Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Ethiopic and Latin Multilingual Text Detection and Script Identification from Videos and Images

By: Atirsaw Awoke Gebrie

April 10, 2018
Addis Ababa, Ethiopia

Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Ethiopic and Latin Multilingual Text Detection and Script Identification from Videos and Images

By: Atirsaw Awoke Gebrie
Advisor: Menore Tekeba

A thesis submitted to the School of Electrical and Computer Engineering in partial fulfilment of the requirements for the Degree of Master of Science in Computer Engineering

April 10, 2018
Addis Ababa, Ethiopia

Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Ethiopic and Latin Multilingual Text Detection and Script Identification from Videos and Images

By: Atirsaw Awoke Gebrie

Approval by Board of Examiners

Dr. Yalemzewed Negash Dean, School of Electrical and Computer Engineering	Signature
Menore Tekeba Advisor	Signature
Internal Examiner	Signature
External Examiner	Signature

April 10, 2018
Addis Ababa, Ethiopia

Abstract

Both caption and scene texts which are found in images and video frames contain valuable information. These texts can be used for many applications to answer questions like what, when, where, and by who to give context to the images and video frames. So, automatic text detection enhances the user's understanding of the media content. In Ethiopia, most street posts and promotional boards are written in multi-lingual characters such as Latin (English, Afaan Oromo etc.) and Ethiopic (Amharic, Tigrigna etc.). In this work, we have studied Ethiopic and Latin multilingual text detection and script identification from videos and images for both caption and scene texts.

After the images and video frames are pre-processed, maximally stable extremal region (MSER) algorithm, aspect ratio and stroke width transform (SWT) algorithm are used to extract text regions and discriminate non-text patterns from texts, respectively. Then texture features are computed using local binary pattern (LBP) from the extracted regions. Finally, support vector machine (SVM) is used to classify text region vs non-text using the computed LBP features. In the next phase of our work, which is script identification, the detected text regions are binarized using Niblack's algorithm. Radon transform was applied on the binarized text regions to detect and correct skew. Segmentation of lines using horizontal projection profile followed by word segmentation using vertical projection profile is done when the text region contains more than one line of text. From the resulting text words, texture features are computed again using LBP and the text words are categorized to their respective script classes using SVM.

We used the International Conference on Document Analysis and Recognition(ICDAR) 2003 data set as well as prepared a new multilingual Ethiopic and Latin script image dataset to evaluate our method. Our text detection method performs better compared

with the state of the art method with precision 5%, recall of 10% and 8% f-measure on ICDAR 2003 dataset. The text detection was also evaluated on our dataset, where 81% precision,74% recall with a f- measure of 77% was obtained. The overall system gives 79.9% accuracy of script identification.

Keywords: Multilingual Text Detection, Maximally Stable Extremal Region, Stroke Width Transform, Support Vector Machine, Optical Character Recognition.

Declaration

I, the undersigned, certify that research work titled Ethiopic and Latin Multilingual Text Detection and Script Identification from Videos and Images is my own work. The work has not been presented elsewhere for assessment. Where material has been used from other sources, it has been properly acknowledged.

Atirsaw Awoke Gebrie: Signature:————

Date of submission: March, 2018

Place: Addis Ababa

This thesis has been submitted for examination with my approval as a university advisor.

Advisor: Menore Tekeba: Signature:————

Acknowledgement

First and foremost, praises and thanks to the GOD, the Almighty, for His showers of blessings throughout my research work to complete successfully.

Special mention goes to my enthusiastic advisor, Menore Tekeba. The door to his office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work but steered me in the right direction whenever he thought I needed it.

In second place, many thank goes to all my dear friends, who has helped me in every aspect of my life during this thesis work. without them, it would have been very difficult to my thesis work to get completed.

Last but not the least, I would like to thank my family: my parents and to my brothers and sisters for supporting me spiritually throughout writing this thesis and my life in general.

Table of Contents

List of Acronyms

CAMSHIFT: Continuously Adaptive Mean Shift

CC: Connected Components

CCA: Connected Component Analysis

ER: Extremal Region

FP: False Positives

FN: False Negatives

HOG: Histogram of Gradient

ICDAR: International Conference on Document Analysis and Recognition

LBP: Local Binary Pattern

MGD: Maximum Gradient Difference

MSER: Maximally Stable Extremal Region

OCR: Optical Character Recognition

RQ: Research Question

SVM: Support Vector Machine

SWT: Stroke Width Transform

TV: Television

TP: True Positives

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Background

The rapid advances in electronic multimedia technologies changes the way in which human's store and process information. Recently with the increasing availability of low cost portable cameras and mobiles, the number of images and videos being captured are growing at an exponential rate. The amount of information that these images and videos harbor is astonishing, the main problem is how to extract the information you are looking for. One of the main attributes of information for images and videos is the text that is located in it. For example, the name of the person being interviewed, the score of a football match, name of the player, locations, brands of products, date and time.

Given the rapid growth of camera-based applications readily available on mobile phones, understanding scene text is more important than ever, and application that are able to answer questions such as,"what does this sign say?" are becoming increasingly popular. This work shows how to detect regions containing text in an image and identifies the script of the text. Text in images and videos can be classified into two categories: caption text and scene text [1, 2]. Caption text is artificially overlaid on the images or video frames at the time of editing [1, 2]. Caption text can have arbitrary font, size,

color, orientation, and location. For video document, a caption text event may remain stationary and rigid over time, or it may translate, grow or shrink, and change color. A good text extraction system must be able to handle as wide a variety of these types of text as possible.

Text naturally occurring within the scene, or scene text, also occurs frequently in image and video [1, 2]. Examples include text appearing on vehicles (license plate, model names, company names on commercial vehicles, bumper stickers etc.), text on signs and billboards, and text on T-shirts and other clothing. Technically, the extraction of scene text is a much tougher task than caption text due to varying size, position, orientation, lighting, and deformation[3].

A variety of approaches to text information detection and extraction from images and videos have been proposed for specific applications. Such as, systems developed to detect text in an image and translate the text to another language in [3, 4], license plate location in [5, 6], page segmentation in [7], Text based image/video indexing in [5, 8], for assisting the visually impaired to walk on roads in [9, 10]and to help tourists to visit sightseeing venues in [4].

A number of approaches, such as connected component based, texture based, edge based, and other methods [11, 12, 13, 14], have been proposed for text detection from images. Most of the methods can be adopted for video scenes. But detecting and extracting text from videos presents a unique challenge due to many undesirable properties,such as variations like alignment, color, motion, edge and compression. These properties make difficult the task of developing efficient text detection and extraction from images [15].

Fortunately, text in videos usually persists for at least several seconds, to give human viewers sufficient time to read it [16]. This temporal nature of videos is very valuable and can be well utilized for text detection in videos.

The text detection and extraction problem can be divided into the following sub problems: (1) detection, (2) localization, (3) tracking, (4) extraction and enhancement, and (5) OCR [15, 17, 18]. Text detection, localization, and extraction are often used interchangeably in the literature. Text detection refers to the determination of the presence of text in a given frame (normally text detection is used for a sequence of images) [15, 18]. Text localization is the process of determining the location of text in the image and generating bounding boxes around the text [15, 18]. Text tracking is performed to reduce the processing time for text localization and to maintain the integrity of position across adjacent frames [15]. Text extraction is the stage where the text components are segmented from the background [16]. However, in our work all these terms can be represented with a single general term which is text detection.

Video signal is basically any sequence of time varying images called frames. To detect and extract texts in videos first, a digital video is decomposed into video frames where a frame is flashed on screen for a short period of time and then immediately replaced by the next one [2, 19]. From these video frames, redundant frames are discarded by performing frame similarity check which results in selection of key frames [2, 17, 19]. This removal of redundant frames can be useful to reduce the amount of memory and time needed for video data processing and complexity greatly [9, 19]. Histogram difference comparison which is used in [9, 10, 20] is employed for this purpose.

Texts taken from the video frames will suffer from low resolution, blur, complex backgrounds, to mention a few issues [12, 18, 21]. In order to minimize the effect of the inherent problems with video frames, pre-processing techniques is essential for text detection from images [18, 22, 23]. In order to obtain an image with more stable text regions, two different filters are used for sharpening and noise removal.

Then we can get into the main task which is text detection and script identification through the following processes. Since text characters usually have consistent color, we begin by finding regions of similar intensities in the image using the MSER region detector which is applied for character candidate extraction in [23, 24]. Although the MSER algorithm picks out most of the text regions, it also detects many other stable regions in the image that do not contain text [23]. For text detection either rule based or machine learning approach can be used.The combination of the rule-based methods and a machine learning approach are is better to produce good result [23, 25]. False regions, which are detected as text region by MSER is minimized using the geometric property of texts which is the aspect ratio. Then SWT Algorithm is used to refine the false positives. Even after the SWT filtering is performed regions which are not a text are also detected as text region candidates. To eliminate those candidate regions which are not text, each candidate text regions is verified using a machine learning technique. SVM classifier is implemented as a classifier in many works like [14, 26] for text detection from images. So, the extraction of features from an image is needed to obtain only the relevant information that is helpful for characterization [18, 23, 26]. Most of features used for text detection come from texture extraction [14]. From this fact a LBP in [14]is implemented as feature extractor for text detection from images. In this work LBP is selected to compute feature vector of each candidate text regions. The feature vector candidate text regions are computed using LBP. SVM classifier finally determines if the

text candidates actually contain a text string using the feature vector as input.

If an image has multilingual segments, the recognition problem become more challenging, as it requires the identification of the languages before the recognition of the content. The text line is segmented in [12] by finding the valleys of the horizontal projection profile followed by word segmentation using vertical projection profile. We used horizontal and vertical projection profile to segment line and word respectively. Here again feature vector of the segmented text block is computed using LBP. These features are used as input for SVM which is used for classifying the texts to their respective scripts.

1.2 Statement of the Problem

The text in images and videos is small in quantity. But often carries useful information of the media contents; they usually present important names, locations, brands of the product, and score of the match, date and time [19]. However, the text in videos can be superimposed on the arbitrary backgrounds or embedded on the surfaces of the objects in the scene with varying font, size, color, alignment, movement and lighting conditions [12, 18, 21]. This makes the text detection task extremely difficult. The aim of this research is to find a proper way to detect texts in both Ethiopic and Latin from videos and images and identify the texts to their respective script.

The process of text detection and script identification from video and images, still after many decades of research works, has not yet reached a high level of accuracy. In addition to this, nothing has been done for Ethiopic character text detection and script identification. To the best of our knowledge, this will be the first attempt to handle the challenges towards the domain of recognition of Ethiopic script from videos and images. It is still hot research issue globally even for those languages with rich computational

linguistic resources like English and French. Since no research has been done for Ethiopic characters and much of the street posts, guides and promotional boards are written in multi-lingual characters both Latin (English, Afaan Oromo etc.) and Ethiopic (Amharic, Tigrigna etc.), text detection and script identification for multi-lingual video and image texts will be a pioneering research work for Ethiopia. This motivates us to research and show a new direction in handling text detection and script identification of Ethiopic and Latin multi-lingual characters.

1.3 Research Questions

The scope of this work is multilingual text detection and script identification from images and videos. The scientific research questions (RQs) are:

- Can preprocessing the target image or video frames with image enhancement techniques improve performance of text detection?
- Can the combination of rule based and machine learning approaches achieve competitive performance for text detection from images and video frames?
- Can the combination of texture features with machine learning algorithm can be used for Ethiopic, Latin script identification of words with blurring, noise, less contrast and complex background?

1.4 Objectives

1.4.1 General Objectives

The goal of this thesis work is to develop a system which will detect texts from images and videos and identify if they are Latin or Ethiopic scripts.

1.4.2 Specific Objectives

- To assess different techniques of text detection and scripts identification.
- To apply the text detection methods on images and videos
- To design script identifier that uses the text detection from images and videos
- Evaluate the performance of both text detection and script identifier implemented using test datasets.

1.5 Significance of the Study

- It provides a research output for researchers in the development of multilingual Ethiopic/Latin OCR from images and videos.
- The research plays a great role in understanding the challenges and steps of text detection and script identification from multilingual Ethiopic/Latin text images and videos.
- It opened a new OCR research direction towards the development of language translation systems which assist tourists and blind peoples.

1.6 Contribution of the Thesis

This paper extends the field of text detection and script identification from videos and images with the following contributions: (1) it handles the case of detecting texts of Ethiopic script in images and videos which is the first work to address the problem;(2) We also introduce datasets for problems of Ethiopic script text detection from images;(3) an end-to end script identification, integrating the proposed text detection algorithm is

performed;(4) the proposed subjective and objective evaluation protocol is detailed and result comparisons is performed with other works using international dataset.

1.7 Methodology

1.7.1 Data Collection

Two types of dataset are prepared used for respective objective and subjective evaluations. For subjective evaluation images which contain text from videos and captured using cameras are prepared. For objective evaluation for each image word level ground truth is prepared using XML format used by DetEval software. In objective evaluation two datasets are used, the one which is prepared by ourselves which contain multilingual Ethiopic and Latin scripts and the is other international benchmark ICDAR dataset.

1.7.2 Designing and Implementation Tools

- Literature Review: reading books, articles, research papers, and materials related to the subject matter which helps for selecting efficient algorithms.
- Preprocessing Data: making appropriate input data for the detection phase like filtering the Key frame selection, Sharpening, Noise removal etc.
- System modeling and design: selecting rule-based refining like aspect ratio and algorithms such as SWT and SVM.
- Implementation: MATLAB R2016-a software has been used for implementing the proposed system.
- Experimentation and discussion: discussion on the results, conclusions and future directions.

1.8 Thesis Outline

This thesis is organized as follows: chapter two is about literature review on text detection and script identification from images and videos which has been done both in local and abroad researchers. Chapter three provides a detailed understanding of script identification from images and videos. Chapter four contains the design and development of this research. Chapter five is about dataset preparation, evaluation protocols and experimentation. The last chapter provides the conclusion of the work and future directions.

Chapter 2

Literature Review

Review of previous works both local and abroad on text detection and multilingual script identification is presented here under this chapter. Current text detection approaches can be broadly categorized into four groups: texture-based approach, region-based approach and hybrid approach and Morphological based text detection.

2.1 Texture Based Approach

These methods are based on the fact that, texts in images have distinct textural properties which distinguish them from the background [17]. In [27] wavelet transform of an image is done to characterize the local energy variations (LEV) of pixels in the successive scale levels. In each scale level, the corresponding local energy variations are computed. The resulting binary map image in each scale level is subsequently analyzed by connected component analysis (CCA) technique to label different objects and background. Finally, all text regions in the consecutive scale levels are fused into the original image and text regions are detected.

The work in [26] proposed a technique that uses a combination of (Continuously Adaptive Mean Shift)CAMSHIFT and SVM for detection and extraction of text. They use a small window to scan the input image, classifies the pixel located at the center of the

window into text or non-text by analyzing its textural properties using a SVM. Then CAMSHIFT algorithm is used to verify text regions which are the result of the texture analysis.

The writers in [11] applied coarse-to-fine detection framework by using different text properties in different detection stages. In the coarse detection, candidate text regions are firstly obtained using property of dense intensity variety and contrast between text and its background. In the fine detection step, Texture property is used to discriminate text with other non-text patterns. Texture features such as wavelet moment features, wavelet histogram features, wavelet co-occurrence features and crossing count histogram features are used to identify text lines from the candidate ones. A forward search feature selection algorithm is used to find effective features and an SVM classifier is used to perform text/non-text classification task.

