FITEHALEW ASHAGRIE DEMILEW ANCIENT GEEZ SCRIPT RECOGNITION USING DEEP CONVOLUTIONAL NEURAL NETWORK NEU 2019

ANCIENT GEEZ SCRIPT RECOGNITION USING DEEP CONVOLUTIONAL NEURAL NETWORK

THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
FITEHALEW ASHAGRIE DEMILEW

In Partial Fulfillment of the Requirements for the Degree of Master of Sciences in Software Engineering

NICOSIA, 2019

ANCIENT GEEZ SCRIPT RECOGNITION USING DEEP CONVOLUTIONAL NEURAL NETWORK

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
FITEHALEW ASHAGRIE DEMILEW

In Partial Fulfillment of the Requirements for the Degree of Master of Sciences in Software Engineering

NICOSIA, 2019

Fitehalew Ashagrie DEMILEW: ANCIENT GEEZ SCRIPT RECOGNITION USING DEEP CONVOLUTIONAL NEURAL NETWORK

Approval of Director of Graduate School of Applied Sciences

Prof.Dr.Nadire CAVUS

We certify this thesis is satisfactory for the award of the degree of Master of Sciences in Software Engineering

Examine committee in charge:

Assoc. Prof. Dr. Kamil DİMİLİLER	Department of Automotive Engineering, NEU
Assoc. Prof. Dr. Yöney KIRSAL EVER	Department of Software Engineering, NEU
Assist. Prof. Dr. Boran ŞEKEROĞLU	Supervisor, Department of Information Systems Engineering, NEU

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original of this work.

Name, Last name: Fitehalew Ashagrie Demilew

Signature:

Date:

ACKNOWLEDGMENT

My deepest gratitude is to my advisor, Assist. Prof. Dr. Boran ŞEKEROĞLU, for his encouragement, guidance, support, and enthusiasm with his knowledge. I am indeed grateful for his continuous support of finalizing this project and for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing this paper without him, everything would not be possible.

Then, I would like to thank my mother Tirngo Assefaw, my father Ashagrie Demilew, my brother Baharu Ashagrie, and the rest of my family for their support, encouragement, and ideas. Without you, everything would have been impossible and difficult for me.

Dedicated to my mother Tirngo Assefaw...

ABSTRACT

In this research paper, we have presented the design and development of an optical character recognition system for ancient handwritten Geez documents. Geez, alphabet contains 26 base characters and more than 150 derived characters which are an extension of the base alphabets. These derived characters are formed by adding different kinds of strokes on the base characters. However, this paper focusses on the 26 base characters and 3 of the punctuation marks of Geez alphabets. The proposed character recognition system compromises all of the necessary steps that are required for developing an efficient recognition system. The designed system includes processes like preprocessing, segmentation, feature extraction, and classification. The preprocessing stage includes steps such as grayscale conversion, noise reduction, binarization, and skew correction. Many languages require word segmentation however, the Geez language doesn't require it so, the segmentation stage encompasses only line and character segmentation.

Among the different character classification techniques, this paper presents a deep convolutional neural network approach. A deep CNN is used for feature extraction and character classification purposes. Furthermore, we have prepared a dataset containing a total of 22,913 characters in which 70% (16,038) was used for training, 20% (4,583) used for testing, and 10% (2,292) was used for validation purpose. A total of 208 pages were collected from EOTC and other places in order to prepare the dataset. For the case of proving the proposed model is effective and efficient also, we have designed a deep neural network architecture with 3 different hidden layers. Both of the designed models were trained with the same training dataset and their results show that the deep CNN model is better in every case. The deep neural model with 2 hidden layers achieves an accuracy of 98.128 with a model loss of 0.095 however, the proposed deep CNN model obtained an accuracy of 99.389% with a model loss of 0.044. Thus, the deep CNN architecture results with a better recognition accuracy for ancient Geez document recognition.

Keywords: Ancient Document Recognition; Geez Document Recognition; Ethiopic Document Recognition; Deep Convolutional Neural Network

ÖZET

Bu araştırma makalesinde, eski el yazısı Geez belgeleri için bir optik karakter tanıma sisteminin tasarımını ve geliştirilmesini sunduk. Tanrım, alfabe 26 baz karakter ve baz alfabelerin bir uzantısı olan 150'den fazla türetilmiş karakter içeriyor. Bu türetilmiş karakterler, temel karakterlere farklı tür vuruşlar eklenerek oluşturulur. Bununla birlikte, bu makale 26 temel karaktere ve Geez alfabelerinin noktalama işaretlerinden 3'üne odaklanmaktadır. Önerilen karakter tanıma sistemi, verimli bir tanıma sistemi geliştirmek için gerekli olan tüm gerekli adımları yerine getirir. Tasarlanan sistem ön işleme, segmentasyon, özellik çıkarma ve sınıflandırma gibi işlemleri içerir. Ön işleme aşaması, gri tonlamalı dönüştürme, gürültü azaltma, ikilileştirme ve eğri düzeltme gibi adımları içerir. Birçok dil kelime bölümlendirmesini gerektirir, ancak Geez dili bunu gerektirmez, bölümleme aşaması sadece çizgi ve karakter bölümlendirmesini kapsar.

Farklı karakter sınıflandırma teknikleri arasında, bu makale derin bir evrişimsel sinir ağı yaklaşımı sunmaktadır. Özellik çıkarma ve karakter sınıflandırma amacıyla derin bir CNN kullanılır. Ayrıca, eğitim için% 70 (16,038), test için% 20 (4,583) ve validasyon amacıyla% 10 (2,292) kullanılmış toplam 22,913 karakter içeren bir veri seti hazırladık. Veri setini hazırlamak için EOTC ve diğer yerlerden toplam 208 sayfa toplanmıştır. Önerilen modelin etkili ve verimli olduğunu kanıtlamak için, 3 farklı katmanı olan derin bir sinir ağı mimarisi tasarladık. Tasarlanan modellerin her ikisi de aynı eğitim veri seti ile eğitildi ve sonuçları, derin CNN modelinin her durumda daha iyi olduğunu gösteriyor. 2 gizli katmanı olan derin sinir modeli, 0.095 model kaybıyla 98.128 kesinliğe ulaşır, ancak önerilen derin CNN modeli 0.044 model kaybıyla% 99.389 hassasiyet elde etti. Böylece, derin CNN mimarisi, eski Geez belge tanıma için daha iyi bir tanıma doğruluğu ile sonuçlanır.

Anahtar Kelimeler: Eski Belge Tanıma; Geez Belge Tanıma; Etiyopik Belge Tanıma; Derin Konvolüsyonlu Sinir Ağı

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

OCR:	Optical Character Recognition
ASCII:	American Standard Code for Information Interchange
EOTC:	Ethiopian Orthodox Tewahedo Church
NN:	Neural Network
ANN:	Artificial Neural Network
HMM:	Hidden Markov Model
DCNN:	Deep Convolutional Neural Network
CNN:	Convolutional Neural Network
ConvNet:	Convolutional Neural Network
2D:	Two-Dimensional
3D:	Three-Dimensional
API:	Application Program Interface
HWR:	Handwritten Word Recognition
HCR:	Handwritten Character recognition
NIST:	National Institute of Standards
GRUHD:	Greek Database of Unconstrained Handwriting
FCC:	Freeman Chain Code
KNN:	K-Nearest Neighbor Network
HT:	Hough Transform
SVM:	Support Vector Machine
MLP:	Multilayer Perceptron
LSTM:	Long Short-Term Memory
SIFT:	Scale Invariant Feature Transform
HMRF:	Hidden Markov Random Field
RGB:	Red Green Blue
DNN:	Deep Neural Network
MNIST:	Mixed National Institute of Standards and Technology
NLP:	Natural Language Processing
PWM:	Parzen Window Method
ReLU:	Rectified Linear Unit
SGD:	Stochastic Gradient Descent

OpenCV: Open Source Computer Vision

NumPy: Numerical Python

CHAPTER 1
INTRODUCTION

1.1. Background

Optical Character Recognition (OCR) is the process of extracting or detecting characters from an image and converting each extracted character into an American Standard Code for Information Interchange (ASCII), Unicode, or computer editable format. The input image can be an image of a printed or handwritten paper. Handwritten character recognition involves converting a large number of handwritten documents into a machine-editable document containing the extracted characters in the original order. Technically, the handwritten text recognition process includes different kinds of step by step stages and subprocess. The first procedure is image acquisition, which involves collecting and preparing images for further processes. Then pre-processing, which comprises operations like noise removal and clearing the background of the scanned image so that the relevant character pixels will be visible. Segmentation process involves extracting lines, words, and characters from the scanned image after the preprocessing stage is completed. Finally, the segmentation step is followed by feature extraction and classifying the characters using the selected classification approach.

Technically, the main procedures of handwritten text recognition are scanning documents or image acquisition, binarization which involves converting the images pixel into perfectly black or white pixels, segmentation, feature extraction, recognition, and/or possibly post-processing (Kim et al., 1999). Additionally, OCR systems are highly influenced by factors like the font or style of writing, scanning devices, and quality of the scanned paper. Moreover, the presence and absence of pixels from the scanned image and the quality of the scanner or camera used to take the pictures from the document can highly affect the recognition process. In order to increase the performance of OCR systems, various preprocessing techniques or strategies should be applied to the original images.

Generally, there are two types of handwriting text recognition systems which are offline and online text recognition. Online text recognition is applied to data that are captured at present or real-time. It’s achieved while a text is being written on touch-sensitive screens of smartphones or tablet computers. For online text recognition, information like pen-tip location points, pressure, and current information while writing are available which aren’t available in case of offline recognition. Thus, online text recognition is easy when it is compared to offline recognition

(Ahmad and Fink, 2016). In offline text recognition, a scanned image or image captured using a digital camera are used as an input to the software to be recognized (Shafii, 2014). Once the images are captured, some pre-processing activities are applied to them in order to recognize the characters accurately. Hence, offline recognition requires pre-processing activities on the images, it is considered as a difficult operation than online recognition.

Ethiopia is the only country in Africa to have its own indigenous alphabets and writing systems (Sahle, 2015). Which is Geez or Amharic alphabets, unlike many of the African countries. Most of the African countries use English and Arabic scripts or alphabets for their languages. The word Geez can also spell as Ge'ez, it's the liturgical language of the Ethiopian Orthodox Tewahedo Church (EOTCs). Ge'ez is a type of Semitic language and mostly used in Ethiopian Orthodox Tewahedo churches and Eritrean Orthodox Tewahedo churches. Geez belongs in the South Arabic dialects and Amharic which is one of the most spoken languages of Ethiopia (Britannica the editors of Encyclopedia, 2015). There are more than 80 languages and up to 200 dialects spoken in Ethiopian, some of those languages use Ge'ez as there writing script. Among the languages, Ge'ez, Amharic, Tigrinya are the most spoken languages and they are written and read from left-to-right, unlike the other Semitic languages which are written and read from right-to-left (Bender, 1997).

There are a lot of ancient manuscripts which are written in Ge'ez currently in Ethiopian especially in the EOTCs. However, we can't find a digital format of those manuscripts due to a lack of a system that can convert them. There is no doubt that the manuscripts contain an intense amount of ancient knowledge, civilizations, and political and religious attitude of those peoples. Similarly, we can find manuscripts and scriptures in Eritrean Orthodox Churches which are written in Ge'ez language. Ethiopian and Eretria have been exercising the same religious belief for many years. Geez is the primary language to be used in the Orthodox churches of both Ethiopia and Eretria for many years. The ancient Ethiopian manuscripts are basically unique and different than the modern handwritten documents. The main differences between the modern and ancient handwritten documents are styles of writing, characters size, background colors of the paper or parchment, the writing materials nature, and morphological structure of the characters. The detail discussion of the differences can be found in chapters 4. Character recognition is a common research field and application of pattern recognition and artificial neural network (ANN) respectively. Thus, we have discussed some of the fundamental concepts of ANN in the following sections.

