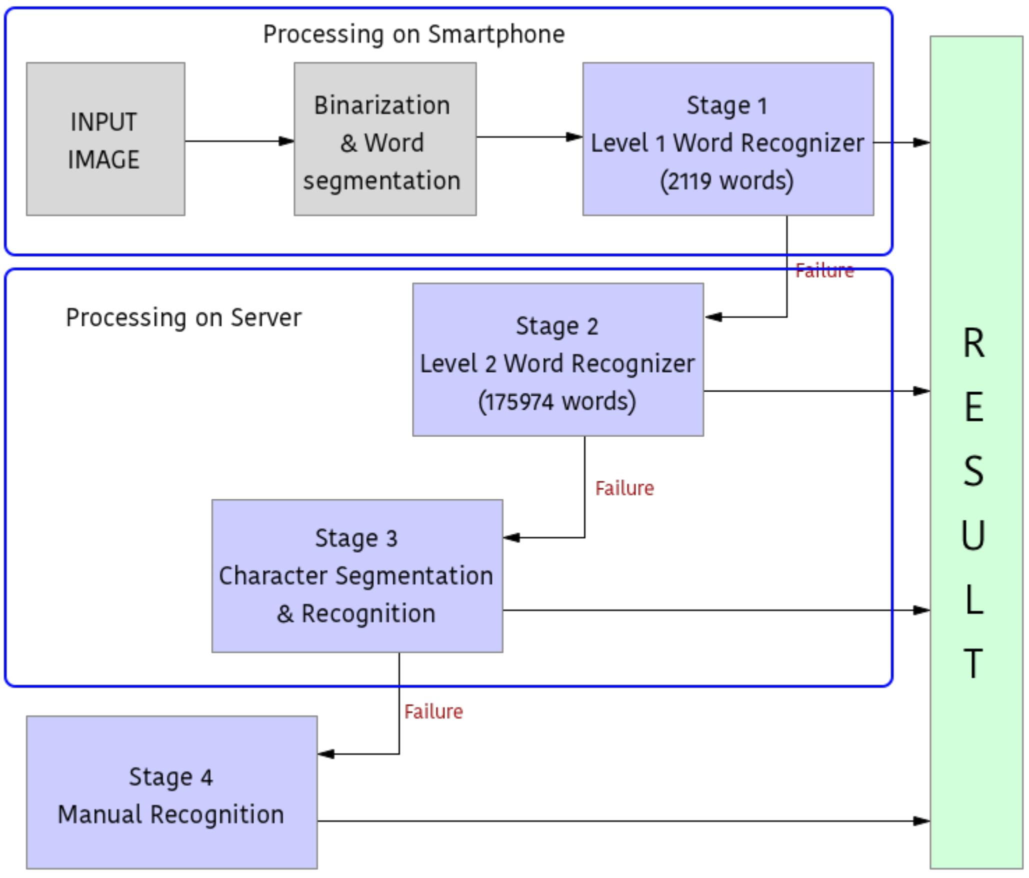

Processing on Smartphone
INPUT IMAGE
Binarization & Word segmentation
Stage 1 Level 1 Word Recognizer (2119 words)
Failure
Processing on Server
Stage 2 Level 2 Word Recognizer (175974 words)
Failure
Stage 3 Character Segmentation & Recognition
Failure
Stage 4 Manual Recognition
RESULT

Figure 7.1: Illustration of the proposed pipeline

1. Stage 1: Level 1 Word Recognizer (2119 words)
   This Recognizer uses the same architecture described so far. Given its small size of 26.4MB, it is ideal for execution on a mobile device. This Level 1 Recognizer has a coverage of 75%, which means for majority of the cases, the pipeline succeeds at this stage and can return the results.

2. Stage 2: Level 2 Word Recognizer (175974 words)
   This Recognizer again is of the same architecture. It's size will roughly be around 375MB. Together with the Level 1 Recognizer, the pipeline achieves a coverage of 98% on the corpus at this stage. This recognizer is meant to be executed on a remote server. When the pipeline doesn't succeed in Stage 1, the image is uploaded to the server, which executes the recognition engine on a more powerful machine and the API returns the recognition results appropriately. If the recognition succeeds, results are returned and the execution terminates. Two things must be noted here. Firstly, training of such a recognizer has not been attempted and its convergence is still in question. Secondly, this is the final Word recognizer in the proposed pipeline, but a cascade of multiple Word recognizers may be implemented in practice.

3. Stage 3: Character recognition engine
   This stage is based on the traditional OCR engines, it requires some more preprocessing of the input image in terms of character segmentation. After the character recognition engine executes, a dictionary can be used to overcome character recognition errors, to improve word level accuracy. If for some reason, the Character recognizer fails at one of its steps of execution, then the image is sent forward to Stage 4

4. Stage 4: Manual recognition
   This stage is reached when the image couldn't be recognized by the pipeline. In this case, all such failure cases are dumped for manual examination, and can be studied to make enhancements to previous stages of the pipeline.

# REFERENCES

1. **Bradley, D.** and **G. Roth** (2007). Adaptive thresholding using the integral image. *Journal of graphics, gpu, and game tools*, **12**(2), 13–21.

2. **Govindaraju, V.** and **S. Setlur**, *Guide to OCR for Indic Scripts*. Springer, 2009.

3. **Kittler, J.** and **J. Illingworth** (1985). On threshold selection using clustering criteria. *Systems, Man and Cybernetics, IEEE Transactions on*, (5), 652–655.

4. **LeCun, Y.** *et al.* (2015). Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*.

5. **Otsu, N.** (1975). A threshold selection method from gray-level histograms. *Automatica*, **11**(285-296), 23–27.

6. **Samet, H.** and **M. Tamminen** (1988). Efficient component labeling of images of arbitrary dimension represented by linear bintrees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **10**(4), 579–586.

7. **Sezgin, M.** *et al.* (2004). Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic imaging*, **13**(1), 146–168.

8. **Smith, R.**, An overview of the tesseract ocr engine. *In icdar*. IEEE, 2007.

9. **Wikipedia** (2016). Optical character recognition — wikipedia, the free encyclopedia. URL `https://en.wikipedia.org/w/index.php?title=Optical_character_recognition`.