One system developed by scholars in [12] for detection of text makes use of Laplacian method based on wavelet and color features. The Maximum Gradient Difference (MGD) map is obtained by moving the window over the image. Text regions typically have larger MGD values than non-text regions because they have many positive and negative peaks. Therefore, they convert the input frame into binary frame and then Fuzzy C-means is applied to classify the feature into two clusters: background and text candidates.

2.2 Region Based Approach

Region-based approaches [17] attempt to use similarity standard of text, such as color, size, stroke width, edge and gradient information. These approaches usually have lower computation cost and the outputs can closely cover text regions. In paper [13] the

authors proposed a method which has three stages: Candidate text region detection, text region localization and character extraction. In its first stage, they used the magnitude of the second derivative of intensity as a measurement of edge strength, and this allowed a better detection of intensity peaks that normally characterizes text in the images. Edge detector is carried out by using a multiscale strategy in which the multiscale images are mainly produced by Gaussian pyramids after successively applying low pass filter and down sample the original image reducing the image in both vertical and horizontal directions. In the second stage, they assume texts are found in clusters arranged compactly. So, this characteristic of clustering is used to localize text regions. In the third stage, the existing OCR engines were used.

For text detection mainly from video images, the authors of paper [14] applied a two-stage methodology. In its first stage, the text lines are detected based on the canny edge map of the image. In the next stage, the result is refined using the sliding window and a SVM classifier. Feature vectors of the candidate regions are obtained using LBP which describes the local edge distribution.

In research work, [28] they propose a (Connected Component) CC-based text detection algorithm, which employs MSER as basic letter candidates. To improve the weakness of MSER they deploy the complimentary properties of canny edges and MSER is combined in edge-enhanced MSER. Further they propose to generate the SWT image of these regions using the distance transform to efficiently obtain more reliable results. Then they apply the geometric as well as stroke width information to perform filtering and pairing of CCs. Finally, letters are clustered into lines.

In other work which uses a connected component-based approach for text extraction [24] is based on color reduction technique. The Color images are converted to grayscale images and edge image is generated using a contrast segmentation algorithm. Luminance value thresholding is applied to increase the contrast between the possibly interesting regions and the rest of the image. Then they applied the horizontal projection of the edge image in order to localize the possible text areas.

In research work [21] The Color-Edge combined algorithm consists of two stages is proposed for text detection and extraction. In the first stage, they detect the text area by applying the Color-Edge combined algorithm. In the second stage, the text background is removed and the left part is binarized for text location. Transition Map method is used to filter the text from the background. The model makes use of the exponential changes of color between the edge of text and background to detect the text area, and then the background is removed. To improve the efficiency of the method, canny edge detection and some morphology operation is performed.

2.3 Hybrid Approach of Texture and Region Based Methods

Hybrid approaches [17] seek to introduce textural property of text regions into region-based approach. In a research work done in [29] they try to combine texture and CC-based information to detect text. They propose first and second order statistical texture features to detect and localize the text in the image. CC extraction is used to segment candidate text components from the localized text region. Finally, morphological operations and heuristic filters are used to filter out non-text components.

The other literature in this regard is the one mentioned in [30]. The authors created text confidence map for a series of different scaling of gray scale images to represent the possibility of text on an area using the waldboost classifier trained by histogram of gradient(HOG) features. They adopted Niblack's algorithm to convert gray scale image in to binary image, and uses the conditional random field to determine whether the candidate area contains text or not. Finally, they applied minimum spanning tree to connect the same line of text.

2.4 Morphological Based Approach

In work [31] the researchers go through the following steps to detect texts. They split the video into frames. Key frame is selected from the frames by edge comparison with the help of the Sobel edge detector. Text candidate regions are generated again using Sobel edge detector. They assume texts in video frames found closer to each other. Thus, morphological dilation operation is performed to remove pixels that are far away from the candidate region.

2.5 Script Identification

In work [22] the authors used traditional document analysis techniques for identifying video text at word level. They use skeleton image and super resolution techniques as pre-processing to overcome the challenges of video text. Gabor filters, Zernike moments and gradient directional features are used for feature computation and SVM as classifier earlier applied in tasks of handwritten character recognition.

The work in [32] investigates the use of text features, namely smoothness and cursiveness, without classifier for video script identification. The authors tested the method with English, Chinese and Tamil scripts. But experimental results show that the features are not good enough to distinguish the scripts when the scripts are similar in structure.

2.6 Amharic-English Script Identification

Even though nothing has been done in Ethiopic script identification from natural images and videos two papers here in our university tries to address the issue from the document images.The scholars in work [33] used observable feature values of a word image after a series of preprocessing and segmentation activities. Preprocessing such as, noise removal, binarization, line segmentation, word segmentation, size normalization and thinning are performed. CCA is performed in different region of the preprocessed word images, used as a feature to identify the language scripts. SVM is applied as script identifier in word level based on the features which is extracted earlier.

The study in paper [34] mainly aimed in investigating techniques that enhances the performance of script identification system proposed in earlier work by examining noise removal techniques and better feature extraction approaches. First, he performs noise removal and binarization using noise filtering technique and Otsu binarization algorithm respectively. Then he implements stage by stage segmentation. After these steps he performs feature extraction followed by classification using SVM.

Chapter 3

Text Detection and Script Identification

Writing is one of the most significant accomplishments of human beings.Writing become a necessity from the needs of storing information, such as commercial transactions, contracts and recording history in general.Texts which are found in images and videos contain valuable information which can be used for many applications.For this thesis work, we are interested in retrieving the textual information stored in videos and images.

3.1 Types of Texts

Texts in images and videos are commonly used for indicating names, locations, and products brands, scores of a match, date, and time, which plays an important role to understand the video content [18]. So, harvesting these texts will help for application like in [26] for Automatic annotation, indexing and structuring of images and in [27] for Video database indexing and retrieval. Text which is found in the images and videos can be overlaid on arbitrary backgrounds and their appearances will be complex with varying font, size, color, alignment, movement, and lighting condition [12, 13, 18, 26, 27]. Therefore, text detection from images is a challenging task [11, 21, 27, 29]. The goal of conducting researches on text detection from images and videos should emphases, on finding a proper way to extract different types of text with arbitrary orientations and complex backgrounds [18]. In general text which is found in images and videos can be

categorized into two as caption and scene text [11, 12, 18, 31].

3.1.1 Caption Text

Caption texts are edited texts or graphic texts artificially superimposed into the video and are relevant to the content of the video [11, 18, 31]. Since caption text is edited by the person who knows content of video, it is useful in several applications by answering when, where and by who of the events [11, 13]. So, the answer can be used for indexing and retrieval purposes [35]. Good examples of caption text are anchor's name, locations, and event in television (TV) news [18, 35]. Artist names, talk show hosts, guests, sport video scores and athlete's highlights are also the main examples of caption texts. So, by detecting text box and recognizing the text, video can be indexed based on TV personality or a location [11, 18]. Texts which is found in the video can be also used for differentiating classes of video into video topics, person appearances, and sports scores [36, 37]. In addition, text detection from images can be also used in content-oriented multimedia coding such as compression and in digital libraries for conversion of image texts to electronic versions [38].

Figure 3.1: A caption text example

The field of video text detection research actually started with the detection of caption text in the early development of the field [36, 39]. The reason for this is, caption text is generally easier to handle since most of the time it has high contrast with its background. As caption texts are manually added into the video by an operator, colors are often chosen to have high contrast with the background to improve visibility and readability [36, 39]. So, to detect caption text most approaches take advantage of the following characteristics [18, 38].

- Text is permanently in the foreground.
- Generally, it is with appropriate lighting condition, which is scene independent.
- Size, font, spacing, and orientation are constant within the text region.
- Text is usually aligned horizontally.
- All text pixels in the same text line have the same or almost the same color.
- Text pixels have a higher contrast compared to its background.
- Most of the time, text found at the bottom of the frame.
- Mostly Text is found in group as collections of words rather than isolated characters.
- The background is generally identical for moving text.
- The same text appears in several consecutive frames.

3.1.2 Scene Text

Scene texts are naturally existing text which are part of the images and videos, usually embedded in objects in the videos [11, 18, 31]. Scene text is textual content caught by a camera as an essential portion of a video scene and images, which shows the

existence of the name of a restaurant, name on T-shirts, commercials in stadiums, name of supermarkets, food containers, nameplates, traffic warnings, and road signs [11, 13, 18, 38]. Scene text has its own importance in developing valuable video applications such as video information retrieval, mobile based scene classifications, Mobile robot navigation to detect text-based landmarks, vehicle license recognition and video object recognition [13, 18]. In computer vision community having the fast development of camera-based applications readily available on mobile phones, understanding scene text is more important than ever, and applications that are available to answer questions such as, "what is the meaning of this sign?" are becoming increasingly popular [18, 40]. But scene text detection is not easy as caption text and it is difficult to develop a general approach for text detection of scene texts from videos and images [18, 29, 37, 40]. OCR is very successful in handling text from scanned documents, however when

Figure 3.2: A scene text example

natural scenes containing both text and various scene objects are given the performance of OCR extremely falls [18, 37, 40],because inputs of most of the OCR engines are well-segmented binary texts which have been distinguished from background pixels. Unluckily, the segmentation of scene text is far harder for natural scene images leading

to poor recognition results [18, 41]. Detecting and recognizing scene texts from natural images and video frames has many additional challenges due to scene text exhibits the following characteristics [18, 38, 40].

- In natural scenes, numerous objects, such as buildings, bikes, leaves of trees, cars or parts of them have similar shape and appearances to text. Text may be found within objects with untidy background. For example, corner of a building can easily be detected as I or wheels of a vehicle as O. This makes difficult to discriminate text from non-text in text detection.

- Often scene texts have decorative fonts usually to attract the attention of viewers. Identifying such unseen characters become challenging for a classifier, which is trained on normal fonts. Scene text can have any orientations and comes with challenges like low contrasts, low resolutions and blur.

- Scene texts have no clean layout. Text regions and amount of text in scene images vary from image to image. It makes text region localization and word and line segmentation harder than usual OCR document images.

- There is variability of size, font, color, orientation, style, and alignment, even within the same word, part of the text may be occluded due to object and camera movements.

3.2 Multilingual Text Characteristics

The characteristics of text that are often considered and employed in video and image text detection are contrast, color, orientation, stationary location, stroke density, font size, aspect ratio, and stroke statistics [26, 42]. These characteristics can be categorized into language dependent and language independent characteristics [42].

3.2.1 Language Independent Characteristics

- Contrast: When a background is complex, only the texts which have high contrast can be detected [28, 42]. Caption Text pixels have a higher contrast compared to scene text.

- Color: Mostly both caption and scene text string are decomposed of a group of character members in uniform color while most background outlines do not have this property [27, 42]. The color-based detection methods make use of the consistent colors of a text which distinguish text from the background color [21, 27, 42]. However, videos and images can contain text strings with more than two colors (polychrome) and it became too difficult to segment the text from the background [18, 36].

- Orientation: In contrast to caption text, scene text can have any orientation [38, 40, 43]. Moreover, it is usually affected by variations in scene and camera parameters such as illumination, focus and motion[16, 29].

- Stationary location: Scene texts of interest are stationary over time. Caption texts can be static or moving like scrolling text. Most methods [7, 19, 39, 44] have exploited the temporal redundancy of video frames to detect text.

3.2.2 Language Dependent Characteristics

According to linguistic classification Both Ethiopic and Latin scripts belongs to alphabetic literal. Their differences in the next four features have their own impacts on the video frames and image text processing [42].

- Stroke density: The stroke density is defined as the number of pixels in a stroke as a measure of its length, for strokes in vertical, horizontal, right and left diagonal

directions of the image. The stroke density of an English text string is roughly uniform [42].

- Font Size: The font size of text to be detected may vary in a wide range of values which also causes to vary the pattern of the text [11, 12, 13]. In both type of the text and both Ethiopic and Latin script texts there may be font size variation.

- Aspect ratio: In English and Ethiopic scripts, characters must be combined to form meaningful words. Except some characters in Ethiopic script like ”nu” and ”na” are single characters which have a meaning. Except the type of text there is no difference in aspect ratio due to the difference of the scripts.

- Stroke statistics: Since both Latin and Ethiopic characters do not contain many intersections except i and j for English and all the family of ”ve” for Ethiopic. There is not much difference in stroke statics of both scripts.

3.3 Key Frame Selection

Video signal can be seen as a series of images called frames. A frame is flashed on screen for a short period of time (usually 1/24th, 1/25th or 1/30th of a second known as a frame rate) and then immediately replaced by the next one [31]. Redundant frames are discarded by performing frame similarity which results in selection of key frames. It is used to reduce the amount of memory and time needed for video data processing and the degree of complexity greatly [45].

To perform key frame selection the frame difference is computed between the current frame and its predecessor frame. The frame difference is compared with certain threshold; if the difference satisfies with the threshold condition then the current frame is selected as a key frame [45]. By continuously repeating the procedure for all frames we can

extract the key frames [46]. The histogram difference of a single frame is mapped with that of its neighbor ones to check for the frame similarity in [46, 47]. Thus, by using this comparison we would be able to eliminate the redundant frames.

3.4 Preprocessing

Noise can be found in any image and it is perceived as degradation of quality. Especially scene text can be exposed to a lot of potential degradations such as blur, out of focus, uneven lighting, low resolution, which makes quality of text also decreases and become difficult to detect [11, 29]. The role of preprocessing stage is for video and image text detection is an improvement of the image data by suppressing undesired degradation of quality.

3.4.1 Sharpening using High Pass Filter

To improve the details that has been blurred, either in error or due to natural effect, sharpening can be applied [23, 48]. An image is sharpened when contrast is boosted between connecting areas with little variation in brightness or darkness. Image deblurring or sharpening is a high frequency strengthening operation or high pass filtering [48]. However high pass filtering should be exercised with caution since image noise resides in the high frequencies of the image spectrum, where the noise magnitude may be comparable to image signal. The kernel of the high pass filter is designed to increase the brightness of the center pixel relative to neighboring pixels [48]. The ideal high pass filter is given as:

$$H(u,v) = \begin{cases} 0 & D(u,v) \leq 0 \\ 1 & D(u,v) > 0 \end{cases} \tag{3.1}$$

Where D_0 is the cut off frequency and D(u,v) is the distance from point (u,v) to the center of the frequency rectangle. If the image size is MxN then its transform is the same size and the center frequency rectangle is M/2,N/2. The distance from any point (u,v) to the center of the Fourier transform is:

$$D(u,v) = \left[(u - (\frac{M}{2}))^2 + (v - (\frac{N}{2}))^2\right] \tag{3.2}$$

3.4.2 Noise Removal using Median Filter

When sharpening performed using high pass filtering in the image, the noise of the image will get higher. So, applying noise removal filter like median filter to reduce noise in an image, and retain the shape and position of the edge is very important [23].

Noise removal using median filter is more effective in terms of eliminating noise and preserving edges and fine details of digital images. As it is applied in many works such as [15, 23] to reduce noise in images for text detection in images. To destroy noise, the median filter processes element patterns of the input image. For each pattern of neighboring elements called window or support, the algorithm calculates the median value that is later used as filtering outcome for the central element of the window [49]. The median filter is well-known for its computational efficiency and its ability to preserve edges, as compared to other linear filtering techniques, in the presence of noise, especially image corruption [18, 49].

$$y(m,n) = median\,[x[i,j], (i,j) \in \omega] \tag{3.3}$$

Where ω represents a neighborhood defined by the user center around location[m,n].