1.2. Overview of Artificial Neural Network

An artificial neural network (ANN) is a network of neural networks which is modeled based on the structure and nature of the human brain. Humans' brain constitutes billions of neural cells connected to each other and they transfer information from one neuron cell to another. Similarly, ANN simulates human brains on computers for the purpose of creating an intelligent machine. It was really difficult to create a machine that can learn new things from the environment by itself. But, the invention of artificial neural networks gives the ability to learn new things for the machines, so that we can create an intelligent machine. Nowadays, ANNs are being applied in nearly every fields.

Technically, a neural network is consisting of three critical parts which are input layers, hidden layers, and output layers. Every input node is connected with hidden layers and each hidden layer are connected with the output nodes. The data will be entered through input nodes and will be processed in the hidden layer then we can get the result of the processed data from the output nodes. Any neural network model constitutes of nodes; which are used for creating the input, hidden and output layers, weights of each node and activation function as shown in Figure 1.4.

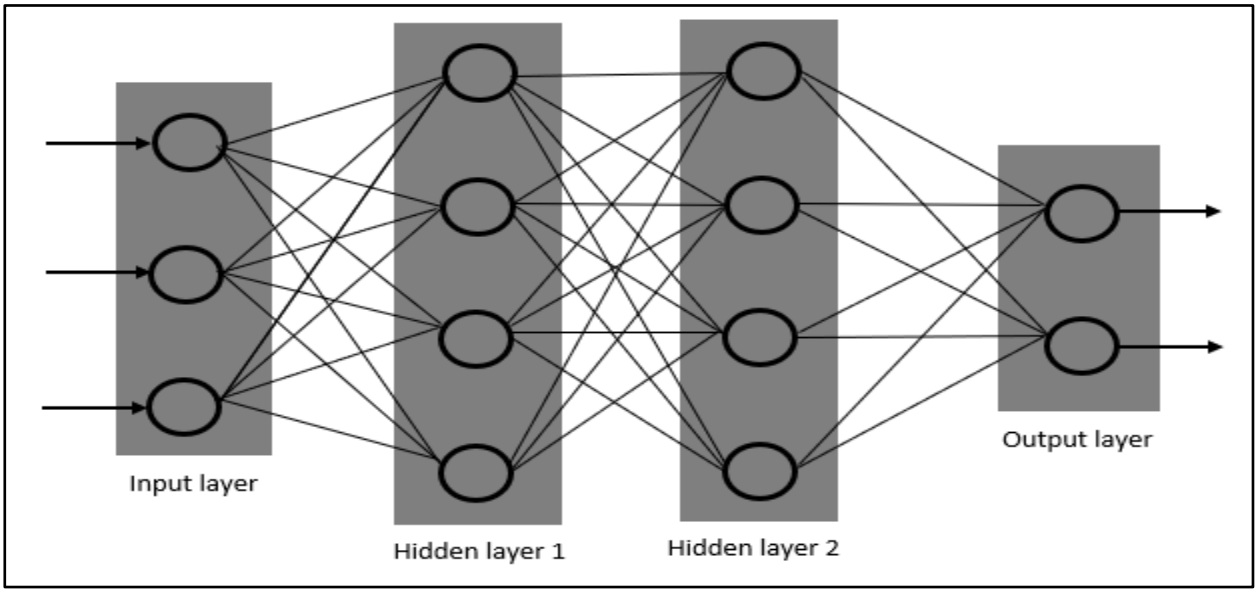


Figure 1.1: Fully connected artificial neural network model

The critical step before designing any neural networks architecture is the process of preparing dataset. The dataset is used for training the neural network so that it can predict the desired questions using the training dataset. Just like humans if we need to solve some specific problems first, we need to train our mind to solve similar problems then we will be able to answer other questions based on our previous experience of training. The are many different ways of how neural

networks learn or get trained such as supervised, unsupervised and reinforcement learning. Some of the most common training techniques of artificial neural networks and their applications are discussed in the following sections.

1.2.1. Supervised learning

Supervised learning is the process of teaching or training an artificial neural network by feeding it with a serious of questions with their respective answers. So, at any time a new question comes the neural network will predict the answer for the question by using its previous training data. We can see an example of a neural network which is designed to decide if a word is a positive or negative word. Thus, the out of the neural network will be either positive or negative. First, we need to train the neural network by feeding it a list of words with their respective answers which are either negative or positive. The words are called the training set and the answers for those questions are called true classes. Then, we can test the accuracy of the model with a new word. The NN will use the previous training data to predict if the entered word is either positive or negative. Some of the applications of supervised learning method are listed below:

- ✓ Face recognition,
- ✓ Fingerprint recognition,
- ✓ Character recognition,
- ✓ Speech recognition.

1.2.2. Unsupervised learning

Unsupervised learning is usually used when there is no dataset with a known answer. In this learning mechanism, the neural network will only be trained with datasets that don't contain any kinds of labels, classifications or answers. In unsupervised learning mechanism, we will use techniques called clustering and reducing dimensionality. Clustering focuses on clustering data into groups based on different characteristics of the data such as type, similarity, and nature. The latter techniques involve compressing the training data of the neural network without affecting and changing the structure or behavior of the data in order to minimize its memory usage. Some of the common applications of unsupervised learning method are listed below:

- ✓ Military operations such as antiaircraft and antimissiles,
- ✓ Customer segmentation on market operations,
- ✓ Fraud detection on banking systems,
- ✓ Gene clustering in medical fields.

1.2.3. Reinforcement learning

Reinforcement learning involves training and making a decision by the neural network through observations of the environment. If the observation is negative then it will readjust the weights for the next time in order to make an expected decision. The process flow of reinforcement learning is just like a conversation between the neural network and the environment. This technique involves training a neural network using rewarding and punishing. In which the main goal of this method is to decrease the punishment and increase the reward. Some of the most common applications of reinforcement training are listed below: For more detailed discussion and implantation of the applications of reinforcement learning see (Mao et al., 2016), (Arel et al., 2010), (Leviney et all, 2016), (Bu et al., 2009) and (Zhou et al., 2017).

- ✓ Resource management on a computer,
- ✓ Traffic light control,
- ✓ Robotics,
- ✓ Web system configuration and for optimizing chemical reactions.

1.3. Statement of the Problem

Nowadays, handwriting recognition has been one of the most interesting and challenging research areas in the field of artificial intelligence, pattern recognition, and image processing. It hugely participates and adds massive concepts and improvements to the development of an automation process. Furthermore, it can also increase the boundary between man and machine in various ways and applications. There are plenty of research works that have been focusing on new techniques, procedures, technologies, and methods. So that it would greatly improve the efficiency of character recognition systems (Pradeep et al., 2011). So far, handwritten character recognition is a very difficult and challenging area of pattern recognition and image processing. The challenge and difficulty in detecting handwritten documents come with different kinds of reasons such as non-uniformity of gaps or spaces between characters, words and lines and cursive-ness of the handwritten character. Due to those issues, the essential procedures of text recognition like line segmentation, word segmentation, and character segmentation will become incredibly difficult. Sometimes it’s difficult to recognize handwritten texts for humans even, some moments will happen in which we couldn’t understand our own handwritten documents.

There are several handwritten recognition research papers made so far for the most popular languages English, Chinese, French and Arabic and they seem to be much appreciable and successful (Assabie and Bigun, 2011). Also, researches on Amharic characters recognition have

been done by researchers starting from the late 19's century (Assabie, 2002) and (Bigun, 2008). Even though the research works are few in numbers, they have established a greater path for newer researchers. Obviously, in ancient times there are no papers and pens like the modern days. The ancient geez scriptures are written on Branna which is made of animals' skin and serves as a paper. That's the main reason that the manuscripts are still safe and haven't been damaged even if they are too old. They use special types of inks as a writing ingredient and it's prepared from plants' leave. These ancient writing materials are really incredible, however, at the same time, they have their own disadvantages for character recognition process because of the nature of the materials.

The first research on Amharic character recognition was made in 1997 by an Ethiopian researcher named Worku Alemu (Alemu, 1997). The researcher tried to conduct segmentation techniques on Amharic characters, and after him, many researchers are contributing their part to the filed. Following him, another researcher applied some preprocessing techniques for the purpose of using the algorithm in formatted Amharic texts. Also, many other researchers have tried to study related fields of Amharic Handwritten text recognition like for postal address recognition, for check recognition and Amharic word recognition. Other researchers in 2011 named Yaregal and Josef published a paper about practices and methods for Amharic word recognition in non-connected and non-cursive Amharic handwritten text using Hidden Markov Models (HMMs) (Assabie and Bigun, 2011).

However, geez handwritten text recognition is still an area which requires many researchers' contribution. More researches are required to improve the feature extraction processes. As well as researches oriented on the feature extraction processes additional researches are appreciable for the improvement of segmentation and classifying processes. Geez, language has a large character richness with a greater morphological similarity among the characters causing a difficult task in the recognition processes. These behaviors of geez characters inspire many researchers to perform their researches on the field and geez character recognition will remain as an active research area for researchers.

It's well-known that pre-processing, segmentation, and classification are the most important task of both handwritten and printed character recognition systems. An advanced level of line segmentation, word segmentation, and character segmentation will result in a good character recognition system. These and other additional problems in character recognition processes attract the attention of new researchers to perform new researches for better improvements to the segmentation process. As well as old researchers trying to improve their previous works. Since

Ethiopia is an ancient country it's believed to have more than a thousand years of history and civilizations. And, in those years of civilizations, there are a lot of books and monographic scriptures. Although, as opposed to the benefits' we can get from those documents we cannot find a well standardized digital form of the documents. Additionally, we can't find a character recognition system that we will be used to convert the manuscripts into machine-editable formats. These problems of getting the ancient monographic scriptures in digital and editable format require and also gets the attention of new researchers. Thus, researchers perform their work and improve older researches to provide well suitable character recognition software or system to convert the handwritten scriptures into digital format.

1.4. The Significance of the Study

Handwritten information's which, are written in different languages can be found abundantly in many places in the world especially, ancient documents, scriptures, and manuscript which contains much valuable information are written by hand. However, a significant amount of the documents containing the valuable information cannot be found on the internet easily due to the fact of being hard and impossible to find a digital or editable format of the information. The process of converting the information into digital forms for every language highly depends on the characteristics of the language and on the morphological structures of the characters (Plamondon and Srihari, 2000).

For countries like Ethiopian which uses a unique kind of languages and that language is exercised inside the country only, the process of converting the documents to digital format will become a very difficult, expensive and time-consuming task. This is due to the lack of extensive research works in the area so, it requires more research work on Ge'ez OCR systems. Since Ge'ez has been a working language for Ethiopian Orthodox Churches for more than 15 centuries, a large number of handwritten documents, scriptures, and manuscripts containing highly valuable information are located in churches. Retrieving the contents of these valuable documents and making them available on the internet for any other users requires the process of converting the documents into digital format. The documents must be digitized which means the Ge'ez characters must be converted into ASCII or Unicode. The Ge'ez characters are called Fidel or "ፊደል" called in a native language. Few research papers have been done on Amharic and Ge'ez characters recognition since 1997. Additionally, there are other researchers who have made their study on lightweight systems such as car plate recognition, postal address recognition and many others (Assabie, 2011, 2002) and (Plamondon and Srihari, 2000) and (Meshesha and Jawahar, 2007). However, Ge'ez or

character recognition requires far more researchers to work in order to achieve well-standardized and advanced level recognition system. Also, the continuity and acceptability of human communication using handwritten texts by itself would get the attention of new researchers to work their study on handwritten recognition.