3.5 Maximally Stable Extremal Region

The MSER method extracts a sum of co-variant regions, from an image called MSERs [50]. According to [8, 50], MSER works in the following steps. First it sorts all pixels by gray value and then it adds pixels incrementally to each connected component as the threshold is changed. When variation of regions with respect to the threshold is minimal it is defined as maximally stable. Each thresholded image consists several CCs which are called Extremal Regions(ERs).ERs in images of different thresholds form a parent-child relationship where child-regions are nested in parent regions. Hence, a component-tree is build. Let Q_1,... Q_2,... Q_n be a sequence of nested ERs. For each ER Q_i a stability measure Q(i) is defined as follows [50, 51]:

$$Q(i) = |Q_{i+\Delta} \backslash Q_{i-\Delta}| / Q_{\Delta} \tag{3.4}$$

Where $|\,.\,|$ denotes the area of a region, A\B denotes the set of pixels in A which are not in B, and Δ is a parameter of the method.Q(i) measures the area of change of the regions $Q_{i-\Delta}$and $Q_{i+\Delta}$ normalized by the region Q_i. Hence, ERs having local minima of Q(i) are defined as MSER. In text detection mostly the MSER is used to generate preliminary regions which are used to construct a model or Extracting character candidates. Since the consistent color and high contrast of text regions with the background leads to stable intensity profiles [8]. Although the MSER algorithm picks out most of the text, it also detects many other stable regions in the image that are not text. The MSER algorithm has been used in text detection from images by combining MSER with Canny edges indifferent literatures like [8, 28, 42]. Canny edges are used to cope with the weakness of MSER to blur. Over a large range of thresholds, the local binarization is stable in certain regions, and have the following characteristics [8, 50].

- Invariance between the image intensities.

- Stable
- Multi scale detection, different shapes, sizes of letters and large structures are detected.
- Affine invariant.
- Better repeatability.
- Small regions are detected.
- It does not work well with motion blur.
- The approach is instead sensitive to natural lighting effects as change of day light or moving shadows.
- It is good for change in orientation and scale.

3.6 Aspect Ratio

Usually text components have regular shape and reasonable size. Aspect ratio is an effective feature for distinguishing between text and non-text components. As a result, it is applied in many works for text detection [23, 28, 29, 30, 40]. If there are a reasonable number of text components with certain height and width in candidate region, most probably the region contains a text [28]. Components satisfying the empirically determined threshold on aspect ratio are kept as potential text regions while the remaining components are discarded [28, 30]. Both in Latin and Ethiopic scripts, characters must be joined to form meaningful except these Ethiopic characters ”nu” and ”na”. So, the feature is useful to filter out false character candidates [23, 28]. To handle elongated characters like ”i” or ”I” in Latin and ”ne”, ”pe”, ”pee”, ”ye”, ”gou” for Ethiopic

a threshold should be small enough. This feature is robust and can be used for many languages [52] . It is calculated using the formula 3.5 [23, 53]:

$$Aspect\ ratio = \frac{max(width, height)}{min(width, height)!} \tag{3.5}$$

3.7 The Stroke Width Transform

The SWT is a method proposed in [54] to detect text from poor images which are degraded by noise, by identifying shapes that have consistent stroke width. SWT tries to detect strokes inside the given image. These are elements of finite width, bounded by two roughly parallel edges. So, Pairs of parallel edges are then used to compute stroke width for each pixel and pixels with similar stroke width are clustered together into connected components of characters [25, 55]. The characters can be filtered from complex background, grouped into words, and superimposed into output image without any knowledge of the language, font and type of text [54]. It has been state of the art in text detection from images since it is introduced for the first time by the author of [54]. Since then so many works such as [23, 28, 55] uses the technique for text detection in complex background.

It detects stroke pixels by shooting a pixel ray from an edge pixel(px) to its opposite edge pixel(py) along the gradient direction (dx). The ray is considered as valid only if the gradient orientations of the pair of edge pixels are roughly opposite to each other. Otherwise, the ray is considered as invalid. All pixels covered by a valid ray are labeled by the same stroke width, which is the distance between the pair of edge pixels. In this way, SWT filters out the background pixels and assigns text pixels with stroke widths.

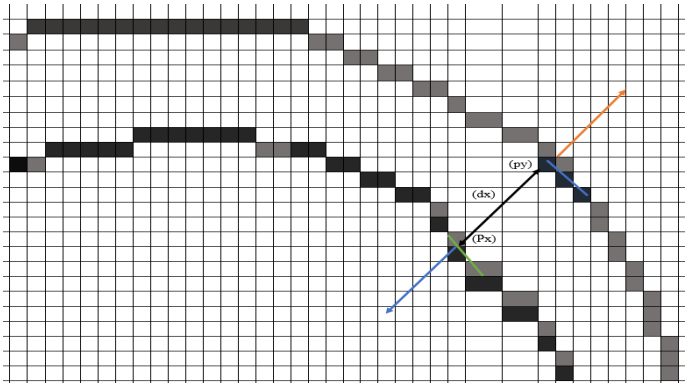


Figure 3.3: Stroke width computation by ray casting [54]

3.8 Binarization

The goal of binarization module is to isolate the pixels of detected text regions into classes of text and background. For document images, simple thresholding is typically capable of converting a gray scale image into a binary image suitable for OCR. This technique assumes that the text and the background colors are uniform which is typically black on white so the image histograms are bimodal [18]. In video, however text often appears against complex, non-uniform backgrounds. The text color may also vary due to uneven illumination of scene text, anti-aliasing or due to bleeding caused by video compression [18]. So various binarization methods [1, 30, 56] have been used to get the binary image which is ready to directly feed into OCR systems. Niblack's algorithm which is used in [30, 56] to binarize natural scene text is better due to its high efficiency and non-sensitivity to image degrading. It calculates a pixel wise threshold by sliding a rectangular window over the gray level image. The computation of threshold is based on the local mean m and the standard deviation s of the pixels in the window and given by the equation 3.6:

$$T_{\text{Niblack}} = m + k * s \tag{3.6}$$

Where m is the mean, k the weight and S the standard deviation.

3.9 Skew Detection and Correction

When video is taken and an image is captured using a camera, skew is inevitably introduced into the image and videos due to various factors [17, 56, 57]. So, it is useful to determine the skewness and orientation of text in the image for the following reasons [57].

- Improves text recognition. Performance of recognition systems degrades if the image is skewed. So, it improves text recognition.
- The text line baselines can be found more robustly if the skew angle is accurately known so it improves visual appearance.

Among several skew detection and correction algorithms Radon transform is better[57], since it is suitable in the presence of noise and reduces time complexity of an algorithm. Applying the Radon transform on an image f(x,y) for a given set of angles is the same as computing the projection of the image along the given angles. The resulting projection is the sum of the intensities of the pixels in each direction, which is a line integral. The result is a new image R(ρ,ϑ).

$$R(\rho,\theta) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\delta(x\cos\theta + ysin\theta - \gamma)\,dx\,dy \tag{3.7}$$

Where theta is the angle of the line, r is the perpendicular offset of the line,f(x,y) is 2D object in Cartesian coordinate x,y and R(ρ,ϑ)is the Radon transform of f(x,y)

3.10 Segmentation

After conducting text localization from images and video frames, to perform script identification the line of the text then the word in each line should be segmented.

Segmentation of characters from video frames and images is not easy as segmenting characters from scanned document images due to the following challenges, the difference in the skew angle between lines, characters or even along the same text line, overlapping words and adjacent text lines and touching characters, low resolution and complex background [12, 18]. To handle this problem, a number of works, try to find efficient way for segmenting characters from images and using vertical and horizontal projection which is applied in [12] are common.

3.11 Feature Extraction

The main goal of feature extraction is to obtain the most relevant information from the original data and represent that information in a lower dimensionality space [11, 22, 33]. The majority of features for text detection come from texture extraction and analysis research area. Because a text can be treated as a periodic repetition of similar patterns with specific alignment presents some of the basic properties of texture [14]. The texture features computed from images in both from non-text image and images with text are used to train a classifier which then classifies a given image as a text and non-text [12, 14].For text detection, four types of texture features such as, wavelet moment features, wavelet histogram features, wavelet co-occurrence features and crossing count histogram features are applied to classify text lines from the candidate ones [11].LBP in [14] is used to discriminate texts from non-text regions for text detection. The texture features computed from text in different scripts are used to train a classifier which then classifies a given text as being one of the script classes [22, 33, 34]. Texture based features such as Gabor filter, Zernike, moments and gradient directional features used in [22] for script identification. Different variations of texture features computed from Gray-level Co-occurrence matrix (GLCM), LBP and HOG are used in [58] for word wise script identification purpose.

3.11.1 Local Binary Pattern

LBP is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number [14, 58, 59]. The basic idea is to summarize the local structure in an image by comparing each pixel with its neighborhood. Take a pixel as center and threshold its neighbors against if the intensity of the center pixel is greater or equal its neighbor, then denote it with 1 otherwise denote 0 [58, 59].

Table 3.1: LBP computation

1	2	2		0	0	0	Binary: 00010011
9	5	6	->	1	1	1	Decimal: 19
5	3	1		1	0	0	

3.12 Classification

The preliminary aim of the classification phase of our system is twofold. First it is used for candidate text regions which come from as output from rule-based filtering to distinguish as text or non-text. Secondly when the image contains multilingual text, each line then each word of the text will be segmented and classified to their respective script class. Different machine learning algorithms are used for text detection from images with great success by eliminating false positives which are detected as text regions [14, 18]. Machine learning algorithms are also applied for Classification of scripts in multilingual environment [33].

3.12.1 Support Vector Machine-Based Methods

In research work [26] the system uses a small window to scan the input image, classifies the pixel located at the center of the window into text or non-text by analyzing its

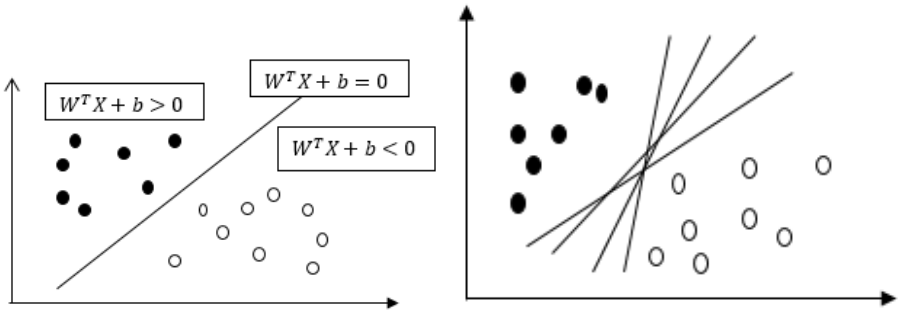
textural properties using a SVM. Four different types of texture features are used to classify text lines from the candidate ones using SVM in [11]. SVM is a linear two class classifier [60, 61]. A linear classifier is the dot product of two vectors, which is referred as an inner product, defined as:

$$w^T * x = \sum_i w_i * x_i \tag{3.8}$$

Where X is the data set, W is the weight vector, b is the bias and small x denotes components in the dataset. The line which classifies the data is called a hyper plane.

$$f(x) = w^T * x + b \tag{3.9}$$

As it is seen in Figure 3.4 (b), there are many hyper planes that classifies the data.



(a) with one optimal hyperplane (b) With four hyperplane

Figure 3.4: A linear classifier[61].

From the possible hyperplanes only one of those achieves maximum classification. If we choose one of the hyperplanes randomly to classify, it might end up closer to one set of datasets compared to others and the result will be biased [63, 64]. So, to avoid this the concept of maximum margin classifier or hyper plane is introduced. The distance between the support vectors and the hyper plane is called a margin. Maximum margin is gives better empirical performance, avoids local minima and better classification [60].For

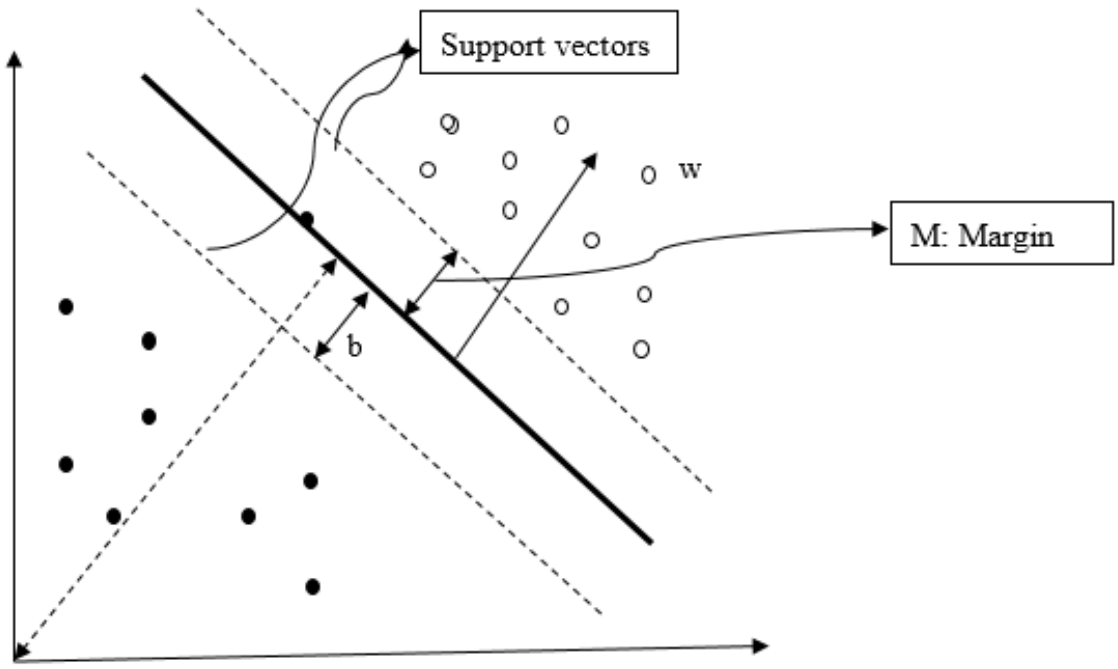


Figure 3.5: Linear SVM classifier with the maximum margin[61]

a given hyper plane[60, 61] among positive and negative samples the margin of the hyper plane defined by the weight vector w with respect to dataset is given by.

$$M = 1/||w|| \tag{3.10}$$

The maximum-margin classifier is the discriminant function that maximizes the geometric margin $1/||w||$ which is equivalent to minimizing $||w||^2$ [60, 61]. This results to the following constrained optimization problem [61]:

$$minimize_{w,b}(1/2) * ||w||^2 \tag{3.11}$$

$$Subject\ to : y_i(w^T * x + b) \geqslant 1, i = 1, ...n$$

Since the assumption is the data are linearly separable, the constraints in this formulation guarantee that the maximum margin classifier classifies each example correctly. But in

practice, data are often not linearly separable; and even if it is, a larger margin can be get by allowing the classifier to misclassify some points [60, 61]. To allow errors, equation 3.11 is replaced with equation 3.12.

$$y_i(w^T * x + b) \geqslant 1 - \xi_i \tag{3.12}$$

Where ξ_{i} are slack variables that allow an example to be in the margin, $1 >= \xi_{\mathrm{i}} >= 0$,(also called a margin error) or misclassified ($\xi_{\mathrm{i}} >= 1$). If the value of its slack variable is greater than 1, An example will be misclassified. So, the sum of the slack variables is a bound on the number of misclassified examples. The aim of increasing the margin, which is minimizing w^2 will be increased with a term $C\sum_i \xi_{\mathrm{i}}$ to correct misclassification and margin errors. The optimization problem becomes

$$minimize_{w,b}(1/2) * ||w||^2 + C\sum_i \xi_i \tag{3.13}$$

$$Subject\ to : y_i(w^T * x + b) \geqslant 1 - \xi_i \;\; \xi_i \geqslant 0$$

The constant C>0 sets the significance of increasing the margin and minimizing the amount of slack. This formulation is called the soft-margin SVM. It is often the data to be classified is not linear and the datasets are not separable. To handle these datasets, kernels are used to non-linearly map the input data to a high-dimensional space. The below mentioned ones are the common kernel functions used in SVM [60, 61]. Polynomial: A polynomial mapping is a popular method for non-linear modeling. The second kernel is usually preferable as it avoids problems with the hessian becoming Zero.

$$K(x_1, x_2) = (x_1^T * x_2 + 1)^p \tag{3.14}$$

Gaussian or Exponential Radial Basis Function: Radial basis functions most commonly with a Gaussian for

$$K(x_1, x_2) = exp(-||x_1 - x_2||/2Q^2) \tag{3.15}$$

Linear: The long established MLP, with a single hidden layer, also has a valid kernel representation.

$$(x_1, x_2) = (x_1^T * x_2) \tag{3.16}$$

Chapter 4

Design and Implementations

As discussed in Chapter 2, text detection and script identification had been conducted by different researchers and most of those previous attempts consider Latin script. In this thesis, therefore, Ethiopic and Latin multilingual text detection and script identification with the design goal that enables to reach accurate text detection from images and videos followed by script identification is proposed. In the following sections, the details about the techniques and the model developed for the proposed solution are described. In section 4.1, the proposed system architecture for multilingual text detection and script identification is introduced. Section 4.2, shows how to select key frames from video. Section 4.3, presents how input data is preprocessed for further activities and get ready for the next component. Section 4.4, 4.5 and 4.6, illustrates the methods we used for text detection. In section 4.7, indicates the feature extraction technique used for both text detection and script identification. Section 4.8, presents classification stage and the proposed algorithm of the system.Finally section 4.9, reveals the challenges towards recognition.

4.1 The Proposed System

The proposed method for multilingual text detection and script identification from video frames and images comprises a serious of tasks.The diagrammatical illustration of the

proposed system is shown in Figure 4.1.

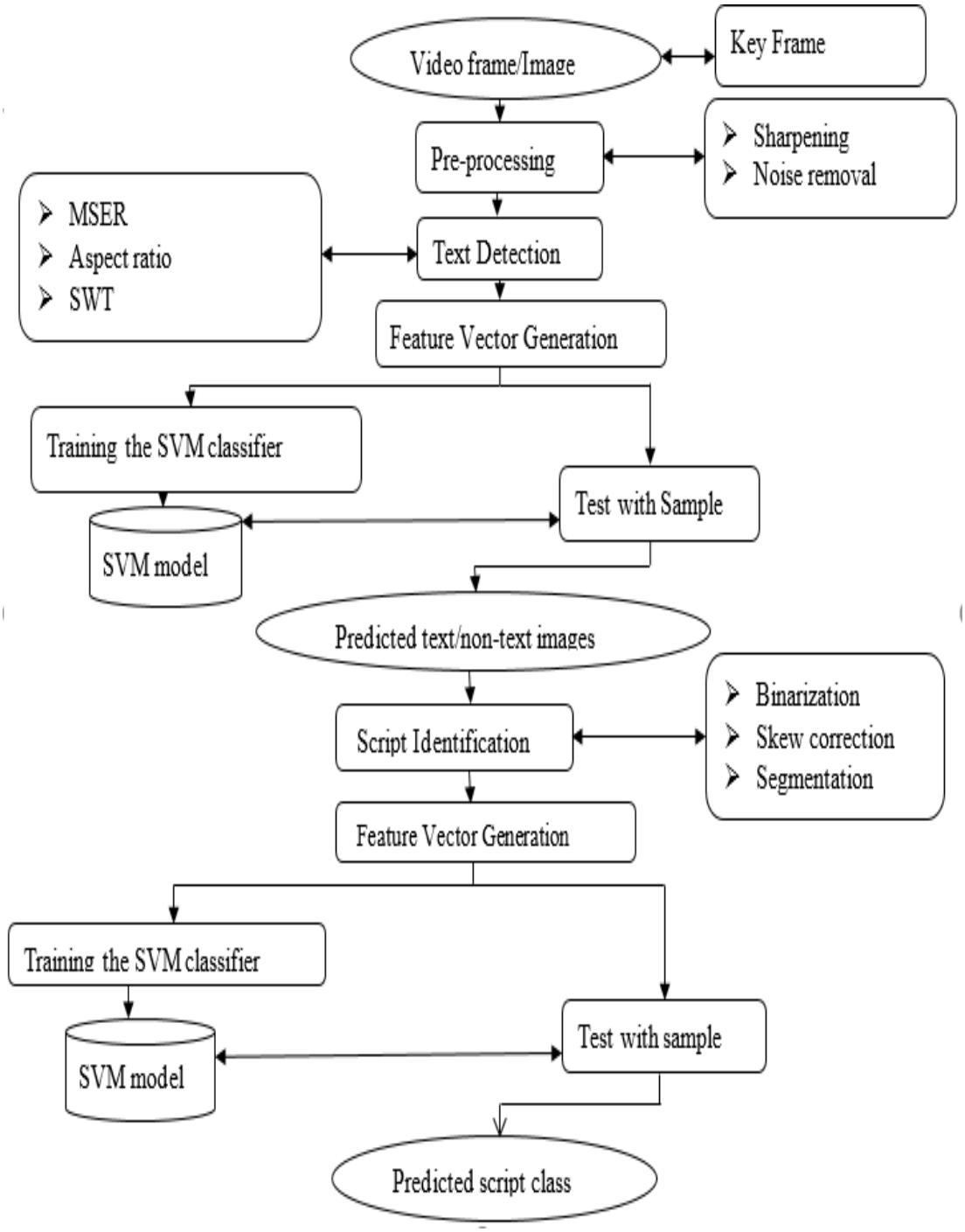


Figure 4.1: Block diagram of the proposed system

4.2 Key Frame Selection

Video is decomposed into frames. Then histogram difference algorithm is used for key frame selection. Frames subsequent to a previously extracted key frame are sequentially compared with the key frame until which is very different from the key frame is obtained.

Algorithm 1: Key frame selection using histogram difference [47].

input : Total number of frames
output: Key frames

for $i = 1 to Number of frames$ **do**
    I$\leftarrow k$;
    J$\leftarrow k+1$;
    S$\leftarrow absdiffernce[i, j]$;
end
mean$\leftarrow mean(s)$;
standard deviation$\leftarrow std(s)$;
threshold$\leftarrow standard\,deviation + (mean * k)$;
if $S < threshold$ **then**
    write image J as a key frame
end

We used the MATLAB function VideoReader to change the video to frames. Then key frame selection is performed using the histogram difference algorithm. The imhist function of MATLAB is used to find the histogram difference of consecutive frames. Also mean and std functions are utilized to find the mean and standard deviation needed respectively. The threshold value is chosen using the following formula.

$$Threshold = standard\ deviation + mean * b \tag{4.1}$$

Where, b is a constant. After trying for various values, we used value of b=4, as the results were as desired using this value. If the difference of two consecutive frames exceeds the threshold, the later frame is considered as the key frame.

4.3 Preprocessing

Nowadays there are a variety of noise filtering techniques. However, there is always a trade-off between the efficiency of a denoising technique, preservation of discontinuity and generation of artifacts in the resulting denoised image. Eliminating the noise without blurring the details too much and enhancing edges without amplifying noise is very difficult [62]. So, when using more than one filters, special care should be taken in order to make sure their effect is important.

4.3.1 Sharpening using High Pass Filter

The biggest problem of text in images and videos are out of focus, blurring and low contrast. To eliminate the effect of these problems sharpening operation is performed. Sharpening is basically a high pass procedure in the frequency domain so high pass filter is used for this operation. It works by increasing contrast along the edges of objects in an image. Increasing the contrast along the edges, making the light side of the edge lighter and the dark side of the edge darker, as contrasts increased as a result the image becomes sharper.

Algorithm 2: High pass filter [48]

input : Image
output: Sharpened version of the image
Let D_0 is the cutoff frequency,M X N is the size of the image and h is a high pass filter
for $i = N/2$ *to* $M/2$ **do**
  if $distancetransform < D_0$ **then**
    h(M,N)=0
    else h(M,N)=1 ;
  end
end

High pass filtering is performed by subtracting low pass filtered version of the image from the original image. In MATLAB Fspecial is used to find the Gaussian lowpass filter and imfilter is used to get the sharpened version of the image.

4.3.2 Noise Removal using Median Filter

Since high-pass filtering increases noise, to cope up with its effect, median filtering is performed. The median filter considers each pixel in the image and looks at its nearby neighbors to decide whether or not it is representative of its surroundings. It replaces the pixel value with the median of neighboring pixel values. The median is calculated by sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value. The sort

Algorithm 3: Standard Median filter [34]

input : Image with noise
output: Image with enhanced quality

Let p(x,y) is the current pixel under consideration and W is sliding window.
where: -M X N is the size of the image.

for $i = 1$ *to* N **do**
  for $j = 1$ *to* N **do**
    sort($W_{\mathrm{i,j}}, \ldots W_{\mathrm{M,N}}$)
    P(x,y)$\leftarrow$ *median*($W_{\mathrm{i,j}}, \ldots W_{\mathrm{M,N}}$)
  end
end

function of MATLAB is used to sort the 3X3 neighborhood of each pixel.

4.4 Maximally Stable Extremal Region

The method of MSER is based on the concept of considering regions which stay nearly the same through a wide range of thresholds. The area is scanned and regions such that their variation with respect to the threshold is minimal are defined maximally stable.

A widely used implementation of MSER detector has the following parameters:

1. Delta: Delta is defined as the change in the intensity thresholds while performing the MSER detection. An extremal region considered stable if the variation in the area at different thresholds is minimal. As delta increases, the change in the area of the detected extremal regions will be higher and hence it will reduce the number of stable extremal regions. A small value for delta (delta=4) is chosen in order to detect low contrast characters in the image.

2. Minimum Area: This parameter decides the minimum area of an extremal region. All the regions with the area less than this parameter will be discarded. Minimum area of 8 is chosen.

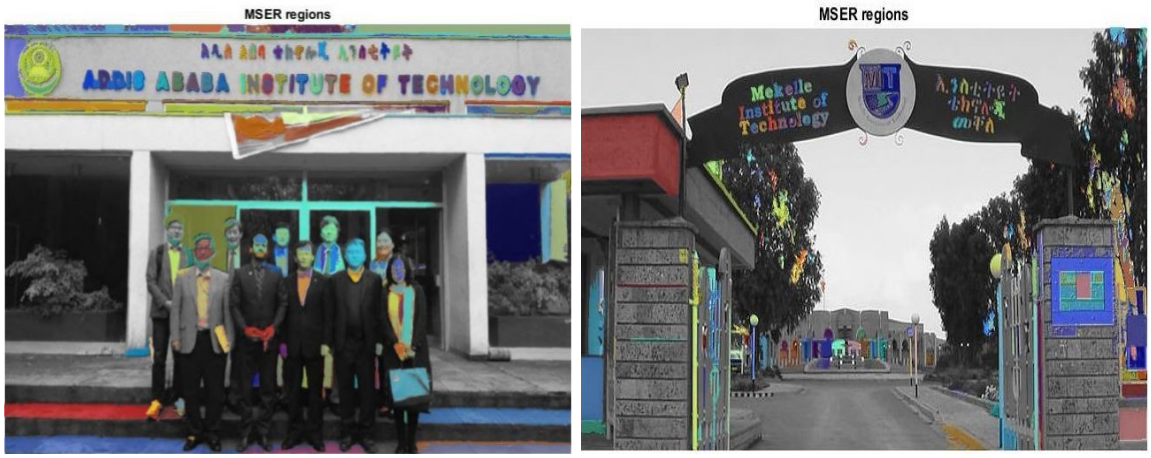
3. Maximum Area: This parameter decides the maximum area of an extremal region. All the regions with the area greater than this parameter will be discarded.Maximum area of 8000 is selected.

4. Maximum Variation: An extremal region is defined as maximally stable if the variation in the area is less than the value specified by this parameter. The value for this parameter ranges from 0.1 to 1.0. When the value of this parameter is lower, then the number of extremal regions becoming maximally stable will be less, and if the value for this parameter is higher, the number of MSER detected will be higher. The value of this parameter chosen is 0.25 to make sure even the low contrast character regions are also detected.

Algorithm 4: Union-find based MSER algorithm [50]

input : Intensity image
output: set of Maximally Stable Extremal Regions

P=BinSort(I)
for *each pixel* $p \varepsilon P$ *do* **do**
| R =Find Stable Regions using union find algorithm
end



(a) For objective evaluation (b) For subjective evaluation

Figure 4.2: MSER detection result.

The DetectMSERFeatures function of MATLAB is employed to find stable regions of the image.

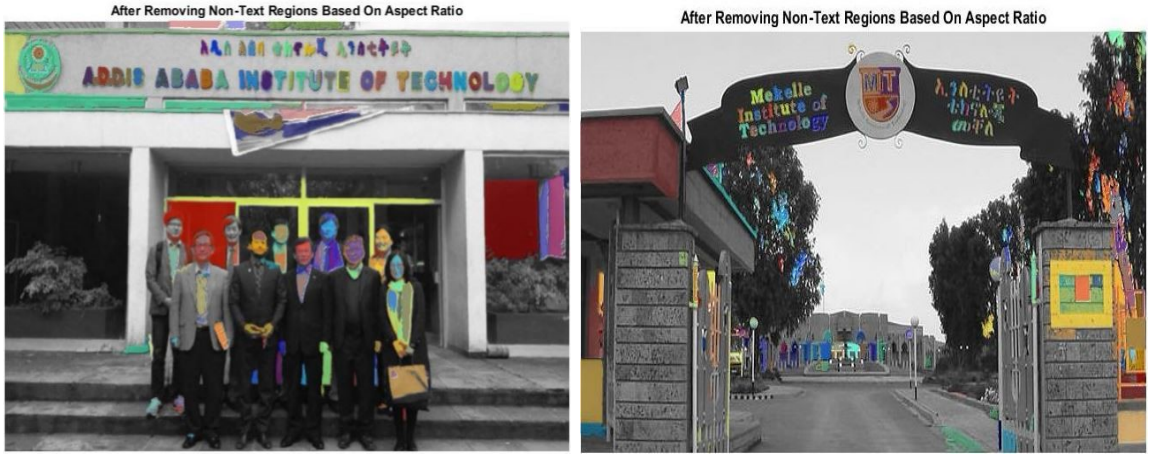
4.5 Aspect Ratio

This property is chosen because text components have regular shape and reasonable size. Aspect ratio is an effective feature for distinguishing between text and non-text components. But when using aspect ratio to filter out text regions, to handle elongated characters like ”i” or ”I” in Latin and ”ne”, ”pe”, ”ye”, ”gou” for Ethiopic a threshold should be small enough. The following threshold is used to discriminate text and non-text components; with aspect ratio greater than 10 and less than 1/10 is discarded. The biggest threshold is used to eliminate big candidate regions while the small threshold is used to discard non-text regions like wires and leaves.

Algorithm 5: Aspect ratio algorithm [34]

input : Text character candidates
output: Refined text regions

Read coordinate values of the candidate characters
for *each character i in candidate region image* **do**
    $CONN_i \leftarrow connectedcomponent(i)$;
    $w \leftarrow width(CONN_i)$;
    $h \leftarrow height(CONN_i)$;
    aspect ratio($CONN_i$)$\leftarrow$ w/h;
end



(a) For objective evaluation (b) For subjective evaluation

Figure 4.3: After removing non-text regions based on aspect ratio.