1.5. Objectives of the Study

1.5.1. General objective

The general objective of this thesis is to design and implement Geez character recognition system for ancient Ethiopian documents using deep convolutional neural networks (Deep CNN).

1.5.2. Specific objectives

- To study the general steps which are required to design and implement the OCR system.
- To explore the main areas that make the character recognition process more efficient.
- To prepare a training and testing dataset for Geez character recognition systems.
- To design a deep convolutional neural network and its weight parameters.
- Carry on a literature review, on handwritten character recognition, segmentation, feature extraction and other areas for Ge'ez character recognition systems.
- Review related works on handwritten character recognition systems which are done on other languages.
- Test the developed prototype of the recognition system using the prepared dataset.
- To study the linguistic structure and nature of the Geez language.
- Distinguish among the different training techniques then, implement a proper and best training algorithm for training the neural network.
- Develop a prototype of the Ge'ez character recognition system for the ancient Ethiopian documents system using Keras APIs in a python programming language.
- Evaluate the performance of the designed prototype of the recognition system based on the prepared testing dataset and provide a conclusion.

1.6. Methodology

The methodologies described below contains all the relevant methodologies that are used and applied to design and develop the proposed system. Which includes a methodology used to collect ancient scriptures for training and testing purpose of the network model. Additionally, the methodologies which are required in order to design and implement the prototype of the optical character recognition system. The most relevant methodologies are described as follows:

1.6.1. Literature review

Many kinds of documents, articles, journals, books and other research papers which are related to the research area are reviewed. As we have tried to explain on the previous sections it's really difficult to get efficient and as many as the required number of researches on Ge'ez language recognition especially, on the ancient Ethiopian document recognition research topic. Thus, we have reviewed many kinds of papers which are related to image processing, handwritten character recognin and optical character recognition in general. Particularly we have reviewed many previous types of research works which are done in many different languages. However, most of the documents we have reviewed are research papers which are done on other common languages especially English, Bangla, and Arabic. We have used the information gathered from those papers to design a better and as possible as efficient Geez document recognition system.

1.6.2. Data acquisition techniques

Most of the images which are required for training and testing the neural network are collected from EOTCs using a digital camera and scanner. The scanned images are then preprocessed in order to feed them to the network. After, the collected images pass through the preprocessing stage a dataset will be prepared. Unlike online character recognition systems, we can't trace and extract the pixels of the characters from the drawer's movement (Plamondon and Srihari, 2000). Therefore, we need to acquire images and apply some preprocessing techniques to the images to extract all the relevant pixels of the characters after removing the unwanted pixels.

1.6.3. Preprocessing methods

Once the images of the ancient scriptures or holographs are collected from the churches, museums, and libraries we made them ready for further processing. Every image passes through different phases like gray scaling, binarizing, noise reduction, skewing correction, and finally classification of the images. After the procedures of preprocessing are applied to the images a dataset will be created. Hence, this database will be used for training and testing of the network later on. We can say almost all the volumes of scriptures that are collected are too aged and damaged. These problems require a more advanced and effective mechanism of preprocessing in order to extract the relevant pixels from the images. Additionally, the preprocessing stage defines the quality and efficiency of the recognition process thus, it needs to be done carefully.

1.6.4. System modeling and implementation

The prototype of the Ge'ez character recognition system is designed using Deep Convolutional Neural Network (DCNN), which is provided by Keras API. By using this API, we have designed

the neural network that is capable of recognizing handwritten Ge'ez characters. Keras is an API designed for image processing purpose and its available for different programming languages such as C++ and python. We have implemented the prototype of the character recognition system using the most powerful programming language for image processing and machine learning which is python. The detail explanation of the methodologies used for preprocessing, segmentation, and character classification are discussed in detail in the following chapters.

1.7. Scope and Limitations

Many ancient Ethiopian documents or manuscripts contain characters, numbers and different kinds of punctuations. The problem with the scriptures being too aged plus being not well-maintained and the challenge of collecting data for training and testing the system makes the character recognition system difficult. Therefore, in this thesis, we have focused only on the 26 base characters of geez alphabets and 3 of the punctuation marks. We can say that any researcher can find a dataset for a language other than Amharic or Ge'ez easily in order to train and test a character recognition system. So, preparing a dataset for the Ge'ez character recognition system becomes a crucial and undeniable process to be accomplished while conducting the development of the OCR system. Amharic alphabet contains 33 base characters, each character having 7 forms so, the total number of characters in Amharic languages is 231 characters where 26 of the base characters are taken from Geez alphabets. Thus, in Ge'ez characters there are 26 base characters each having 7 different forms which means the total number of characters in Ge'ez language is 182 characters. In this research work, we have covered only the 26 base characters of Ge'ez languages and 3 of the punctuation marks of the language. So, indirectly we are covering more than 70% of Amharic base characters.

Furthermore, Geez language uses different kinds of numbering systems and it represents them using unique kinds of symbols. Geez, number recognition is not covered in this paper. The nature of Ge'ez numbers is much more complicated and difficult than the Latin numbers. Some of the example of Ge'ez numbers and their English numeral equivalent are 1 "፩", 2 "፪", 10 "፲", 12 "፲፪", 30 "፴", 100 "፻", 1000 "፲፻". The nature of Geez numerals are complicated, most of Ge'ez character is written as a connected word rather than a single character. For more information about Geez numerals, visit these sites (Amharic numbers, 2019) or (Numbers in Geez, 2019).

1.8. Document Organization

The document is organized with seven chapters starting from the introduction to the conclusions and future work section. In the second chapter, we have reviewed some relevant and related articles

with the proposed system. The major design and development methodologies used while implementing the prototype are discussed in chapter three. A detailed explanation and discussion of nature, styles of ancient Geez documents and the recognition processes of handwritten characters are discussed in the fourth chapter. The development of the proposed ancient Geez document recognition system is explained in chapter five. Additionally, in chapter six the results obtained from the proposed system and its comparison with other systems are presented. Finally, in the last chapter, the conclusion of the overall document is presented with future works.

CHAPTER 2
LITERATURE REVIEW

Handwritten character recognition (HCR) has been an active and interesting area for researchers. Many research papers, published and unpublished journals, and other studies have been done by researchers in order to improve the efficiency and accuracy of the recognition process. Handwritten character recognition involves many relevant phases or stages like data acquisition, preprocessing, classification, and postprocessing. And each phase has its own objectives, their efficiency defines the accuracy of the overall recognition process. As we have tried to explain it before those processes need to be improved in order to come up with a better and efficient character recognition system. Many researchers have tried to improve each of those phases at different times, and still many papers are being published presenting new and efficient ideas about preprocessing, postprocessing and other relevant techniques. However, most of the researches which are done on the recognition process are language dependent and they lack being generalized. Especially it is difficult to find enough researches on Ge'ez or Amharic character recognition for both handwritten or printed document recognition process. We can say that it's almost impossible to find researchers done on any processes of character recognition for ancient Ethiopian scripts or manuscripts.

The overall process of handwritten character recognition is much difficult than printed character recognition which is applied on scanned printed documents. There is a huge variation in handwritten styles among people. Many people have their own writing style even its difficult for a person to use the same handwriting style every time he/she writes. Also, poor and cursive handwritings are really difficult to recognize, this makes the segmentation and classification process difficult. If the paper is written in cursive handwritten style then the process will not be handwritten character recognition (HCR) most likely it will be handwritten word recognition (HWR). This and many other reasons make the handwritten character recognition process especially the segmentation phase much harder than printed documents recognition. Phases like line segmentation, word segmentation, and character segmentation are the most relevant and advanced phases of any character recognition system, especially for handwritten character recognition. As we have discussed it HWR is difficult than printed document recognition. Moreover, ancient document character recognition is much difficult and advanced than even the handwritten character recognition process.

Basically, ancient documents are very noisy so, they require an advanced noise removal technique. Since the noise removal method affects the overall process of the recognition system, it needs to be done carefully. The background of most of the ancient documents is not white, we can even say they don't have a common background color. As a result, the noise removal process will be very difficult and if this phase cannot be achieved in a quality manner then the other phases will be much more difficult than expected. Most of the ancient documents are aged beyond our expectation. As a result, the letters, words and also the statements are not fully clear even for human eyes. There are many different ancient languages and documents which are written using their own languages. Nowadays, those documents are being converted into machine-editable texts by many researchers. Experts are studying them to discover the valuable and immense ancient knowledge of those people. It's believed that the ancient Ge'ez documents contain unexplored contents of ancient Ethiopian civilization, cultural and religious practices.

2.1. Handwritten Recognition

In a research (Pal et al., 2007) a handwritten Bangla character recognition was proposed by the researchers. The recognition mechanism used by the researchers was based on the information extracted from the bidirectional arc tangents through the application of a feature extraction method on the grayscale images. They have applied a "two by two" mean filtering technique and they applied this mean filtering technique for four iterations. After the mean filtering is done, they conduct a non-linear normalization of size on the images. Gradient image was relevant at this stage so, in order to obtain the gradient image, a Roberts filter was applied to the images. Finally, the quadratic classifier was used by the researchers for the purpose of classifying the characters. After all those processes and techniques were applied to the images the result was not that much enormous and outstanding even if it is great progress. They have achieved 85.9 percent of accuracy on Bangla dataset of containing 20,543 test samples by applying five-folds cross-validation method. The research conducted was not on a single character recognition but on compound characters making the recognition process much difficult. Because of the techniques they used which is a statistical classifier the accuracy of the recognition system was good even if very poor and noisy images were tested. The result of the noise images shows that they don't have that much difference with the non-noisy images according to their research paper. The proposed system uses a rejection mechanism which is used to reject the inputted image when it is difficult to be processed and recognized. Though the degree of error increases when the degree of rejection is low and while the degree of rejection increases the degree of error decreases. They have used a rejection criterion based on a discriminant function which will help them reject the character when the possibility of

recognition is too low. However, in our opinion, the system shows a better accuracy for the noisy images because of the criteria used to reject the characters. The proposed system in (Pal et al., 2007) rejects much of the compound characters when it can't recognize it well. Thus, character rejection by the system doesn't give us any promise that the character could be recognized if some kind of change is made on the system. However, it means the proposed system couldn't predict those characters at the time.

Furthermore, in another research (Alom et al., 2018) the researchers applied a Deep Connected Convolutional Neural Network (DCNN) on handwritten recognition of Bangla alphabets, numerals, and special characters. They have focused on implementing and testing different kinds of common DCNN models which are Fractal Net, All convolutional neural network, Dense Net, VGG-16 NET, and Network in Network. And they have concluded that the Deep Network model of the DCNN is much better than the others by achieving a better degree of accuracy on the process of Bangla handwritten characters recognition. They have used a Linux environment which is situated using Keras library which runs on the top of Theano. The deep network model of DCNN makes it possible for them to achieve 99.13 percent accuracy on handwritten Bangla dataset called CMATERdb. The dataset contains more than 6000 numerals each numeral has more than 600 scaled images, 15000 of isolated alphabets and 2231 special characters for both testing and training image samples. Each character in the dataset is a grayscaled image and their dimension is scaled down to 32 * 32 pixels. Their degree of accuracy in the recognition process shows us that the convolutional neural networks are better than the other techniques out there, especially the DCNN network is the best model among the other neural network models for Bangla handwritten recognition process. Procurement of the 99.13% accuracy on handwritten characters recognition is really remarkable and their work obtains a better accuracy than the research made in (Pal et al., 2007).