For this the regionprops function of MATLAB is used. Then the aspect ratio of each connected component is calculated form the BoundingBox property of the function.

4.6 The Stroke Width Transform

The basic inspiration of the researchers to use stroke width extraction algorithm is that stroke width almost remains the same in a single character; however, there is significant change in stroke width in non-text regions as a result of their irregularity. The initial step of stroke width extraction is to get skeletons of the MSERs remained. On every

foreground pixel on the skeleton, distance transform is applied to compute the euclidean distance from this pixel to the nearest boundary of the corresponding MSER. Then we obtain a skeleton-distance map. If the standard deviation of the stroke width of the character candidates is large, it is less likely to be a true character. As it can be seen in Figure 4.4, some false positives are eliminated after this procedure. A value of 0.5 is chosen as a threshold for removal of non-text regions.

$$Threshold = \frac{standdard\,deviation_{\text{stroke width}}}{mean_{\text{stroke width}}} \tag{4.2}$$

Algorithm 6: Algorithm of SWT [28]

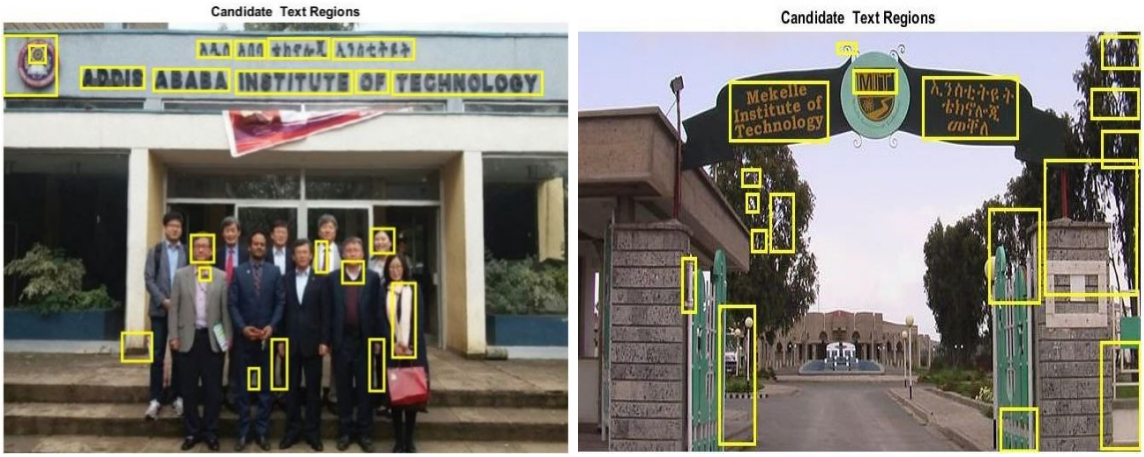
input : Text candidate image I
output: Refined text regions

For each MSER regions it finds out the stroke width image
D=DistanceTransform(BW)
D=round(D)
for *each foreground pixel in D* **do**
  pVal=D(p) LookUp(p)= neighbours of p whose value < pVal
end
MaxStrok = max(D)
for *Stroke* = 1 *to MaxStroke* **do**
  StrokeIndex=find(D==Stroke)
  NeighborIndex=Lookup(StrokeIndex)
  for *each NeighborInndex* **do**
    D(NeighborIndex)= Stroke;
    NeighborIndex = Lookup(NeighbourIndex);
  end
end
SW=D

(a) For objective evaluation (b) For subjective evaluation

Figure 4.4: After removing non-text regions based on stroke width transform.

The MATLAB Bwmorph function has been used to find the skeleton of the image and bwdist to compute the distance transform of the image. In order to distinguish the text regions from not text regions, mean and std functions of MATLAB are applied respectively. Even after the stroke width filtering is performed, regions which are not text are also detected as text region candidates. We first merge the individual characters into a single connected component using a small bounding box threshold which is 0.02. To eliminate those candidate regions which are not text, each of candidate text regions within bounding box, as shown in Figure 4.5, are cropped and their respective feature vector is computed with LBP which is the input for SVM. Finally depending on the feature vector, the SVM classifier classified each region as text and non-text. Those regions which are classified as text will be the output of the system.



(a) For objective evaluation (b) For subjective evaluation

Figure 4.5: Candidate text regions.

4.7 Feature Extraction

LBP which is a feature extraction algorithm is chosen because one of the aspects of text is that it has a unique pattern for each character. These can be used for constructing a feature descriptor for a window. The LBP operator assigns a decimal number for each pixels of an image which, called LBP codes. This pattern is used to encode the local structure around each pixel. Each pixel is compared with its eight neighbors in a 3 x 3 neighborhood by subtracting the center pixel value. The results which are less than the center pixel value are encoded with 0 and others with 1. A binary number is obtained by concatenating all these binary codes in a clockwise direction starting from the top-left one and its corresponding decimal value is used for labeling. The derived binary numbers are referred to as LBP codes. In this work, ExtractLBPFeatures function which is found in MATLAB is used to extract the texture feature vector.

Algorithm 7: Algorithm of LBP [59]

input : Text candidate image
output: Feature vector

Let M X N is the size of the image
for $i = 1$ *to* M **do**
  for $j = 1$ *to* N **do**
    initialize the pattern histogram H=0
    for *each center pixel* $P(x,y) \varepsilon I$ **do**
      P(x,y)$\leftarrow$ $LBP(P(M,N))$;
      if $neighborValue > Value\ of P(x,y)$ **then**
        setBit value=1
        else setBit value=0 ;
      end
    end
  end
end

4.8 Classification

In this research work,we used SVM algorithm for both text detection and script identification from images and video frames. Both text detection and script identification phase of the research is a two-class problem. The fitcsvm function of MATLAB is used for classification. To use SVM for classification the following steps are recommended by [63].

1. Transform data to the format of an SVM package.
2. Conduct simple scaling on the data.
3. Consider the RBF kernel.
4. Use cross validation to find the best parameter c and γ.
5. Use the best parameter c and γ to train the whole training set.
6. Test.

SVM requires that each data instance is represented as a vector of real numbers. Since the features extracted in the case of this study are all real valued there is no need for

transformation. The next step is scaling. SVMs assume that the data it executed is within a standard range; usually either 0 to 1 or -1 to 1. So, it is must to make sure that for each dimension, the values are scaled to lie roughly within this range. The main advantage of scaling is to avoid attributes in greater numeric ranges from dominating those in smaller numeric ranges. Another advantage is to avoid numerical difficulties during the calculation. So, the default standardize parameter of the fitcsvm function is set to true. Thirdly, model selection is performed. Even though there are common kernels used in SVM, we must decide which one to try first. The RBF kernel is often recommended for first consideration in classification [63]. The RBF kernel chosen due to its ability to handle classification problems with non-linear boundary. Then, after experimenting on several values of Gamma in grid search, we set the value of gamma (γ) to 71. Finally, to obtain the classification results,10-fold cross validation has been used. First, creation of randomly generated 10-fold cross-validation index of the length of the size of the dataset has been done. This index contains equal proportions of the integers 1 through 10. These integers are used to define a partition of the whole dataset into 10 disjoint subsets. We used one division for testing and remaining nine divisions for training. This has been done 10 times, each time changing the testing dataset to different division and considering remaining divisions for training. Thus 10 sets of feature vectors containing training and testing dataset in the size ratio of 9:1 has been got. The built-in function crossval of MATLAB is used to perform cross validation. Finally, testing is performed using test dataset.

4.9 Script Identification

After text detection is performed the next task of this work is script identification. Script identification can be done since the output of the system which is the localized text regions may contain multiple lines of text with different scripts which can be applied

as input for script identifier. It can be done in two ways. One; it is possible to perform script identification directly on the detected color text blocks. The other; since the goal of the thesis is to finish all the preprocessing steps of OCR, so that the final output of our system can be directly forwarded to the OCR system. To perform the script identification phase, stage-by-stage segmentation is used. The following operations performed respectively to make the OCR operation easy.

4.9.1 Binarization

For document images the simplest way of image binarization is to choose a threshold value and classify all pixels with values above the threshold as white, and all other pixels as black. But in video frames and images, however text often appears against complex, non-uniform backgrounds. The text color may also vary due to uneven illumination of a text, anti-aliasing or due to bleeding caused by video compression. These problems are further compounded by the low resolution of video images.

Therefore,Niblack's binarization algorithm is chosen due to its ability of handling color images. Instead of calculating a single global threshold for the entire image, several thresholds are calculated for every pixel by using specific formulae. The method uses the mean and standard deviation of the local neighborhood, defined by a window centered around the pixel.

Algorithm 8: Algorithm of Niblack's binarization [64]

input : Color image
output: Binary image

For each pixel in the image
if $pixeValue > (mean + k * standard\,deviation)$ **then**
    text =pixel
    else pixel=background ;
end

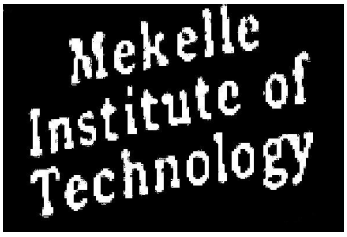


Figure 4.6: Binarization of skewed image taken from subjective evaluation test.

4.9.2 Skew Detection and Correction

Most OCR systems typically assume that images were printed with a single direction of the text. So that both the process of capturing and the acquisition process did not introduce a relevant skew. But the process of taking images using camera made the above assumption wrong and the detection and correction of skew became mandatory. For skew detection and correction Radon transform is selected due to its efficiency and less execution time.

Algorithm 9: Algorithm of Radon transforms [57]

input : Skewed image
output: Skew corrected image

apply cany edge detector
apply radon transform
find the value of theta for maximum intensity
calculate the skew angle using:
$\theta = \theta, if\, \theta \leq 90^{\circ}$
$\theta = \theta - 180^{\circ}, otherwise$

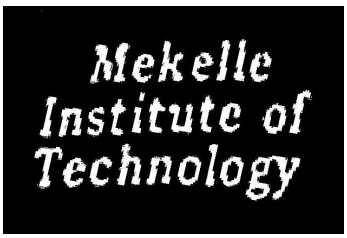


Figure 4.7: After correction.

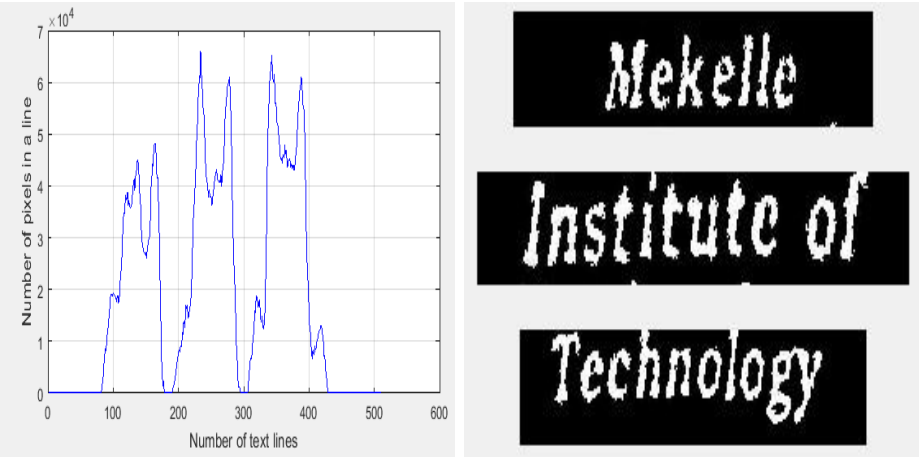
4.9.3 Text Line Segmentation

After the binarization, the next step is to extract individual text lines for feature extraction. To segment the image into text lines, the valley of the horizontal projection computed by a row wise sum of black pixels is used. The space between two consecutive horizontal projections where the histogram height is minimum denoting one boundary line. Using these boundary lines, the image is segmented into its text lines.

Algorithm 10: Algorithm of Text line segmentation [12]

input : Image with more than 1 line of sentence
output: Each line segmented

Plot the projection profile and scan the image horizontally the value pixel is 0 where is background,1 where there are letters.
Letterlocations=horizontalprojection >0
StartingRow=find(the value of apixel>0)
EndingRow=find(the value of apixel=0)
for $K =: to\ length(projection\ profile)$ **do**
| crop out each line using startingRow and EndingRow
end



(a) Horizontal projection profile of an image (b) Segemeted Text line segmentation

Figure 4.8: Text line segmentation using horizontal projection profile.

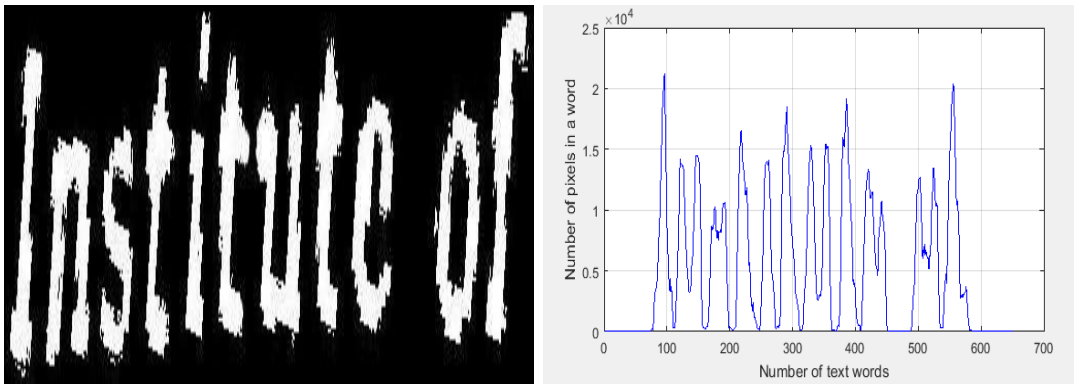
4.9.4 Text Word Segmentation

Word segmentation process is the most important one in the OCR system. Because after word segmentation, the resulting words can be used as an input for character segmentation which will be directly entered to the recognition system. The basic algorithm is based on the vertical projection of a segmented line; we will look for space between rising edge and falling edge of the space in words. Then after getting the space between the words, by using the space in the vertical projection profile each word is segmented. As each word in a line is separated with a distance we use that concept for word segmentation. We scan the segmented line image vertically, then when there is a gap the projection becomes 0. Then we can crop out each word using the point when the projection touches ground.

Algorithm 11: Algorithm of Text word segmentation [12]

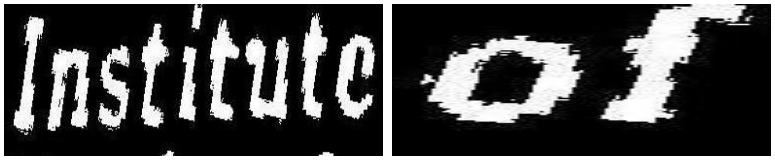
input : Image with more than 1 word sentence
output: Each word segmented

Plot the projection profile and scan the image vertically the value pixel is 0 where is background,1 where there are letters.
Letterlocations=verticalprojection >0
StartingRow=find(the value of apixel>0)
EndingRow=find(the value of apixel=0)
for $K =: to\ length(vertical projection profile)$ **do**
| crop out each word using startingRow and EndingRow
end



(a) Text line with more than 1 word (b) Vertical projection profile of words

Figure 4.9: Text word segmentation using vertical projection profile.



(a) Segmented word (b) Segmented word

Figure 4.10: Segmented word.

Finally, to perform the script identification the feature vector of each segmented word is computed using LBP and classification is performed using SVM to their respective script class.

Chapter 5

Experimental Results and Discussion

In this section, we describe our evaluation of the text detection and script identification of the system. In addition, we show how our system performs compared to other text detection systems when measured against standard benchmarks. The proposed method is implemented using MATLAB R2016a on a system with Intel R CoreTM i7 CPU@2.00GHz , 4GB RAM and 64-bit windows operating system.

5.1 Dataset and Evaluation

5.1.1 Dataset

As far as we know, there is no work done in detection and script identification of Ethiopic text from image and videos. Due to this there is no available Ethiopic text image database used for conducting an experiment. So, we prepare multilingual scene and caption text dataset. The dataset contains 400 images, which are taken from video containing 50 caption images and 350 scene images and street scenes using a pocket camera. For each word within each image, the ground truth (for detection) is a manually labeled and prepared. The scene text in the images is representative of common text found in streets or shops in different part of the country, mainly guide boards and billboards with complex background. The ICDAR images are taken from multiple sources, and include images of book covers, street signs, as well as store signs.

So, our dataset is prepared in the way to be as good as ICDAR dataset. Images should contain diversity of texts and the complexity of the background in the images should be high to make sure it is challenging. The text may be in different languages (Amharic, Tigrigna, Afaan Oromo, English or mixture of both), fonts, sizes, colors and orientations. The background may contain vegetation (e.g. trees with leaves) and repeated patterns (e.g. windows and bricks).

The second dataset used is the ICDAR 2003 dataset which evolves from a series of robust reading competitions held by ICDAR. The dataset consists of 462 images including 229 for training and 233 for testing. For each word within each image, the ground truth is fully annotated.

5.1.2 Evaluation

We set two types of evaluation units in the dataset; objective and subjective evaluations. Objective evaluation, the word level as shown in Figure 5.1(a), is used in the ICDAR dataset. However sometimes, it is hard to partition text regions within an image into individual words based on their spacing; it is almost impossible and non-trivial to perform word partition. Therefore, we consider subjective evaluation, the evaluation metric to use regions which contain the text rather than a word as shown in Figure 5.1(b).

(a) Sample ground truth for objective evaluation (b) Sample dataset for subjective evaluation

Figure 5.1: A multilingual scene text example.

The performance of our approach for both objective and subjective evaluations is evaluated by measuring its detection rate, based on precision, recall rate, and F-measure. Precision is defined as the number of true positives (TP) over the number of true positives plus the number of false positives (FP).

$$Precision = TP/(TP+FP) \tag{5.1}$$

Recall is defined as the number of true positives (TP) over the number of true positives plus the number of false negatives (FN).

$$Recall = TP/(TP+FN) \tag{5.2}$$

These quantities are also related to the (F-Measure) score, which is defined as the harmonic mean of precision and recall.

$$F-Measure = 2*((Precision*Recall)/(Precision*Recall)) \tag{5.3}$$

Except the experiment conducted in section 5.2.6, in all our experiments 10-fold cross validation is used.

5.1.2.1 Objective Evaluation

This evaluation unit is based on the ICDAR word level evaluation mechanism. DetEval software is used for evaluation. DetEval is a tool for the evaluation of object detection algorithms. It takes as input two different sets of XML files, each one describing rectangles (bounding boxes) of text locations. One of the sets corresponds to ground truth rectangles, while the other one is to detected rectangles produced by the system. These sets of rectangles are matched and the performance measures are calculated. The sample below is prepared using the format of the ICDAR text detection competition. One rectangular bounding box is described by the tag," taggedRectangle. Its geometry is described by x, y, width and height respectively.

Table 5.1: The ground truth of the sample dataset

<image>

<imageName>sen.JPG</imageName>

<resolution x="640" y="480" />

<taggedRectangles>

<taggedRectangle x="146" y="528" width="281" height="46" >

</taggedRectangles>

</image>

Table 5.2: The detected result by the system

<image>

<imageName>sen.JPG</imageName>

<resolution x=”640” y=”480” />

<taggedRectangles>

<taggedRectangle x=”135” y=”515” width=”305” height=”64” >

<taggedRectangle x=”41” y=”586” width=”489” height=”74” >

</image>

The first experiment is conducted on our prepared dataset. For both the training and testing 400-word blocks are extracted using our system. All the dataset is fully annotated using the MATLAB function of imrect to prepare the ground truth with coordinates. The 400 non-text images which is used as a negative sample is also prepared. The text candidate classifier is trained and tested on this dataset using cross validation, which gives unbiased result. Finally, DetEval software is used for objective evaluation and the following results are obtained.

Table 5.3: Text detection result with preprocessing

Precision	**Recall**	**F_measure**	**Time(sec.)**
81%	74%	77.5%	21

To realize the impact of Preprocessing phase in the performance of the system, we conduct an experiment without preprocessing stage. The results are shown as follows in the Table 5.4.

Table 5.4: Text detection result without preprocessing

Precision	**Recall**	**F_measure**	**Time(sec.)**
73%	68%	70%	15

An important observation is that the methods exhibit significantly poor performances when the text detection system is applied without preprocessing.

As far as we know there is no any work to compare our method with it by conducting an experiment with same dataset. So, to compare with other methods we conduct an experiment on the publicly accessible ICDAR 2003 dataset. We compare our approach with existing methods which conduct experiments on ICDAR 2003 dataset in Table 5.5. The comparison is made with the state of the art method developed by Boris Epshtein.The comparison was made with the result of our system with preprocessing steps.

Table 5.5: Text detection result in ICDAR dataset

.	**Criteria for Assessment**				
Method	.	**Precision**	**Recall**	**Fmeasure**	**Time(sec.)**
	Ours	78%	70%	74%	13
	Method[54s]	73%	60%	66%	0.94

The state of the art method developed in [54] uses an image operator named SWT to calculate the value of stroke width for each image pixel. It detects stroke pixels by shooting a pixel ray from one edge pixel to its opposite edge pixel along the gradient direction. The ray is considered as valid only if the gradient orientations of the pair of edge pixels are roughly opposite to each other. Otherwise, the ray is considered as invalid. All pixels covered by a valid ray are labeled by the same stroke width, which is the distance between the pair of edge pixels. Based on the fact that one of the most important features which distinguishes the text from other components in a scene image is its approximately constant stroke width transform, the SWT operator is applied to detect texts in natural scene images. In this way, SWT filters out the background pixels

and assigns text pixels with stroke widths.

Table 5.6: Execution time in ICDAR dataset

Parts of the approach	**Time(sec.)**
High pass filtering	1.2
Median filtering	5.3
MSER	2.4
Aspect ratio	1.1
SWT	1.8
Candidate linkage	0.6
Classification	0.6

The performance comparison of the proposed method with other methods in literature is given in the Table 5.5. The experimental results demonstrate the effectiveness of the proposed method. Our method performs better than the results obtained by the state of the art method [54]. We believe the improvement in the performance comes from the usage of MSER and classification stage. The text regions have been successfully extracted since, the MSER is used to generate preliminary regions. Even after non-text filtering using the Geometric property of texts called aspect ratio and SWT algorithm is performed, regions which are not a text are also detected as text region candidates. These non-character text regions are eliminated using the SVM classifier which we think, improved our performance than the state of the art method. The other thing in the classification stage is that candidate text region classifier has a very good impact on detection performance while having a very small computational cost.

To compare the proposed method with the state of the art method, we use ICDAR 2003 dataset. But there is ICDAR 2011 dataset which, we cannot get available for testing our system. However as per the results of research work given in [65] the result they obtained for ICDAR 2011 is better compare to ICDAR 2003 result.

We also obtained computational performance data and compared it with the state of the art. As shown in the result in Table 5.6, average detection time per image of our method is too slow compared to that of the state of the art. In Table 5.6, we can see that in general the proposed approach takes 13 seconds to process an image. From the different stages of the system, as it is shown clearly in the result Table 5.6, the most expensive part is the extraction of MSER and median filtering. Even though the result of our method, shows that it performs better than the state of the art but it takes a very high computation time. We believe this is due to the very noisy nature of the images in the dataset, since most of them are from real world scenes which is characterized by high blur, complex background, different font, size, color, orientation, location, position, orientation, lighting, and deformation. In addition, it is due to numerous objects, such as buildings, bikes, leaves of trees, cars or parts of them have similar shape and appearances to text with similar colors. The more the image is with all the above-mentioned characteristics the more it takes time in finding the MSER region. Another reason why our method really takes much execution time is the preprocessing stage. The filtering techniques we use to make the image more stable also takes high execution time specially the median filtering technique.

The time taken in the stage of our SWT is almost double compared to the time taken for the whole system of SWT in the state of the art method. We believe the key factor affecting the computation time of our SWT stage is due to a large number of text

candidate regions which come from the MSER and aspect ratio filtering. If the portion text candidate regions are large and noise in the image is large, it will take more time in the process of stroke width filtering.

5.1.2.2 Subjective Evaluation

For most images as shown in Figure 5.1(b), it is hard to partition and prepare a ground truth for a word text in images. As a result, even if the detector successfully identified the text in the image, the match scores between a bonding box for an entire block of text and bounding box for a single word tended to be very low. In this evaluation mechanism there is no ground truths consisted of bounding boxes for individual words, while our detection system outputted bounding boxes for entire group of text regions. We consider the evaluation metric to use regions which contain the text rather than a word level objective evaluation mechanism. The performance measure in the subjective evaluation is its detection rate, defined as the ratio between the number of detected text images and all the given images containing text. The performance is merely inferred from the OCR results, as the text extraction performance is closely related to the OCR output. After performing a text detection, we prepared dataset of 800 images. The text containing image set contains 100 caption and 300 scene images and the non-text images contains 400 randomly selected images from different sources.

Table 5.7: Text detection result for subjective evalaution

Precision	**Recall**	**F_measure**	**Time(sec.)**
97%	84%	90.5%	19

Table 5.7 illustrates the performance of the subjective evaluation of our approach for the text detection in our prepared dataset. It is important to note that, these results represent the ability of our method to find a text instance in the image.The precision

term represents the percentage of reasonably accurate text localizations,while the recall term represents the methods ability of finding the text regions. However, as each unit block is determined as text or non-text, precise localization results could not be generated. The detected text region may contain single line or multiple lines. When it consists more than one line of text in the image,it is later used in line and word segmentation for script identification.

5.2 Script Identification

We also presented a script identification module that takes text blocks as input and identifies the script of the text. Each script is viewed as a different texture and the texture information is captured by computing the histogram of local binary patterns. Identification is carried out by an SVM trained on text blocks from the two scripts which are Latin and Ethiopic. The main idea of this module is to identify the script of the text rectangles detected in the images so that these rectangles can be further processed by their respective recognition engines. We tried to investigate and evaluate our approach in two ways which are end-to end pipeline and cropped word script identification.

5.2.1 End-to End Script Identification

This task combines all the preprocessing steps needed for multilingual text script identification. It takes an input image, finds the bounding boxes of all the words, and then discriminates each word in to its respective script class. Due to the challenges of text detection from images and videos our system fails to detect some text regions and produces some non-text regions with text bounding boxes. We give all the text images which comes as output of the text detection stage bounding boxes to script identifier and examine its capability of distinguishing scripts.

The dataset prepared for experimentation comes from the result of text detection both in multilingual and monolingual images. Most words appear to come from scene text which is 600 in number, but there are also images that appear to be caption text which is the remaining 200. We have excluded the words that have mixed script for this task due to false detection bounding boxes. The dataset is diverse and challenging since the images are captured from ICDAR 2003 dataset for Latin, and from different locations for Ethiopic script under different imaging conditions. Majority of the images are with low resolution, noise or blur. We evaluate the performance of our script identifier based on the measure of accuracy. Which is defined as follows:

$$Accuracy = ((Correctly\ Identified\ Words)/(Total\ Number\ of\ Words)) * 100 \qquad (5.4)$$

Table 5.8: Confusion matrix and accuracy of end to end color image script identification

	Confusion matrix		**Overall accuracy%**
	Ethiopic	Latin	
Ethiopic	330	70	79.9%
Latin	91	309	

5.2.2 Cropped Script Identification

Since in a practical system, input texts are usually detected by a text detector, which would sometimes fail to give faultless bounding boxes. To test the impact of poor text detection in the performance of script identification, we prepare a word dataset by cropping word images from original text images. We first harvest a collection of images from ICDAR 2003 dataset for Latin script and from our prepared dataset for Ethiopic script. Text word images are then cropped manually by hand from these images. For each script category, 300 scene and 100 caption word images are collected. The cropped

word images contained two or more characters with variable space between characters. All the words are also color images and cross validation was used.

Table 5.9: Confusion matrix and accuracy of cropped color images

	Confusion matrix		**Overall accuracy%**
	Ethiopic	Latin	85.1%
Ethiopic	355	45	
Latin	74	326	

5.2.3 Challenges

Script identification from images and videos is difficult because of the following reasons: Text in images and videos frequently spotted on complex background such as outdoor sign-boards and hoardings, written in different fonts and styles. Fonts and color of the text have large variations. The other challenge we face is the quality of the image is often degraded by distortions such as low resolutions, noises and varying light conditions. Finally, Ethiopic and Latin scripts share a subset of characters that have nearly the same structural layout. Further investigation was done to understand the reasons behind the mis-classification. The low resolution and blurred images of the words of both scripts were separated to form a dataset. Additional dataset was formed by selecting the sharp and better resolution images of the two scripts. In total the low resolution and blurred image dataset was formed using 200-word images comprising of 100 Latin and 100 Ethiopic words. While, the better resolution and sharp image dataset comprised of 200 words, having 100 Latin and 100 Ethiopic words.

Table 5.10: Confusion matrix and accuracy for good quality images

	Confusion matrix		**Overall accuracy%**
	Ethiopic	Latin	
Ethiopic	78	22	79%
Latin	20	80	

Table 5.11: Confusion matrix and accuracy for low quality images

	Confusion matrix		**Overall accuracy%**
	Ethiopic	Latin	
Ethiopic	68	32	69.5%
Latin	29	71	

5.2.4 Full Pipeline Script Identification

Image binarization plays a vital role in text segmentation which is used in OCR systems and improve the recognition accuracy. Due to improper binarization, important information may get lost unintentionally which will result poor script identification. As the result, the classifier may not perform well because the feature extractor may not describe the texture of the scripts properly. Therefore, to observe the impact of the binarization module in the performance of script identification, we perform two experiments. First, we conduct end to end script identification which includes our binarization module. Then we conduct script identification using cropped clear binarized images and compare the results.

For the end to end with binarization script identification the datasets of image sets were extracted using our text detection method. After text detection, to perform the script identification phase, stage-by-stage segmentation is used. Here, it should be noted that

to perform script identification after text detection the following post processing tasks are performed.

- Binarization: It is the process of converting a pixel image to a binary image.
- Skew detection and correction: The tilt angle induced during capturing an image and videos can adversely affect the overall process of recognition and significantly complicate the segmentation of line and words in the image. Hence it is required to detect and correct the skew angle before proceeding to further steps in OCR.
- When our image is from subjective evaluation which is difficult to prepare a ground truth, it may contain several text lines near each other. So, we have to perform line segmentation, then word segmentation from the lines which contain more than a word in it using our word segmentation technique.

We use the same dataset with the end to end color text script identification. In addition to this we also include text detection results from subjective evaluation to see the effect of skew correction, line segmentation and word segmentation. The dataset comprised of 400 Latin and 400 Ethiopic words. For all the experiments we apply cross validation technique to compute the script identification accuracy. The reason for using cross validation is that it provided unbiased results over the complete dataset.