In a research work (Kavallieratou et al., 2003) the researchers explained about handwritten character recognition of English and Greek isolated alphabets using horizontal and vertical histograms in addition to the newly introduced radial histogram. They have used the proposed technique on English dataset called NIST and in Greek dataset called GRUHD in which the datasets contain an isolated character of each language. The recognition system was trained with 2000 alphabets and 128 classification classes for each symbol. The system was tested by 500 sample images for each symbol in which the training and testing process for each languages dataset were done separately. Also, they have applied lexicon to improve the recognition system's

accuracy and their result was encouraging but not that much efficient than the research made on (Alom et al., 2018) Bangla characters. For the English dataset, they have achieved an accuracy of 98.8%, 93.85, 91.4, and 82.7 for digits, uppercase characters, lowercase characters, and mixed characters respectively. In the same case for the Greek dataset, they have achieved an accuracy of 94.8%, 86.03%, 81%, and 72.8% for the digits, uppercase characters, lowercase characters, and mixed characters respectively. Also, in another research (Nasien et al., 2010) they conducted handwritten English character recognition on the same dataset as (Kavallieratou et al., 2003) NIST, using a Freeman Chain Code (FCC) technique. As a result, they have achieved 86.007%, 88.4671%, and 73.4464 for English lowercase, uppercase, and mixed samples respectively. This shows that the research made in (Kavallieratou et al., 2003) results with better accuracy than the research made in (Nasien et al., 2010) for the handwritten recognition of lowercase, uppercase and mixed characters of the NIST dataset.

In a paper (Pradeep, 2012) the researchers conducted English handwritten recognition on English dataset using different models of neural network techniques. The neural network structures or methods used in this research are K-Nearest Neighbor Network (KNN), Template Matching, Feed Forward Back Propagation Neural Network and Radial Basis Function Network for the implementation and development of the character classifier. Each neural network models used in (Pradeep, 2012) research paper is trained with the same English dataset containing 5200 characters in which each the 26 characters has 200 different images. And each character image has a dimension of 30* 20 pixels which will be an input for the designed neural network for each model. After training and testing each of the proposed neural network models with a similar dataset, the researchers obtained an accuracy of more than 90% for only 2 characters using the template matching technique. The accuracy of Feed forwarded neural network was more than 90% for 23 characters out of the 26 English characters. For the KNN algorithm or technique, the degree of accuracy was greater than 90% for only 8 characters. The last technique was radial basis function neural network model and the accuracy was more than 90% for 14 characters. And we can conclude that the feed-forwarded NN model was the best among the other according to the research (Pradeep, 2012) and the worst was template matching.

2.2. Ancient Document Recognition

Indeed, a vast number of ancient documents can be found in online digital libraries. However, most of the documents found in the digital libraries are scanned images of those ancient documents but not in machine editable format. Ancient document recognition involves converting of those

scanned images of the ancient documents to machine editable format, ASCII or Unicode. Ancient script or document recognition is far harder than the printed character recognition and handwritten character recognition processes because of many reasons. The main problems of ancient documents are the bad quality of the documents because of their age, the non-standard alphabets, the ink used to write the documents (Laskov, 2006). Additionally, the paper used in ancient time was not the same kinds of documents we use in modern times (Laskov, 2006). So far, this area remains as an active and attractive research area for researchers, many researchers are contributing their part on the filed even in the early days. In research work (Juan et al., 2010) the researchers conduct ancient document recognition. Correspondingly their paper describes three different kinds of systems in which the first two are concerned with the recognition. And the third system was intended on the alignment or confirmation of the corresponding handwritten characters with the transcription or text recognition of the documents. Also, the researchers propose three recognition processes which are preprocessing, line segmentation, and feature extraction, and Hidden Markov Model (HMM) for training and classification. The preprocessing stage of the proposed system involves skew detection and correction, background and noise removal. And in the line segmentation process each line of the scanned document is segmented and extracted out from the image. Then after line segmentation is done liner normalization and slant correction techniques are applied to the preprocessed images. After all of the first parts are completed an HMM is used as it's intended to do. They have tested their proposed system with Spanish ancient documents and the estimated accuracy of the post editing was 71 percent. Even though their result is not that much efficient compared to the previous papers but, it’s progress. Also, since the proposed system is dealing with ancient and degraded documents it’s expected to see such kind of results.

In research work (Laskov, 2006) the researchers conducted ancient document recognition on Christian orthodox church neume symbols. Neumes are symbolic structures that are used to represent musical notes by the orthodox church and still, they are being used in the churches. As the researchers described in their paper, they have first conducted noise removal and binarization techniques on the scanned images of the ancient documents containing the neumes. Then the segmentation stage is done on the images which are containing the neume symbols and other especially symbols. The segmentation techniques used involves processes such as segmentation of each line from the preprocessed images and extracting each symbol from the extracted lines. Then the recognition or classification process was conducted by the researchers. A successful line segmentation was achieved based on the Hough Transform (HT) techniques and after the line

segmentation is completed each symbol were extracted from the lines based on vertical projection techniques (Dimov, 2001). They have concluded that the Hough Transform (HT) method is a better approach for segmentation, especially neume symbol recognition. However, they haven't specified anything about their testing of the system and the accuracy of recognition achieved by their system on their paper so, we can't compare their paper with other similar works done.

In a research paper (Malanker and Patel, 2014) the researchers conducted an ancient Devanagari document recognition using different kinds of character classification approaches. The approaches used by the researchers are ANN, Fuzzy model and Support Vector Machine (SVM). The researchers followed the common recognition steps starting from preprocessing to the classification stage. They first used a median filter for smoothing the image and then binarizing the image using the Otsu global thresholding approach. Also, the binarized image is smoothed again using the median filtering method. This median based smoothing technique helps the researchers for reducing noise from the scanned images. They have used many different kinds of feature extraction methods which are statistical, gradient, box approach, and zone approach. A neural network-based approach was used for the classification purpose, the researchers designed three different kinds of neural network models. The output of the three different neural network classifiers is combined using a connection mechanism of a fuzzy model-based scheme by representing them as an exponential function. When the exponential function is fitted the fuzzy does the recognition, the fuzzy sets are obtained from the normalized distance which is obtained using the box approaches. They have achieved an accuracy of 89.68% using a neural network classifier and 95% accuracy using a fuzzy model classifier for the numerals and 94% using SVM and 89.58 using multilayer perceptron (MLP) for the alphabets. Based on their results the neural network and support vector machine approaches aren't suitable for ancient Devanagari document recognition system. And the fuzzy model classifier approach seems much more suitable for the ancient Devanagari document recognition according to their results.

Unlike the other recognition techniques in research (Yousefi et al., 2015) the researchers propose an ancient Fraktur document recognition system that doesn't include any kind of binarization process. The researchers jump the binarization stage and go directly to the training of the neural network using a random grayscale text line. As their research describes they had prepared a huge amount of dataset from ancient documents. Then, they tested the dataset on both binarizing-free system and a system which includes a binarization process to differentiate the real gap between them. The neural network model they used for the classifier was 1-dimensional Long Short-Term

Memory (LSTM). The proposed LSTM is an example of recurrent neural networks and it doesn't have any segmentation processes. The LSTM model accepts directly the raw data as an input to the network and finds the relevant pixel area from it. The researchers prepared a dataset containing ancient documents of 55000 random lines extracted from 1762 pages for the training set and 3000 random lines extracted from 100 pages for the test set. Finally, the LSTM NN achieved an error rate of 0.38% for the grayscale line image and a 0.5% error rate for the binarized images of text lines. Their result shows us that the binarization free method acquires a 24% improvement in the recognition process.

Furthermore, in research (Diem and Sablatnig, 2010)) paper made the researchers conducted 11 century ancient Slavonic script recognition, they propose a binarization free recognition system just like the research made in (Yousefi et al., 2015). The system they proposed contains two distinct processes which are character classifier and localizer. Firstly, the system extracts the features using Scale Invariant Feature Transform (SIFT) and the features are classified using a Support Vector Machine (SVM). Then finally the characters are recognized using a weighted voting method. In which the character with the maximum voting count wins the selection process. Eventually, the researchers trained the SVM using 10 samples each for all the 107-character classes. When the voting technique is applied to the recognition process the classifier recognizes the characters with 98.13% accuracy. Which shows that only 2 of the 107 charters are wrongly classified and their result supports that the non-binarization method used in the research (Yousefi et al., 2015) is good for ancient document recognition.

The results of the research (Malanker and Patel, 2014) support the research done in (Yousefi et al., 2015) which explains the binarization free approach of ancient document recognition. For character localization techniques the researchers used artificial clustering technique. They have explained that the clustering technique degrades the accuracy of the system. The researchers tried to show that by comparing a system designed with a clustering localization technique and nanoclustering techniques. Furthermore, the results they achieved can distinguish the difference between the two approaches very well. The system with a clustering technique achieved an accuracy of 83.2% and the system without the artificial clustering the classifications accuracy was 83.7%. However, the obtained accuracy might not have that much difference although we don't have to forget that the slightest improvement is great progress.

2.3. Geez or Amharic Handwritten Recognition

It is believed that Amharic alphabets are derived from Ge'ez alphabets, but some experts don't agree about Geez being the ancestor of the Amharic language. Although, Ge'ez alphabets contain 28 base characters with each base character having their own different 7 kinds of shapes. However, Amharic alphabets contain all of the Ge'ez alphabets and they have an additional of around 6 or more base characters which are not a part of Ge'ez alphabets. So, any research or paperwork made on Amharic character recognition is also suitable for Ge'ez characters, since Amharic alphabets in composes all of the Ge'ez characters. That's why we start with the title "Ge'ez or Amharic handwritten recognition". Therefore, in this section, we have reviewed research works of both Amharic and Ge'ez document recognition. In research (Assabie and Bigun, 2011) two researchers conducted handwritten Amharic word recognition in unconstrained characters using Hidden Markov Models (HMMs) based on two different approaches. The first approach is a feature level, which means a model is constructed that can recognize words based on their features. However, the features of the characters are obtained from concatenated characters that are going to form a word. In the second approach is Hidden Markov Models (HMMs) level, which means characters derived from the HMMs technique are concatenated to form a word model recognition system.

Since the proposed system only involves word recognition it doesn't involve segmentation of characters, however, it involves line and word segmentation process (Assabie and Bigun, 2011). As the researchers described at the time while they were doing their research there was no dataset that will be used for the Amharic word recognition process. So, they have prepared a dataset for training and testing their proposed system. They used 5*5 matrices and 0.83 standard deviations of Gaussian filtering technique for noise reduction of noisy documents. But for characters having a smaller size they used gaussian filtering window of 3*3 matrices and 0.5 for deviation parameter unlike for the normal character sizes. For the preparation of the dataset 307 handwritten pages were collected from a total of 152 different writers and the pages were scanned with a resolution scale of 300dpi. After the pages are scanned a total number of 10,932 words were extracted from the pages for the case of preparing the training dataset for the word recognition model. In addition to that, the words in the dataset were divided into two equal parts of different types which are separated as poor and good quality images of words based on their noisiness and coloring scheme. In conclusion, the researchers obtained an accuracy of 76% for the good quality images using the first approach which uses the feature level concatenation method. And 53% for poor quality images with a total training dataset of 10,932 words. Moreover, for the HMMs level model they obtained

an accuracy of 66% for good quality images and 42% for the poor-quality images. And their conclusion shows that the feature-level concatenation model is better than HMMs level word concatenation models for handwritten Amharic word recognition. However, any of the three proposed systems doesn't seem an efficient recognition system for the handwritten Amharic recognition.

In research (Cowell and Hussain, 2003) the researchers proposed two kinds of statistical approaches for Amharic character recognition. In the first approach, each character is compared with a matching template to find its matching template. However, the second approach proposes a characteristic signature that is derived from the characters. Accordingly, these signatures will be used for comparing signatures with the set of signature templates. The signature of the system has been rescaled to fifty times smaller before conducting the recognition. But as a drawback, the rescaled images also, participate in reducing the accuracy of the recognition system. In which, while doing the process of rescaling the images, some relevant pixel information is lost. Also rescaling the image may result in creating another character or structure by compressing and condensing the pixels together. The first approach specifies that the degree of similarity will be generated by comparing the characters with the template characters. Then, the highest comparing degree of character template will be considered as a recognized character.

However, in the second approach of the research (Cowell and Hussain, 2003), a signature for each character is produced through an iteration technique. In which the iteration is achieved by looping through the overall pixels of the character images. The second approach compares the number of black pixels which are counted from 100 rows and 100 columns then it will be compared with the number of black pixels on the templates of signatures. In this research, the researchers used confusion matrices to show how many characters on what percentage are miss-classified for both approaches. For the first approach, 4 pairs have a maximum of degree confusion which is 97% and 37 pairs have a degree of confusion 90% or more. And for the second approach 377 resulted with more than 90% of confusion on classifying characters. Also, 2 pairs of characters result with a degree of 99% confusion, and 23 pairs result with a degree of 98% confusion while classifying characters. A character having a confusion degree of more than 90% on good quality images will surely have confusion on recognizing character with poor quality. As their result shows that the signature template-based character recognition for Amharic characters is worse than that of character template classification of Amharic characters.

Furthermore, in research (Alemu and Fuchs, 2003) the researchers proposed Amharic handwritten recognition system for legal bank check using Hidden Markov Random Field (HMRF) approach. As the researchers explained the HMRF approach is chosen by them because they want to recognize the characters based on the relationship they have among the surrounding characters. The proposed system involves three main processes which are feature extraction, character classification, and deriving contextual information based on the syntactical structure of the bank checks. Initially, the original images pass through the preprocessing stage which includes binarizing, segmentation, and rescaling the images into a common resolution. A new feature extraction technique is proposed by the system. The proposed feature extraction method computes feature of a character pixel based on the surrounding pixels rather than the pixels of the character. The proposed feature extraction algorithm computes the surrounding pixels for a given pixel based on the pixels directions which are up, down, left, and right. The algorithm counts the consecutive pixels points for each direction. Then, it compares the pixel count with the predefined threshold value and derives the features using a four-bit binary number.

In addition to that, the researchers in (Alemu and Fuchs, 2003) also have prepared a dataset which will be used for training and testing the proposed recognition system. The dataset is prepared from a total of 7240-character images, which are collected from different individuals. For the purpose of generating contextual information, the researchers used two methods which are word and character tree approaches. These two approaches are used to predict the specific words or characters by comparing them, with the frequently used characters or words in bank checks. Finally, the researchers collected 20 check paper containing a total of 713 individual characters to be used by the proposed system. The researchers designed two independent systems for the purpose of checking if the contextual information technique has a better effect on the recognition process. The first system was designed without the application of the contextual information technique and it obtains a recognition accuracy of 89.06%. And when the contextual information technique is applied the recognition system obtained an accuracy of 99.44%. The accuracy which is obtained from the first approach seems good. However, the accuracy obtained by the second is approach is better and really encouraging for a further research study on HMRF approach of the character recognition process. Eventually, the HMRF approach has the highest degree of accuracy so far when it is compared with the total reviewed researches done on this paper. Although, the research in (Alom et al., 2018), which is based on a deep CNN has the second highest degree of accuracy which is 99.13 %.

Also, in research (Birhanu and Sethuraman, 2015) the researchers conducted Amharic character recognition for real-time document recognition and they proposed an artificial neural network approach (ANN). The proposed system follows the three most common parts which are preprocessing, segmentation and recognition. Even though they have followed a standard and advanced character recognition process their result is not that much satisfying. After the proposed neural network is trained with the prepared dataset, the classification accuracy for the training image was an average of 96.87%. However, the accuracy of the system for additional testing images was 11.40% which is not satisfying and expected. Furthermore, in the paper made by Meshesha and Jawahar (2007), proposes an Amharic recognition system using linear discriminant analysis and support vector machines (SVMs) on real-time documents. The OCR system they proposed has been tested on 11 different kinds of fonts rather than handwritten documents and they have obtained an average accuracy of 96.99%.

2.4. Ancient Ethiopic Script Recognition

Even though there are tons of ancient Ge'ez and Amharic scripts, finding enough and as many as research papers which are done on those ancient documents is not possible. So, far we can only find some researches which are done on ancient script recognition of either Geez or Amharic characters. In a research made (Siranesh and Menore, 2016) the researcher conducted ancient Ethiopic manuscript recognition based on a deep network. After the neural network is designed it's trained based on unsupervised learning algorithm of a greedy layer. The proposed system includes all of the necessary processes that are useful for the recognition system to be as efficient as possible. The system first converts the images into grayscale images. Technically, RGB images are stored as three-dimensional matrices, but grayscaled images are stored as two-dimensional matrices making the recognition process much easier. As a result of the conversion of the RBG images into grayscale images, it saves memory usage and processing time required to perform any kinds of operations on the images. Then the grayscale images are binarized using a hybrid binarizing technique which is derived from the two most common methods which are global and local thresholding. Then, the binarization is followed by skew correction processes. In addition, the segmentation process is divided into two sections which are the line and character segmentation. For the process of the line and character segmentation, the researchers used horizontal and vertical projections respectively.

Furthermore, the researchers prepared a dataset for training and testing the proposed system (Siranesh and Menore, 2016). The dataset prepared contains 2400 images representing the 24 base

characters of Geez alphabet and each base alphabet has a repetition of 100 images. The characters were collected from a total of 200 pages. The dataset contains a total of 7065 characters which are normalized into the scale of 30*30 pixels. Among the total of the 7065 characters, 70% of the dataset was used for training and the rest were used for testing purpose. The researchers designed three kinds of DNNs in which the first model contains 2 hidden layers. However, the second and third models are designed using three and four hidden layers respectively. And each model was trained and tested separately with the same dataset. For the first model having two hidden layers the classification error with 300 neural nodes and 150 epochs is 0.063889. And for the second model which has three hidden layer and 300 nodes with similar epoch value results in a classification error of 0.0625. Finally, for the last model having four hidden layers having 300 units and with 150 epochs the classification error 0.077778 which was not as expected. Therefore, after analyzing the result obtained in (Siranesh and Menore, 2016) research we can conclude that a deep neural network approach with three hidden layer shows a better performance for ancient Ge'ez recognition, according to the proposed research in (Siranesh and Menore, 2016).

Furthermore, in another research, the researchers conducted degraded Ethiopic handwritten recognition (Bigun, 2008). The proposed system specifies a recognition system using writer-independent approaches. The proposed recognition system is based on a method which uses a primitive strokes technique. In which the characters are made of primitive stroke techniques that use the strokes spatial, strokes combination, and their relations to form a structure or character. However, if the connections among the strokes cannot be formed and detected sufficiently the characters cannot be recognized according to the research. Incomplete and missed strokes are common problems while doing handwritten character recognition systems. Thus, this approach is not much suitable for HCR.

Furthermore, the researchers propose two kinds of datasets which are used for training and testing the proposed system (Bigun, 2008). In which the first dataset contains grayscale images of with 300dpi resolution and the images are collected from EOTCs, this dataset is named as Group1. And the second dataset contains grayscale images with 300dpi resolution and they are collected from different public writers, and it's named as Group2. The second dataset (Group2) was further divided into two datasets which are named as Group2a and Group2b. Both groups contain handwritten Amharic characters which are written from different writers. Group2a dataset contains 59 pages which are written by different 177 writers by copying them from sport, politics, and economics newspapers. And Group2b contains handwritten characters written by 152 different

writers which are written three times. The characters which are included in Group2b dataset are those characters which are not often used for writing or speaking. So, Group2b contains handwritten characters which are rarely used nowadays. Therefore, the datasets are separated based on the place where they are collected. Moreover, the researchers used a Gaussian filtering technique for smoothing the images (Bigun, 2008). They used a 5*5 filtering matrix for the first dataset (Group1) and 3*3 filtering matrix for the second dataset images (Group2a and Group2b). Lastly, the proposed system obtained an accuracy of 87% for Group1 dataset, 76% for Group2a dataset, and 81% for Group2b dataset after the system is trained by the prepared datasets. The accuracy obtained for the first dataset which collected from the EOTCs shows us that their approach was suitable for ancient documents recognition.

2.5. Summary

So far, we have discussed handwritten and ancient document recognition processes for Amharic, Geez and other languages. Each of the discussed researches proposes different kinds of recognition approaches. Such approaches include deep network, convolutional neural network, Hidden Markov Model, and other models. Also, we have seen many kinds of papers which are done on Bangla handwritten character recognition, Handwritten Latin Character recognition, Amharic handwritten character recognition, and Amharic handwritten word recognition. Furthermore, some of the systems followed a segmentation free technique for image preprocessing.

Being obvious that most of the ancient documents are very noisy and degraded documents they require some kind of efficient preprocessing especially noise removal techniques. But some researchers conducted direct recognition systems without applying any binarization technique and their result seems quite brilliant. Even if some researchers suggest binarization free technique results in efficient recognition accuracy. There are also other research works suggesting that binarization makes the recognition of ancient documents as efficient as possible than that of the binarization free approach.

Additionally, there are many kinds of techniques and approaches to handwritten character recognition such as ANN, SVM and so on. The neural network approach seems much successful and better than the other. Especially the deep connected convolutional neural networks result with a greater handwritten character recognition accuracy. So, as we have discussed it in the methodologies of the development section, we are going to use deep convolutional neural networks (Deep CNN) in order to develop a recognition system for the ancient Ethiopic Scripts. The proposed system will necessarily include line, word, and character segmentation, smoothing,

binarization and noise reduction methods and deep convolutional neural network-based classification model.

CHAPTER 3
METHODOLOGY

3.1. Overview of Research Methodology

Research methodology describes the type of methodology used to assure that the research is completed successfully. In our context, we have divided the research methodologies into two broad groups which are used while conducting the design and implementation of the proposed system. The research methodologies that we have used for the proposed system are grouped as data collection and system development methodologies. The data collection methodologies used are intended in collecting images for preparing dataset that will be used for training and testing the proposed system. Besides the design and development methodology explains the methodologies used to design and implement the proposed system.

Moreover, at each step, it may require a different and independent research methodology to be applied for obtaining the expected deliverables efficiently. Thus, each methodology we use on each layer affects the deliverables of the subsequent phases. Although, there are many methodologies of postprocessing they are not discussed in this chapter because they are not covered in this thesis. In general, we have explained and reasoned out the methodologies that are used to design and implement the prototype of the system. Even though the literature review is also considered to be among the design and development methodologies it is not discussed in this chapter. Since it has been clearly explained and discussed in the previous chapter.