Table 5.12: Confusion matrix and accuracy Full Pipeline Script Identification

	Confusion matrix		**Overall accuracy%**
	Ethiopic	Latin	
Ethiopic	289	111	75.2%
Latin	88	312	

Table 5.13: Confusion matrix and accuracy cropped binary image script identification

	Confusion matrix		**Overall accuracy%**	Time in seconds
	Ethiopic	Latin	94.2%	1.1
Ethiopic	380	20		
Latin	26	374		

5.2.5 Performance of Skew Correction Method

Script identification is performed mainly to avoid human interference in OCR and to increase the accuracy of OCR. The skew makes more difficult to visualize the images and it increases the complexity of any kind of automatic image recognition, which degrades the performance of OCR. Different researchers use different evaluation metrics. But when evaluating skew correction algorithm, it is sufficient to detect and correct the angle of inclination so that line segmentation and word segmentation can be done effectively.

We prepare a dataset of 20 binary images with horizontal orientation without skew. Then we tilt the images with different angles using imrotate function of MATLAB. Finally, we give for the skew detection and correction system and see the results in terms of accuracy and execution time. We get average accuracy value of 98%. Even though there is adrift of 2% in the accuracy of the detection and correction method we feed the corrected image for line segmentation. But there is no any problem in the accuracy of the segmentation which shows the skew correction method is effective. The average skew correction time of our method for an image is 5.4 seconds.

5.2.6 The Impact of Cross Validation

As it is stated in each experiment, cross validation is used for evaluation to get unbiased results. The final experiment is conducted to see the effect of cross validation in the performance of the classifier. Instead of using cross validation, like in all experiments done before, we employ conventional validation which is train/test model evaluation. The biggest challenge here is finding optimal splitting proportion of training and testing sets. A common study trend is to split the sample into training set and an independent test set, where the former used to develop the classifier and the later to evaluate its performance. The dataset is the same dataset used in cropped binary image script identification, which are 400 words for each script. We split the dataset in two pieces, sets of 70% for training and 30% for test so that the model can be trained and tested on different data.

Table 5.14: Confusion matrix and accuracy of cropped binary image without cross validation

	Confusion matrix		**Overall accuracy%**	Time in seconds
	Ethiopic	Latin		
Ethiopic	343	57	84.6%	0.2
Latin	66	334		

As the result Table 5.14 without cross validation and the result in Table 5.13 with cross validation describes a fair way to properly estimate model prediction performance is to use cross validation as a powerful general technique. For the purpose of validation we use 10-fold cross validation.

5.3 Answers to the Research Questions

According to chapter 1, there are three research questions generally concerning the improvement of text detection and script identification. In this section, we briefly recall each question and provide adequate answers based on the research work.

RQ1. Can preprocessing the target image or video with image enhancement techniques improve performance of text detection?
The RQ1 is mainly about obtaining a high accuracy in the case of complex text images. To answer this research question, we implemented two effective preprocessing filters, namely the sharpening using high pass filter and noise removal using median filter to make a better text detection system. The importance of the preprocessing stage of text detection system lies in its ability to remedy some of the problems that may occur because of the built-in factors of text images. Therefore, using effective preprocessing algorithm makes the detection system more robust. To enhance image that has been blurred, either in error, as a natural effect, sharpening is done. Since high-pass filtering increases noise in the sharpening operation to cope up with its effect and for eliminating noise and preserving edges and fine details of images median filtering is performed. Finally, we gain the performance increase of 8% in precision, 6% in recall and f-measure of 7.5% due to the addition of the preprocessing stages as shown in Table 5.3 and Table 5.4.

RQ2. Can the combination of rule based and machine learning approaches achieve competitive performance for text detection from images and video frames?
We start by using MSER algorithm to find text regions. Even though MSER finds most of the text regions, it also detects many other stable regions in the image that are not

text. We eliminate non-text regions step by step. First, we use rule-based approach which uses geometric Property called aspect ratio Followed by non-text region refining using SWT algorithm. Then we applied machine learning approach to verify candidate text regions as text and non-text. As results in Table 5.5 shows the combination of the two approaches yields better result compared to the state of the art method.

RQ3. Can the combination of texture features with machine learning algorithm can be used for Ethiopic, Latin script identification of words with blurring, noise, less contrast and complex background?

To answer the last question which is RQ3, we used LBP and SVM for feature extraction and classification respectively. Feature extraction is one of the basic functions of script identification. It involves measuring those features of the input pattern which are relevant to the classification. It should also be robust and easy to compute. So, we use LBP as feature extraction algorithm for script identification. The main task of classification is to use the feature vectors provided by feature extraction algorithm to assign the text to a category of their script. It is very important to choose appropriate feature extraction and classifier algorithm which can perform best in categorizing the input word image text into their respective class of script. We used LBP for feature extraction and SVM as classifier for script identification.

5.4 Discussion

1. The experimental results demonstrate the effectiveness of the proposed method. The performance of any method is generally dependent on the quality of text present in the image. Images with text in simple colors and layout can be effectively detected due to the ability of the method to successfully extract characters as components. Further, it is observed that the method is able to localize text regions of different scripts in

both natural scene and caption texts taken from camera and video, which proves the robustness of the proposed method.

2. As the result of the proposed method is compared with that of the state of the art method, as shown in Table 5.5, it clearly shows that the performance of the proposed system is better in detection quality but slower in detection speed. The key factor affecting the computation time of our approach is the number of detected candidate characters. If the portion of text and noise in the image is large, it will take more time in the stage of MSER text candidate detection. This makes the refinement process also slows down.

3. Experiments without any pre-processing resulted in a comparatively lower accuracy as compared to the accuracy obtained using preprocessing techniques. This happens because when images suffer from low resolution, blur, complex backgrounds, the preprocessing stage helps to obtain a picture with more stable regions which in turn results an increase the accuracy of the system. As shown in the Figure 5.2(b), the system couldn't detect the text regions before preprocessing is performed. As it can be seen in Figure 5.2(c), the quality of the image is enhanced; and hence the ability of the text detection is also improved as shown in Figure 5.2(d).

4. MSER segmentation results were found to be insensitive to changes in MaxVaration and MinDiversity. However, segmentation results were sensitive to changes in delta, and if delta was set to low it could result in over-segmentation or if delta was set too high it could have resulted in under segmentation. When delta is set to low, more non-text components are extracted as text candidate regions, which makes the non-text filtering more difficult since it adds several false positives. In addition, it decreases the accuracy of the system especially in case of objective evaluation.

(a) Image before preprocessing (b) Result before preprocessing

(c) Image after preprocessing (d) Result after preprocessing

Figure 5.2: The effect of preprocessing on the result of text detection.

As we can see from Figure 5.3 (b), when we set the delta parameter of MSER to small value as in Figure 5.3 (a) it returns several text candidates. Non-character regions can be also classified as character regions (false positives), and sometimes these regions can be attached to nearby text regions. These create bigger bounding box, as shown in the Figure 5.3 (b), of candidate text regions of delta =1 which will make the detection count as incorrect one in case of objective evaluation. When delta =4 the number of candidate text regions decreases as it can be seen in Figure 5.3(d). Separating text word when there is a touching of characters between text of words to fix bounding box is also another issue especially in skewed text image.

(a) MSER Regions when Delta =1 (b) Result of text detection when Delta =1

(c) MSER Regions when Delta =4 (d) Result of text detection when Delta 4

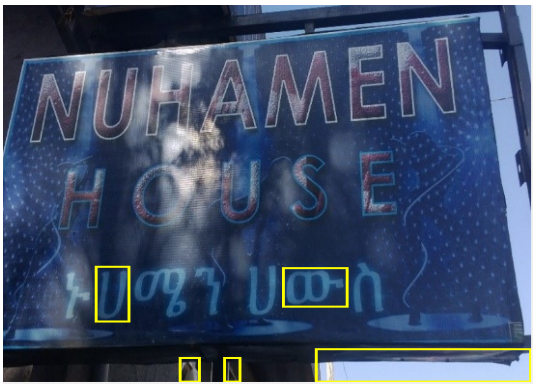
Figure 5.3: The effect of delta on the result of text detection through MSER.

5. As the results in the subjective and objective evaluations are noted, there is a big performance difference between the objective evaluation and the subjective evaluation results. Such a result is expected,because the text detection method simply extracts text like structures in the image. But the evaluation methods consider the detected boundary of text regions. So, even though the method may have actually found the text, as shown by the subjective evaluation results, it is unfairly penalized in the strict objective evaluation.

6. As the detection results reveal, the most important factors that affect the performance of the text detection method is complexity of background, low contrast, blur, specular reflection, and irregular text structure. In Figure 5.4 (a) the system couldn't detect text

regions due to the complexity of the background and low contrast of the text with its background. Figure 5.4 (b) shows an incomplete detection due to light reflection. Due to the reflection multiple characters in a word are not detected and classified as non-characters. This will produce two text detections in different parts of a word, and they will be counted as in correct detections as shown in Figure 5.4 (b). The other biggest problem in text detection comes from regular backgrounds. Several objects, such as buildings, bricks, leaves, windows or parts of them have similar shape and appearances to text. Since they have similar textual structures to text both the rule based and the machine learning phase of the system failed to distinguish these type of non-text backgrounds from texts, as seen in Figure 5.4 (c). Non-character regions are classified as character regions and sometimes these regions can also be attached to nearby text regions, creating bigger bounding boxes, which will make the detection count as incorrect as shown in Figure 5.4 (c).

7. These experiments revealed cross validation increases the discriminative capacity of the classifier when applied on the same dataset. This performance gain is due to train/test validation procedure does not use all the available data and the results are highly dependent on the choice for the training/test split. The instances chosen for the test set may be easy or too difficult to classify and this can skew the results. Furthermore, the data in the test set may be valuable for training so train/test validation performance may suffer, again leading to skewed result. So, since the performance estimate is less sensitive to the partitioning of the data in cross validation it performs better. The other thing to notice here is, the time in cross validation is slower than the normal train/test validation. This happens because in cross validation, the training algorithm has to be rerun from scratch 10 times. This means it takes 10 times as much computation to make an evaluation. Even though the time of cross validation is not



(a) Complete failure

(b) Incomplete detection from ICDAR 2003 dataset

(c) Large bounding box detection from ICDAR 2003 dataset

Figure 5.4: Error cases of text detection due to different reasons.

10 times that of the normal train/test validation, it is evident that the execution time increases due to repetition training and testing of cross validation.

8. We show results on end to end as well as cropped word script identification. We observe that despite high variations in images such as complex background, illumination change, low resolution our method is successful in its performance far better than the state of the art method used so far. We have observed that text is wrongly classified to wrong script due to the above-mentioned challenges. It should be noted that, end-to end script identification is far more challenging than script identification in cropped words. The final result shows the error collected because of incorrect text detection and

subsequent operations such as, binarization and segmentation.

Chapter 6

Conclusion and Recommendation

In this chapter we conclude this thesis by discussing the proposed method and comparisons of our proposed approach with the state of the art method. Finally, we also provide the future directions of this thesis.

6.1 Conclusion

The objective of this research work was detecting texts from images and video frames and identifying the type of script used in writing. Text detection in images and video scenes is a difficult problem. The difficulty is due to wide dissimilarity in fonts, sizes color and textures of text regions embedded in scenes. The location of the text in the image also follows a random behavior. Text can occur in any part of the image. The presence of repeating patterns and complex backgrounds in unconstrained images increase the difficulty for detecting text in video scenes. Encoding all these variabilities in a rule-based approach is extremely challenging. So, in this work, we combine rule-based approach with machine learning methods to generate a model to discriminate text and non-text regions in video frames and images.

The video containing the text is decomposed into frames. Histogram difference comparison is performed to choose the key frames. In order to minimize the effect of noise in images

and video frames, pre-processing techniques has been applied in input images and video frames. Then since text characters usually have consistent color, we begin by finding regions of similar intensities in the image using the MSER region detector. Although the MSER algorithm picks out most of the text, it also detects many other stable regions in the image that are not text. Geometric property of a text character which is aspect ratio followed by SWT are used to filter out non-text regions.After the geometric filtering and SWT is performed, regions which are not a text may be detected as text region candidates. To eliminate these candidate regions which are not text, the respective feature vector of each candidate text regions is computed by LBP. The feature vector is then used as input for SVM classifier which decides whether the candidate regions are text or not.

Script identification is very important for development of multi-script OCR systems. The next phase of the research work was script identification. A SVM based classifier for identification of word wise Latin and Ethiopic script from images and video frames was developed. After the text detection phase the feature vector of candidates which are classified as text are computed using again LBP and then feed to SVM for identification of their respective script classes.

The method performs better on text detection result using precision, recall rate, f-measure level compared with the state of the art method with precision 78%, recall of 70% and 74% f-measure on ICDAR 2003 dataset. While the state of the art method gives 73% precision, recall of 60% and 66% of f-measure. The text detection method was also evaluated on our dataset, where 81% precision,74% recall with a f- measure of 77% was obtained. The end to end script identification which integrates the text detection method performs 79.9% of accuracy.

6.2 Recommendation

This thesis opens some possibilities for future research.

- The techniques proposed in this thesis are manly designed for detection and script identification of text from images and video frames. We did text detection from video by decomposing a video into a frame.But it should be noted that applications such as translations of a text to another language must be a real time. So instead of treating the video as a sequence of images like what we have done, the detection phase should be real time. So that it can track the existence of text in live videos. Making the system real time can be interesting directions of research in the future.

- As shown in Table 5.5, MSER and median filtering takes high execution time. So, finding a way to perform parallel operation on MSER and median filtering will decrease the execution time which can be a future work.

- In both text detection and script identification phase, LBP and SVM were used to compute features and to perform classification respectively. Other feature extraction techniques such as gray level co-occurrence matrix and classifiers like deep learning should be considered in order to see their impact in the performance of the system.

- Multilingual OCR of languages which uses Latin and Ethiopic script is not done. These can be a good direction of research in the future.

References

[1] Gllavata, Julinda, Ralph Ewerth, and Bernd Freisleben. "A text detection, localization and segmentation system for OCR in images." *Multimedia Software Engineering, 2004. Proceedings. IEEE Sixth International Symposium on.* IEEE, 2004.

[2] Berclaz, Jérôme, et al. "Image-based mobile service: automatic text extraction and translation." *Proceedings of SPIE.* Vol. 7542. No. EPFL-CONF-162245. Society of Photo-Optical Instrumentation Engineers, 2010.

[3] Yang, Jie, et al. "Automatic detection and translation of text from natural scenes." *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on.* Vol. 2. IEEE, 2002.

[4] Chang, Shyang-Lih, et al. "Automatic license plate recognition." *IEEE transactions on intelligent transportation systems* 5.1 (2004): 42-53.

[5] Anagnostopoulos, Christos-Nikolaos E., et al. "License plate recognition from still images and video sequences: A survey." *IEEE Transactions on intelligent transportation systems* 9.3 (2008): 377-391.

[6] Kapse, Sneha S., and Pravin Kshirsagar. "Text Based Video Indexing and Retrieval by Using DLER Technique."

[7] Shim, Jae-Chang, Chitra Dorai, and Ruud Bolle. "Automatic text extraction from video for content-based annotation and retrieval." *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on.* Vol. 1. IEEE, 1998.

[8] Deshpande, Samruddhi, and Revati Shriram. "Real time text detection and recognition on hand held objects to assist blind people." *Automatic Control and Dynamic Optimization Techniques (ICACDOT), International Conference on.* IEEE, 2016.

[9] Ghatak, Sanjoy, and Rangpo SMIT. "KEY-FRAME EXTRACTION USING THRESHOLD TECHNIQUE."