3.2. Data Collection Methodologies

Data collection methodologies are the methods that are used to collect any kind of data that are used while carrying out the research for the purpose of achieving the research objectives. In our case, the data collection is directly related to the process of image collection for the aim of preparing a dataset. It's a critical thing to prepare a dataset so that it can be used for training and testing the proposed system. Generally, the images are collected from ancient scriptures, missals, homilies, and documents which are usually found in Ethiopian Orthodox Tewahedo Churches (EOTCs). Some of the documents that we have found are very degraded making the process of preparing dataset difficult. We have used two common image collection methods which are using digital camera and scanner. They are explained in the following sections.

3.2.1. Image capturing using a digital camera

A digital camera is the primary data collection devices that we have used to collect the images in order to prepare the dataset. We have used a digital camera as a primary data collection tool because of the difficulties and problems of using a scanner for scanning the documents. Among the different reasons, the size and conditions of the documents are the major problems. Some of the documents are too big for scanners and also the ingredients used to prepare the scripts or documents are not suitable for scanning. This is because of the nature and behavior of the writing materials such as the codex or scroll used to write are very thick, sometimes the bindings used are made of wood and other thicker materials. These and other behaviors of the ancient documents requires collecting images with digital cameras. Some examples of the documents that are difficult and impossible to scan images through scanner can be found in chapter 4.

3.2.2. Image capturing using scanners

Scanning document is the usual and recommended way of collecting data for image processing and related fields such as handwritten character recognition. However, in the case of collecting data from ancient and too aged documents, this method becomes sometimes impossible. As we have tried to explain it above; the size, style, and materials which are used to prepare most of the documents make the scanning process nearly impossible or hard. Nevertheless, we have used scanners for collecting images whenever possible. Although, every image used for training and testing the proposed system are gathered either through a digital camera or scanner.

3.3. System Design Methodologies

In this section, the techniques used to implement the prototype of the system are described with their benefits. This section covers all the preprocessing techniques starting from the grayscale to binarization and noise removal methods. Also, the approaches used as a feature extractor and character classifier for the proposed system are presented in this section.

3.3.1. Preprocessing methodologies

It’s obvious that building an efficient character recognition system requires more complicated processes. The selection of an efficient character classifier and feature extraction approach highly depends on the complexity of the images containing the characters to be extracted. Since we are dealing with aged ancient handwritten documents, the complexity of the images is obvious. The complexity of the image is dependent on the unclear background, the invisibility of the characters, broken or incomplete character, and so on. Thus, the feature extraction and classification stage

highly depend on the preprocessing stages. With the intention of developing an efficient prototype, we have used some relevant preprocessing techniques.

These preprocessing techniques include smoothing, noise removal, skew detection and correction, and morphological operations. Almost we can say all ancient documents are highly noisy and require some advanced level noise removal and smoothing techniques. We have used a Gaussian kernel of 3*3 for blurring and smoothing the images in order to remove edges and unwanted noises. After applying the Gaussian blurring technique, we have used non-local mean denoising for removing the rest of the noises that are left on the images. Furthermore, we have used opening and closing methods for reforming the characters as well as removing and adding pixels for the purpose of correcting the effects of noise removal and smoothing methods.

Furthermore, a skewed document can happen during scanning and copying the documents. Skew can be visualized as a slope of the text lines in the x-axis, this should be correct otherwise it will affect the accuracy of the character recognition system (Cheriet, 2007). Sometimes skewness of documents can happen intentionally to underline something which is relevant. However, in many cases it's unintentional and it requires to be eliminated as much as possible. We have used a Hough Transform (HT) method to detect and correct skewness from the images. Initially, it calculates the rotation degree of the image and then rotates the document according to the calculated degree to correct the skewness.

3.3.2. Binarization methodology

In general, there are three types of binarization techniques which are global, local, and adaptive thresholding methods. The global thresholding technique uses only one threshold value for the entire image however, the local thresholding method divides the entire image into patches and uses one threshold value per each patch. Additionally, the adaptive thresholding uses a different threshold value for each pixel in the image. We have used a global thresholding method called Otsu thresholding which is sometimes referred to as “Optimal thresholding” and considered to be the best and fastest method according to researches made for degraded documents in (Liu et al., 2003) and (Liu et al., 2003).

3.3.3. Segmentation methodologies

Segmentation is the critical part of character recognition which involves segmentation of lines, words, and characters respectively from the inputted image and making the image ready for feature extraction. We have used counter tracing for each segmentation process. Contours tracing is one

of the difficult and relevant tasks of image processing. It involves tracing and getting the border or boundaries of the characters. Hence, if we can locate and trace the boundaries of each character indirectly it means we exactly know where the characters are located on the images and the extraction process will become possible and easier. We have used counter finding algorithms that trace the contours of each character from an image. this algorithm was originally designed by Satoshi Suzuki and Keiichi Abe (Suzuki and Abe, 1985).

3.3.4. Classification and feature extraction methodologies

There are different kinds of character classification approaches which are used by many researchers such as Support Vector Machines (SVM), structural approaches, statistical models and Artificial Neural Networks (ANNs) they are discussed in detail in Chapter 4. Among those classifications, we have used a deep neural network-based character classification approach. The proposed system's prototype is implemented using modern deep convolutional neural networks (Deep CNNs) as a feature extraction method. CNNs have a special type of layer called convolutional layer and it's usually followed by a pooling layer. The main goal of the convolutional layer is to extract the relevant features of the characters and provide the features to the classification layers. Usually, after the last layer of the convolution model, a dense or fully connect layer will be placed to perform the classification process, using the extracted features.

Nowadays, the CNNs approach is becoming the most efficient feature extraction approach in image processing when it is mixed with dense layers of classifiers. The dense layers use the features which are extracted from the convolutional layers after a convolution operation is performed with the kernel or filter vector. This classification approach shows a greater improvement on recognition accuracy of a MINIST English handwritten dataset which hasn't been achieved by any other classifiers so far. Furthermore, the CNN models are much efficient than the other approaches based on the computational power required to compute the weights of each node and it's briefly explained in chapter 4. Also, a DNN model is designed that doesn't have any CNN layers for comparing the results obtained by the proposed system.

3.4. Summary

In this chapter, we have discussed the methodologies that we have used for designing and implementing the prototype of the proposed system. We have divided the methodologies discussed into two broad groups as methodologies of data collection and system designing and development methodologies. In image collection methodologies we have explained two types of image capturing methods which are using digital cameras and scanners. Even though we have used

scanners rarely because of the different reasons which are explained alongside with each method, its included as a data collection methodology.

Furthermore, in the second methodology group, we have covered the methodologies used to design and implement the prototype of the proposed system. The flow we used to describe and present the methodologies for design and development is more general. The design and development methodologies are sectioned into five different parts which are preprocessing, binarization, segmentation, feature extraction, and classification techniques. The methodologies discussed in the preprocessing section are methods used for smoothing, noise removal, skew detection and correction, and morphological operations. Also, we have explained the method and technique used for segmentation.

CHAPTER 4
ANCIENT ETHIOPIC SCRIPT RECOGNITION

4.1. Overview of Ancient Geez Scripts

Unlike many other African countries, Ethiopian scripts are written using indigenous writing style and alphabets called Geez. The most common and famous language in Ethiopia which is used for writing and any kinds of communication service is Amharic in recent days. Nowadays the Amharic language is spoken by more than half of the Ethiopian peoples. Amharic alphabets are believed to be derived from Geez alphabets. In fact, Amharic alphabets contain all of the Geez alphabets plus additional more alphabets. Even though, both languages use similar kinds of alphabets they have a big difference based on their grammar, the way they are spoken, and other reasons. Almost all of the ancient Ethiopic documents are written using Geez alphabets.

Currently, we can find a lot of ancient religious, historical, and liturgical documents which are written in Geez. Most of these documents are not converted into digital format and they are not easily accessible. As a result, the documents are in danger of disappearing. Also, some of the documents we found are really difficult to convert them into digital format. Sometimes it is impossible to read and understand them even by our own eyes. Usually, these kinds of problems are a result of damaged parts or sections of the documents and pages. In our research, we have seen that most of the ancient documents that are found are degraded and aged. Moreover, some of the documents are completely damaged making the recognition process nearly impossible. However, most of the documents which are found in the EOTCs are kept safe and they can be digitized. Most of the documents which are kept in the EOTCs are religious and holy books.

In ancient times relevant information is written in a paper-like material which is made of animal skins and it's called in native words "ብራና" or "Branna" which is a codex. The codex has been the primary choice of many ancient civilizations for the purpose of writing and keeping information safe. Because of the nature of the codex, the documents will take a longer time to be aged and degraded (Siranesh and Menore, 2016). Most of the ancient Ethiopic documents can be found in EOTCs, libraries, and private or nonprivate institutions. But the majority of the documents can only be found in Orthodox churches of the country and getting privilege for accessing them might not be easy. Because of many reasons, the documents are considered as top-secret treasures of the

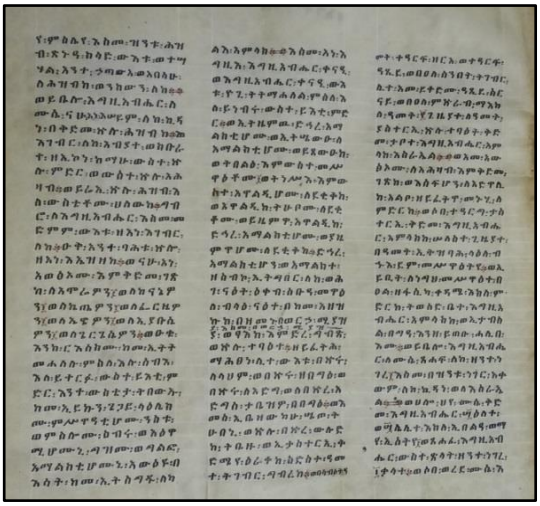
EOTCs. So, finding enough documents easily for the preparation of the dataset is not possible even with a proper privilege to access them.

Geez, was declared as the official language for the Ethiopian people at the beginning of the third century (Ethiopian Literature, 2018). This was as a result of the king's order "King Ezana" who was ruling the country declaring Christianity as a national religion of the country around 340 AD. Nevertheless, Christianity was in practice in the country before the king's declaration and so geez language. In this context, we can understand that the documents and literature which survive until now are almost more 1600 years old. Correspondingly, many Ethiopian ancient documents can be found in some European countries (Ethiopian manuscript collections, 2019). Most of the documents are believed to reach those countries at the beginning of the 15th century. Those documents were a copy and original documents which were taken by some Ethiopian and Egyptian religious travelers (Ethiopian manuscript collections, 2019). The travelers were going to the holy land of Jerusalem for worshiping. There are around 2700 ancient Ethiopian documents collectively in three European countries holding them in their libraries according to (Ethiopian manuscript collections, 2019).