[10] Liu, Guozhu, and Junming Zhao. "Key frame extraction from MPEG video stream." *Information Processing (ISIP), 2010 Third International Symposium on. IEEE, 2010.*

[11] Ye, Qixiang, et al. "Fast and robust text detection in images and video frames." *Image and Vision Computing* 23.6 (2005): 565-576.

[12] Vinod, H. C., S. K. Niranjan, and G. L. Anoop. "Detection, Extraction and Segmentation of Video Text in Complex Background." *International Journal on Advanced Computer Theory and Engineering* 5 (2013): 117-123.

[13] Liu, Xiaoqing, and Jagath Samarabandu. "Multiscale edge-based text extraction from complex images." *Multimedia and Expo, 2006 IEEE International Conference on*. IEEE, 2006.

[14] Anthimopoulos, Marios, Basilis Gatos, and Ioannis Pratikakis. "A two-stage scheme for text detection in video images." *Image and Vision Computing* 28.9 (2010): 1413-1426.

[15] Jain, Anil K., and Yu Zhong. "Page segmentation using texture analysis." *Pattern recognition* 29.5 (1996): 743-770.

[16] Jung, Keechul, Kwang In Kim, and Anil K. Jain. "Text information extraction in images and video: a survey." *Pattern recognition* 37.5 (2004): 977-997.

[17] Zhang, Jing, and Rangachar Kasturi. "Extraction of text objects in video documents: Recent progress." *Document Analysis Systems, 2008. DAS'08. The Eighth IAPR International Workshop on*. IEEE, 2008.

[18] Lu, Tong, et al. Video Text Detection. Springer, 2014.

[19] Li, Huiping, David Doermann, and Omid Kia. "Automatic text detection and tracking in digital video." *IEEE transactions on image processing* 9.1 (2000): 147-156.

[20] Rathod, Ganesh I., and Dipali A. Nikam. "An algorithm for shot boundary detection and key frame extraction using histogram difference." *International Journal of Emerging Technology and Advanced Engineering 3.8* (2013): 155-163.

[21] Zhang, Xin, Fuchun Sun, and Lei Gu. "A combined algorithm for video text extraction." *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*. Vol. 5. IEEE, 2010.

[22] Sharma, Nabin, et al. "Word-wise script identification from video frames." *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013.

[23] Liao, Wen-Hung, Yi-Hsuan Liang, and Yi-Chieh Wu. "An integrated approach for multilingual scene text detection." *Soft Computing and Pattern Recognition (SoCPaR), 2015 7th International Conference of*. IEEE, 2015.

[24] Gllavata, Julinda, Ralph Ewerth, and Bernd Freisleben. "A robust algorithm for text detection in images." *Image and Signal Processing and Analysis, 2003. ISPA 2003. Proceedings of the 3rd International Symposium on*. Vol. 2. IEEE, 2003.

[25] Neumann, Lukas, and Jiri Matas. "Real-time scene text localization and recognition." *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012.

[26] Kim, Kwang In, Keechul Jung, and Jin Hyung Kim. "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm." *IEEE Transactions on Pattern Analysis and Machine Intelligence*25.12 (2003): 1631-1639.

[27] Mao, Wenge, et al. "Hybrid Chinese/English text detection in images and video frames." *Pattern Recognition, 2002. Proceedings. 16th International Conference on.* Vol. 3. IEEE, 2002.

[28] Chen, Huizhong, et al. "Robust text detection in natural images with edge-enhanced maximally stable extremal regions." *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE, 2011.

[29] Kumuda, T., and L. Basavaraj. "Hybrid Approach to Extract Text in Natural Scene Images." *International Journal of Computer Applications* 142.10 (2016).

[30] Pan, Yi-Feng, Xinwen Hou, and Cheng-Lin Liu. "A hybrid approach to detect and localize texts in natural scene images." *IEEE Transactions on Image Processing* 20.3 (2011): 800-813.

[31] Thilagavathy, A., et al. "Tamil Text detection in videos."*International Journal of Engineering and Innovative Technology (IJEIT)*Volume 3, Issue 9, March 2014

[32] Phan, Trung Quy, et al. "Video script identification based on text lines." *Document Analysis and Recognition (ICDAR), 2011 International Conference on.* IEEE, 2011.

[33] SERTSE, ABEBE. "Bilingual script identification for optical character recognition of amharic and english printed document." (2011).

[34] ABEBAYEHU, SAMUEL. "Amharic-English Script Identification in Real-Life Document Images." (2012).

[35] Lienhart, Rainer, and Wolfgang Effelsberg. "Automatic text segmentation and text recognition for video indexing." *Multimedia systems* 8.1 (2000): 69-81.

[36] Lu, Tong, et al. "Video Caption Detection." Video Text Detection. Springer London, 2014. 49-80.

[37] Xu, Changsheng, et al. "Live sports event detection based on broadcast video and web-casting text." *Proceedings of the 14th ACM international conference on multimedia*. ACM, 2006.

[38] Chen, Datong, and Juergen Luettin. "A survey of text detection and recognition in images and videos." *No. EPFL-REPORT-82656. IDIAP, 2000.*

[39] Wolf, Christian, Jean-Michel Jolion, and LIRIS INSA de Lyon. "Model based text detection in images and videos: a learning approach." *Laboratoire dInfoRmatique en Images et Systemes dinformation, Palmas, TO (2004).*

[40] Anand, Mishra. "Understanding Text in Scene Images." (2016).

[41] Elagouni, Khaoula, et al. "Text recognition in videos using a recurrent connectionist approach." *International Conference on Artificial Neural Networks*. Springer, Berlin, Heidelberg, 2012.

[42] Lyu, Michael R., Jiqiang Song, and Min Cai. "A comprehensive method for multilingual video text detection, localization, and extraction." I*EEE transactions on circuits and systems for video technology* 15.2 (2005): 243-255.

[43] Yin, Xu-Cheng, et al. "Multi-orientation scene text detection with adaptive clustering." *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015): 1930-1937.

[44] Bouaziz, B., T. Zlitni, and W. Mahdi. "AViTExt: Automatic video text extraction." CoRR abs/1301.2173 (2013).

[45] Sheena, C. V., and N. K. Narayanan. "Key-frame Extraction by Analysis of Histograms of Video Frames Using Statistical Methods." *Procedia Computer Science* 70 (2015): 36-40.

[46] Liu, Guozhu, and Junming Zhao. "Key frame extraction from MPEG video stream." *Information Processing (ISIP), 2010 Third International Symposium on.* IEEE, 2010.

[47] Rathod, Ganesh I., and Dipali A. Nikam. "An algorithm for shot boundary detection and key frame extraction using histogram difference." *International Journal of Emerging Technology and Advanced Engineering 3.8* (2013): 155-163.

[48] Zafeiridis,p., et al. "A New Sharpening Technique for Medical Images using Wavelets and Image Fusion." *Journal of Engineering Science and Technology Review 9.3 (2016).*

[49] Suganthi, A., and M. Senthilmurugan. "Comparative study of various impulse noise reduction techniques." Int J Eng Res Appl 3.5 (2013): 1302-6.

[50] *Region detectors Retrieved December 23, 2017, from*`http://www.micc.unifi.it/delbimbo/wp-content/uploads/2011/03/slide_corso/A34%20MSER.pdf`.

[51] Matas, Jiri, et al. "Robust wide-baseline stereo from maximally stable extremal regions." *Image and vision computing* 22.10 (2004): 761-767.

[52] Zhou, Gang, et al. "Detecting multilingual text in natural scene." *Access Spaces (ISAS), 2011 1st International Symposium on.* IEEE, 2011.

[53] Opitz, Michael. "Text Detection and Recognition in Natural Scene Images." *Computer Vision Lab Institute of Computer Aided Automation,Vienna University of Technology,German*,Tech. Report.CVL-TR-11,September 27, 2013.

[54] Epshtein, Boris, Eyal Ofek, and Yonatan Wexler. "Detecting text in natural scenes with stroke width transform." *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.* IEEE, 2010.

[55] Yao, Cong, et al. "Detecting texts of arbitrary orientations in natural images." *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on.* IEEE, 2012.

[56] Du, Yuning, Genquan Duan, and Haizhou Ai. "Context-based text detection in natural scenes." *Image Processing (ICIP), 2012 19th IEEE International Conference on.* IEEE, 2012.

[57] Aithal, Prakash K., et al. "A fast and novel skew estimation approach using radon transform." *International Journal of Computer Information Systems and Industrial Management Applications* 5 (2013): 337-344.

[58] Sharma, Nabin, Umapada Pal, and Michael Blumenstein. "A study on word-level multi-script identification from video frames." *Neural Networks (IJCNN), 2014 International Joint Conference on.* IEEE, 2014.

[59] *Local binary patterns Retrieved December 23, 2017, from* `https://bytefish.de`.

[60] Jakkula, Vikramaditya. "Tutorial on support vector machine (svm)." *School of EECS, Washington State University* 37 (2006).

[61] Ben-Hur, Asa, and Jason Weston. "A user's guide to support vector machines." *Data mining techniques for the life sciences* (2010): 223-239.

[62] Wang, Yao. "Image filtering: noise removal, sharpening, deblurring." *Polytechnic University, Brooklyn (2006).*

[63] Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin. "A practical guide to support vector classification." (2003): 1-16.

[64] *Niblack local thresholding December 23, 2017,from* `https://www.mathworks.com`.

[65] Liu, Shuping, et al. "Text detection in natural scene images using morphological component analysis and Laplacian dictionary." IEEE/CAA *Journal of Automatica Sinica* (2017).

Appendix

```
%%%The code for the text detection phase of the system
tic
 colorImage = imread('Test image.jpg');
 I = rgb2gray(colorImage);
% Detect MSER regions.
[mserRegions, mserConnComp] = detectMSERFeatures(I, ...
    'RegionAreaRange',[1 8000],'ThresholdDelta',4);
figure
imshow(I)
hold on
plot(mserRegions, 'showPixelList', true,'showEllipses',false)
title('MSER regions')
hold off
% Use regionprops to measure MSER properties
mserStats = regionprops(mserConnComp, 'BoundingBox','Image');
toc
tic
% Compute the aspect ratio using bounding box data.
bbox = vertcat(mserStats.BoundingBox);
w = bbox(:,3);
h = bbox(:,4);
aspectRatio = w./h;
% Threshold the data to determine which regions to remove. These thresho
filterIdx = aspectRatio' > 10 ;
filterIdx = aspectRatio' < 1/10 ;
% Remove regions
mserStats(filterIdx) = [];
mserRegions(filterIdx) = [];
% Show remaining regions
figure
imshow(I)
hold on
plot(mserRegions, 'showPixelList', true,'showEllipses',false)
title('After Removing Non-Text Regions Based On Aspect Ratio')
hold off
toc
tic
% Get a binary image of the a region, and pad it to avoid boundary effect
% during the stroke width computation.
regionImage = mserStats(6).Image;
regionImage = padarray(regionImage, [1 1]);
```

```
% Compute the stroke width image.
distanceImage = bwdist(~regionImage);
skeletonImage = bwmorph(regionImage, 'thin', inf);
strokeWidthImage = distanceImage;
strokeWidthImage(~skeletonImage) = 0;
% Show the region image alongside the stroke width image.
figure
subplot(1,2,1)
imagesc(regionImage)
title('Region Image')
subplot(1,2,2)
imagesc(strokeWidthImage)
title('Stroke Width Image')
% Compute the stroke width variation metric
strokeWidthValues = distanceImage(skeletonImage);
yb=strokeWidthValues;
u=var(strokeWidthValues(:));
v =std(strokeWidthValues);
z=mean(strokeWidthValues);
x=v/z;
strokeWidthMetric = std(strokeWidthValues)/mean(strokeWidthValues);
b=strokeWidthMetric;

% y = var(strokeWidthMetric(:));
% Threshold the stroke width variation metric
strokeWidthThreshold = 0.5;
strokeWidthFilterIdx = strokeWidthMetric > strokeWidthThreshold;
% Process the remaining regions
for j = 1:numel(mserStats)

    regionImage = mserStats(j).Image;
    regionImage = padarray(regionImage, [1 1], 0);

    distanceImage = bwdist(~regionImage);
    skeletonImage = bwmorph(regionImage, 'thin', inf);

    strokeWidthValues = distanceImage(skeletonImage);

    strokeWidthMetric = std(strokeWidthValues)/mean(strokeWidthValues);

    strokeWidthFilterIdx(j) = strokeWidthMetric > strokeWidthThreshold;

end
```

```
% Remove regions based on the stroke width variation
mserRegions(strokeWidthFilterIdx) = [];
mserStats(strokeWidthFilterIdx) = [];

% Show remaining regions
figure
imshow(I)
hold on
plot(mserRegions, 'showPixelList', true,'showEllipses',false)
title('After Removing Non-Text Regions Based On Stroke Width Variation')
hold off
toc
tic
% Get bounding boxes for all the regions
bboxes = vertcat(mserStats.BoundingBox);

% Convert from the [x y width height] bounding box format to the [xmin yr
% xmax ymax] format for convenience.
xmin = bboxes(:,1);
ymin = bboxes(:,2);
xmax = xmin + bboxes(:,3) - 1;
ymax = ymin + bboxes(:,4) - 1;
% Expand the bounding boxes by a small amount.
expansionAmount = 0.02;
xmin = (1-expansionAmount) * xmin;
ymin = (1-expansionAmount) * ymin;
xmax = (1+expansionAmount) * xmax;
ymax = (1+expansionAmount) * ymax;
% Clip the bounding boxes to be within the image bounds
xmin = max(xmin, 1);
ymin = max(ymin, 1);
xmax = min(xmax, size(I,2));
ymax = min(ymax, size(I,1));
% Show the expanded bounding boxes
expandedBBoxes = [xmin ymin xmax-xmin+1 ymax-ymin+1];
IExpandedBBoxes = insertShape(colorImage,'Rectangle',expandedBBoxes,'Line
figure
imshow(IExpandedBBoxes)
title('Expanded Bounding Boxes Text')
% Compute the overlap ratio
overlapRatio = bboxOverlapRatio(expandedBBoxes, expandedBBoxes);
% Set the overlap ratio between a bounding box and itself to zero to
% simplify the graph representation.
```

```
n = size(overlapRatio,1);
overlapRatio(1:n+1:n^2) = 0;
% Create the graph
g = graph(overlapRatio);
% Find the connected text regions within the graph
componentIndices = conncomp(g);
% Merge the boxes based on the minimum and maximum dimensions.
xmin = accumarray(componentIndices', xmin, [], @min);
ymin = accumarray(componentIndices', ymin, [], @min);
xmax = accumarray(componentIndices', xmax, [], @max);
ymax = accumarray(componentIndices', ymax, [], @max);
% Compose the merged bounding boxes using the [x y width height] format.
textBBoxes = [xmin ymin xmax-xmin+1 ymax-ymin+1];
% Remove bounding boxes that only contain one text region
numRegionsInGroup = histcounts(componentIndices);
textBBoxes(numRegionsInGroup == 1, :) = [];
figure
% imshow(textBBoxes)
% Show the final text detection result.
ITextRegion = insertShape(colorImage, 'Rectangle', textBBoxes,'LineWidth
%  measurements = regionprops(colorImage, textBBoxes);
measurements = regionprops(textBBoxes);
disp(measurements)
imshow(ITextRegion)
title('Candidate  Text Regions')
cc=1;
toc
for K = 1  : size(textBBoxes,1)
  count=1;
figure
subimage = imcrop(colorImage,textBBoxes (K,:));
 imshow(subimage)
 path = 'C:/Users/weyra\Desktop\das\images\24/';
    imwrite(subimage, strcat(path, sprintf('%d.jpg',K)),'jpg');

end
```