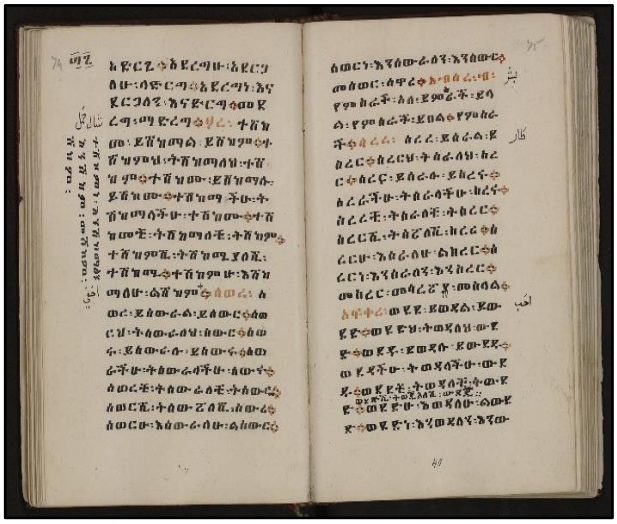
4.1.1. Ancient Geez script writing techniques

Most of the ancient Geez documents don't follow a common and standardized writing style. The documents don't have a common size, some of them are as smaller as a pocket-sized notebook and some have a size of 40-50cm in height. Additionally, unlike the most modern documents, the pages don't have a single column. There is no common way of using columns in pages but they are sometimes divided into two or three columns depending on the size of the documents. And also, unlike modern handwriting styles, there is no common ink color for writing the documents or they don't use a common color. Even in a single page, there might be used more than a single color. Most the ancient Geez documents are written using two different kinds of colors which are black and red. Especially, while writing religious documents the writers use red color to indicate that the word is a holy word or names as the name of God or saints. Figure 4.1 shows some of the common writing styles of the ancient Geez documents.

(a) A book having 2 columns per page



(b) A book having 3 columns per page



(c) A book having a single column per page

(d) A small pocket-sized book

Figure 4.1: Common writing styles of ancient Geez documents

4.1.2. Geez alphabets

Geez, language is sometimes referred to as Ethiopic language and its characters are called “abugida” or in a native writing “አቡጊዳ”. Geez, alphabet contains 26 base characters and each base character has an additional 6 more characters which are the result of the addition of the vowels. Unlike English or Latin languages, the Geez alphabets don’t have a separate vowel character. However, the vowels are implicitly attached with the base characters to form a new sound and shape. Therefore, the total number of Geez alphabets is 182 including both the base and extension

characters. This total number of the Geez alphabets does not include the punctuation marks and numbers of the language.

Furthermore, the Geez alphabets don't have uppercase and lowercase characters unlike the Latin alphabets this will make the recognition and other processes much easier. Hence, we have tried to explain in the previous sections the proposed system is only designed to recognize the 26 base character excluding the extension characters. It's obvious that we can't recognize a Geez document by training a neural network with only the base characters. However, we can conclude that if the deep convolutional neural network approach is suitable for recognizing ancient Geez documents. Also, expanding the neural network model to classify all the 182 characters is easy we just need to prepare the dataset in order to train and test the system.

Generally, the Geez alphabets which are not base characters are much similar to their base character. Only some kinds of strokes are added in the base characters in order to create a derivative of the characters. Typically, a single stroke is added on the base character but not always, for example: "ገ" and "ጉ", "ለ" and "ሊ", or "ሠ" and "ሤ". For example, one of the base alphabets of the Geez "አቡጊዳ" is "መ" and we can derive a new character from the base character. We can achieve this by adding the vowel "ኡ" which is similar to the Latin vowel "U". Evidently, if we want to attach the vowels in Geez the vowel is attached implicitly with the character and they combinedly will derive a new character. Therefore, if we add the vowel "ኡ" to the base character "መ" they will combinedly form "መ፡" which have a more similar shape with the base character with a single additional stroke. The following table 4.1 shows all the characters and the derived characters of Geez alphabets.

Table 4.1: Geez alphabets

	E	U	I	A	IE		O
H	ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
L	ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
H	ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
M	መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
S	ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
R	ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
S	ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
Q	ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
B	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
T	ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ
H	ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
N	ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
A	አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ
K	ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
W	ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ
A	ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ
Z	ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ
Y	የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ
D	ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
G	ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ
TT	ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
PP	ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ
TS	ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ
TS	ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
F	ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
P	ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ

4.1.3. Features of Geez alphabets

Most of the Geez alphabets have a higher degree of morphological structure similarities. Making the recognition process much difficult to recognize the characters as expected. Sometimes even for our own eyes will become difficult to differentiate accurately among the characters. We can take an example of "ለ" and "ሰ", "ደ" and "ጸ", "ሃ" and "ነ", or "ከ" and "አ" also there are other bases and derived characters having similar morphological structures. From those examples, we can see that Geez character have a morphological similarity which will make the recognition process much more difficult than expected. However, we can create a grouping mechanism for the characters based on their morphological structure in order to understand them easily. Sometimes researchers group them based on their frequency of use.

Basically, if we can group the character based on their morphological shapes that will make a mechanism to differentiate and understand the alphabets easily. Among the total of 26 base characters, 2 of them are three-legged we can group them as a three-legged character based on their morphology. Moreover, 9 base alphabets of the total characters have two legs which mean, one-third of the total characters of the Geez alphabets are two-legged. Additionally, there are 7 characters which are having only a single leg and some of them might have a smaller circle like a stroke on their single leg. Finally, four of the total base character have a rounded shape on their bottom and the other has horizontal and irregular shapes on the bottom side. In table 4.2 we have summarized the alphabets of the Geez language based on their structure. As the table shows we have divided the characters into six independent groups based on their morphological structure. Note that most of the derived characters have similar morphological structures with their base characters with only additional stroke on their bottom or top depending on the characters or style.

Table 4.2: Features of Geez alphabets

Character Group	Characters	Count
Base characters having one leg.	ቀ, ተ, ጎ, ነ, የ, ገ, ፐ	7
Base characters having two legs.	ለ, ሰ, በ, አ, ከ, ዘ, ደ, ጰ, ጸ	9
Base characters having three legs.	ሐ, ጠ	2
Base characters having rounded bottom.	ሀ, ወ, ዐ, ፀ	4
Base characters having a horizontal stroke on their bottom.	ረ, ፈ	2
Base characters having an irregular shape on their bottom.	መ, ሠ	2
	Total = 26	

Moreover, like any other language, the Geez language has its own way of representing punctuation marks. There are around nine punctuation marks which are going to be used while writing Geez documents. As we know it, in many popular languages like English uses spaces to separate words. However, in Geez scripts we don't use spaces to separate words rather we use a colon like a symbol "፡". Though, in recent days most Geez or Amharic writers don't use these punctuation marks especially the word separator, section mark, and paragraph separators. However, most of the punctuation marks basically word separator and comma are usually used by the ancient Geez writers, unlike the modern writers. Also "፣" is used for listing things consecutively or it serves as a comma for Geez writing systems. Although Geez may have some kinds of additional punctuation marks that cannot be found in Latin alphabets or other popular languages. In table 4.3 we have tried to show the most commonly used punctuation marks of the Geez languages and their equivalent meanings and usage as possible.

Table 4.3: Commonly used punctuation marks of Geez

Punctuation	Name	English Equivalent	Usage
፡	Word separator	Space ( )	For separating words
፠	Section mark	Section mark (§)	For referring a section of a text, a page or a book
።	Full stop	Period (.)	For terminating a sentence
፣	Comma	Comma (,)	For a slight break between words
፤	Semicolon	Semicolon (;)	For separating two sentences having a similar idea
፥	Preface colon		For presenting a speech from an expressive preface
፧	Question mark	A question mark (?)	For indicating a statement is a question
፨	Paragraph separator	New Paragraph	For indicating a beginning of a new paragraph

4.1.4. Main differences between ancient and modern Geez writing systems

The basic difference between many ancient and modern writing systems lays with the type of materials used on the time of writing the documents. Most of the ancient Geez scripts are written on a codex called branna and this codex is prepared from animal skin. The primary reason for the ancient literature to use animal skins is for the purpose of keeping the documents safe and endured. Obviously, animal skins are much stronger and durable than that of the modern writing materials, which are basically papers made of plants. Moreover, there are other main differences among them just like the size of the characters, colors used to write, the background of the scrolls, and the material used to bind the documents. The main difference between the ancient and modern document are listed and described as follows:

- **Writing papers:** Most of the ancient Geez documents are written on paper like material which is called branna, it's basically a parchment made of animal skin in which modern documents are written on papers which are made from plants. But sometimes we can find ancient Geez information written on the stones and walls of churches too.
- **Writing ink:** It's obvious that most of the modern documents are written using a pen but, sometimes other kinds of writing materials used like a pencil, fountain pen, and ballpoint

pains. However, ancient Geez scripts are written using ink which is usually made of plant's leaves and a pointy material basically prepared from bird's feathers.

- **Backgrounds:** Generally ancient documents background are noisy and very poor in quality. Also, they don't have a common background color like many modern documents. However, modern documents background is usually white and clean.
- **Size of the characters:** In modern times the characters used to write the information in papers are smaller and equal in size. However, the ancient scripts don't have uniformity on character sizes and shapes; sometimes they are too big or sometimes too small.
- **Writing styles:** In ancient times while writing the Geez documents the writers use an artistic kind of writing techniques in order to attract the readers. Though, this is kinds of artistic drawings are not usually used in modern documents. However, in modern time writers might use some kind of cursive or especially writing techniques to attract readers.
- **Punctuation marks usage:** Moreover, ancient Geez documents mostly use punctuation marks especially word separator "፡" and full stop or period "።", unlike modern document doesn't usually use the punctuation marks. The modern Geez writing techniques are mixed with the Latin punctuation marks.
- **Document binding materials and techniques:** The types and methods of binding used by the ancient document writers are not like the modern documents binding techniques and materials. Most of the Geez ancient document uses binding material which is made of animal skin and wood, it might be 2 to 3cm thick. However, there are also some documents which don't have a binder at all or it is damaged or removed by someone.

Generally, most of the ancient document writing materials and techniques make the recognition process more difficult than modern time materials and techniques. As we have tried to explain above most of the ancient Geez document writing materials and techniques are not good for the recognition process. However, in order to meet our main goal, we need to create a system that will decrease those difficulties and help us to achieve the main objective of the thesis. We will provide some kinds of binarization, nose removal, smoothing and other techniques which are suitable for studying degraded or ancient documents. And we will be covering those techniques one by one on the coming sections of this chapter.

4.2. Offline Handwritten Character Recognition System

In general, many character recognition systems either handwritten or printed recognition follows a common step by step process with a little variance. These variances may include differences in

the process, techniques and implementations strategies. However, some researchers proposed a recognition system without common recognition steps. Just like the research made in (Yousefi et al., 2015), the researchers proposed a recognition system which doesn't include a binarization technique. Even if binarization is the most commonly practiced process of many handwritten recognition systems, some might not include it. Furthermore, in another research (Pal et al., 2007) the researchers conducted a character recognition system that doesn't include any kinds of noise removal techniques. They don't use any noise removal technique because of the approach they used for the classification step. The classifier just takes the raw data of the images and performs the classification operation, most likely we can say, it jumps the preprocessing stage. Moreover, many researchers conducted a recognition system that doesn't follow the most common and standard step and obtained a better result. Though, in this paper, we will be following the most common and standard steps of any character recognition systems with some relevant changes. In Figure 4.2 we have shown the general and most common steps of a handwritten character recognition system. However, many researchers perform a recognition process without the postprocessing phase. But they end up suggesting that, if postprocessing process is added to a system it will increase the recognition accuracy.

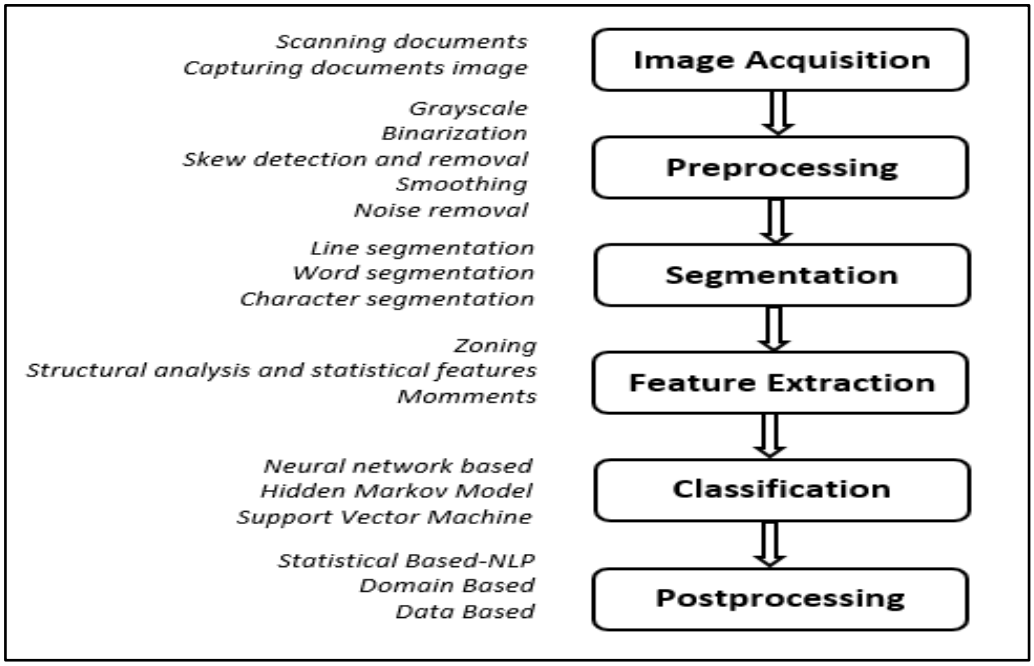


Figure 4.2: General steps and process of handwritten character recognin system

4.2.1. Image Acquisition

Image acquisition is the first and most important step of any printed or handwritten character recognition systems. Generally, for any kind of image processing system image acquisition is the

primary and fundamental task before conducting any kind of activities. Note that each step of the character recognition process is highly dependent on the previous. Basically, most documents that are used for images recognition are captured using a scanner. However, sometimes there can be documents that are difficult to scan using a scanner because of the materials used. Thus, digital cameras might be used to capture the documents or section of the relevant documents' pages.

4.2.2. Image Preprocessing

Once the relevant images are collected the images will pass through the preprocessing stages. The purpose of the preprocessing stages is to remove irrelevant pixels to form a better representation of the images and make them ready for the subsequent stages. The preprocessing stage involves different step by step processes such as grayscale conversion, binarization, smoothing, skew correction and noise removal.

1. **Scale down:** Before starting any kind of processes on the images, each image should be scaled down or normalized to a common size. This will improve the speed and accuracy of the recognition process. Images captured with a digital camera are bigger in size and if we start processing these images directly the process will take a longer time. Therefore, before processing the images we need to scale down the images to a common size of mostly between 300dpi and 600dpi. This will improve the recognition process by reducing the memory load and execution time of the system.
2. **Grayscale**: Usually, the images which are collected using a scanner or camera are RGB or colored images. And if we are dealing with an image processing approach, which can't be affected by the color of the images then, it's a good practice to convert each image into a grayscale image. Basically, images are stored in a three-dimensional array or vector matrix but we can convert those images into a grayscale image. Then the grayscale image is stored as a two-dimensional matrix making the recognition processes much easier. Commonly, in colored images, every pixel or point is denoted with three values of RGB, in which each value is represented as an 8-bit integer by the range of 0 to 255. However, in grayscale images, every point is expressed with a single value that varies between 0 to 255. Each pixel value in grayscale images represents how much that point is bright or white. Basically, a pure white is represented with 255 and a pure black with 0. Figure 4.3 B shows the grayscale version of the original images. There are different methods of grayscale conversions as shown in Equation 4.1, 4.2, and 4.3.

$$gray = (red * 0.3 + green * 0.59 + blue * 0.11) \quad (4.1)$$

$$gray = (\min(red, green, blue) + \max(red, gree, blue))/2 \quad (4.2)$$

$$gray = (red + green + blue)/3 \quad (4.3)$$

3. **Binarization:** Binarization is the most important and critical process of both printed and handwritten character recognition. There are many different methods of binarization technique. However, there is no better and generalized binarization technique. Some binarization techniques might be suitable for some kind of recognition procedures and some might not be suitable depending on the nature and type of the recognition system being developed. However, letting the user chose binarization methods at runtime will result in a better result. Binarization process involves converting the pixels of the grayscale image into either completely white (255) or completely black (0) based on a given threshold value. Thus, the image pixels are represented as either black or white rather than gray. Binarization techniques are broadly grouped into two main categories which are local and global binarizations as we can see it in this paper (Puneet and Garg, 2013). These binarization techniques are grouped based on the way they chose a thresholding value. In the global thresholding method, a single threshold value is used for the entire image. However, in the case of local thresholding, a number of threshold values can be used for different sections of the image (Chaki et al., 2014). Otsu thresholding is considered as one of the most successful global thresholding methods. This method uses histogram shaping and clustering technique for converting the gray image into binarized images (Otsu, 1979). Technically the Otsu binarization algorithm dived the image in two distinct parts as a background and foreground parts of the image. Then it calculates the minimum threshold value that separates the background of the images with the foreground of the images. Let z is the number of grayscale layers and *t* is the layer value for which every pixel value greater than *L* is black and the rest are white. Then if we consider *c1* is a set of all white pixels and *c2* is a set of all black pixels then, *p1= {0,1, 2, ... r}* and p*2= {r+1, r+2, ... z}*. So, we can find the threshold value using the maximum ratio of the class variance and total variance using Equation 4.4.

$$= \frac{\sigma_B^2}{\sigma_T^{2\cdot}} \quad (4.4)$$

Where sigma B and T are the class variance for black pixels and total variance respectively.

$$\sigma_B^2 = w_0 w_1 (\mu_1 - \mu_0)^2 \quad (4.5)$$

$$w_1 = \sum_{i=0}^{t} p_1 w_1 = w - w_1 \quad (4.6)$$

In which both w_0 and w_1 are the functions of r, p_i the probability of the i^{th} pixel of the grayscale image and W is constant, then we can give p_i. The new values can be found using eq 4.8 in which μ_T is constant and u_t, u_0, and u_1 are functions of t.

$$p_i = n_i / n \quad (4.7)$$

$$\mu_t = \sum_{i=0}^{l-i} i p_i, \mu_t = \sum_{i=0}^{t} i p_i, \mu_0 = \frac{\mu_t}{w_0}, \mu_1 = \frac{\mu_T - \mu_t}{1 - \mu_0} \quad (4.8)$$

Then the final sigma T can be determined using the following equation

$$\sigma_T^{2\cdot} = \sum_{i=0}^{l-1} (i - \mu_t)^2 p_i \quad (4.9)$$

Figure 4.3. (c) shows the Otsu binarized version of the original image based on the Otsu binarization technique.

(a) Original colored image (b) Grayscale image (c) Otsu binarized image

Figure 4.3: Image preprocessing stages of the handwritten document

4. **Noise removal:** Usually, handwritten documents are very noisy and degraded documents so, some kinds of noise removal techniques should be applied to obtain a better recognition result. Generally, noise removing is a critical and essential procedure while dealing with handwritten recognition systems. There are different kinds of image filtering approach like Gaussian, non-local means denoising (Buades et al., 2011), and media filtering which are the most known techniques. Generally, these filtering approaches are applied to the image to remove unwanted and noisy pixels points from the images. The following Figure 4.4 shows an example of noise ancient document.

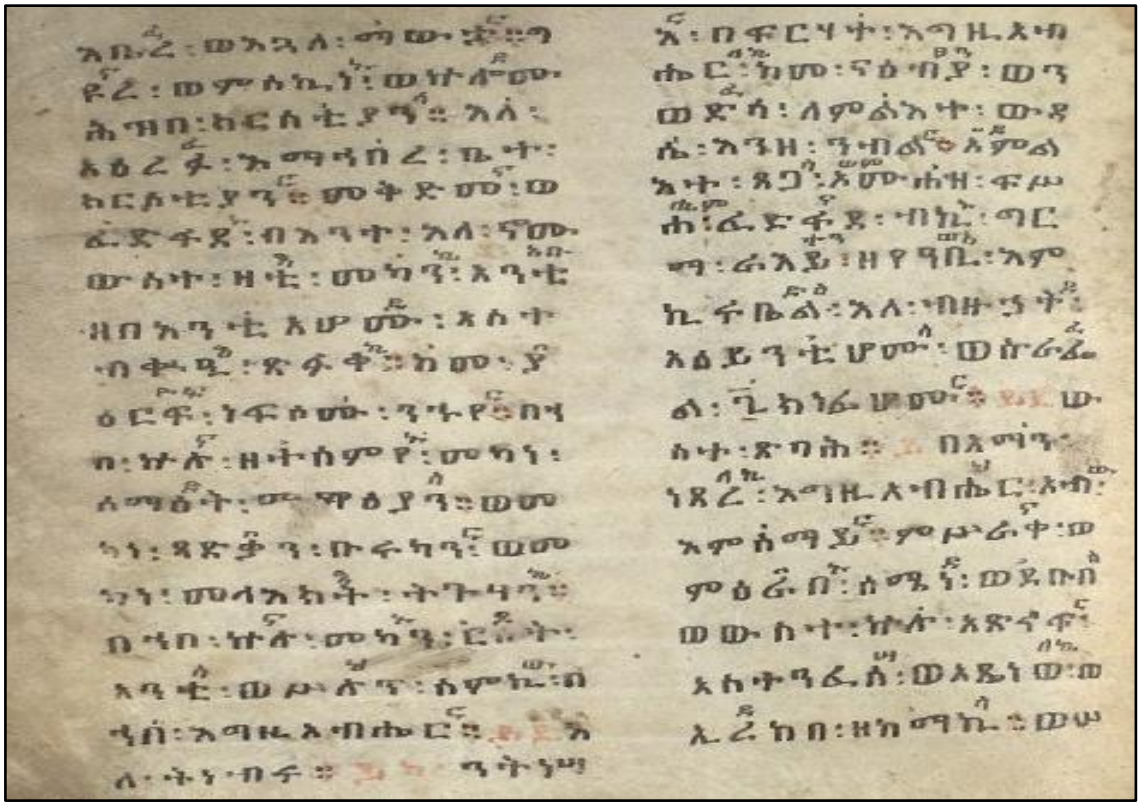


Figure 4.4: Example of noise ancient Geez document

5. **Skew detection and removal:** Skew of documents can happen accidentally or intentional but usually occurs unwantedly. However, sometimes document or a section of a document skewing can be used to emphasize statements and for efficient space use while writing. Also, in ancient Geez scripts, skewness might happen deliberately but it's not a usual thing. Moreover, document skew can happen also in the scanning or image capturing process. The skew of the document can affect the accuracy of the recognition system highly so, it needs to be corrected before the recognition process starts. Basically, skew detection and correction step is preceded by the binarization and noise removal steps. However, there is no efficient and better skew

correction mechanism that can be applied for any kinds of systems but Hough transform method is considered to be the most efficient (Arulmozhi et al., 2012) technique on some cases. The first step is to find the rows and columns of the image and find the first and the last black pixel of the rows and calculate the rotation angle of the document if it's skewed then rotate the image based on the obtained rotation degree of the document. There are different kinds of skew detection and correction techniques and algorithms such as (Verma and Malik, 2015):

- Hough transforms,
- Principal component analysis,
- Extreme points,
- Random transform.

4.2.3. Segmentation

Generally, the efficiency and accuracy of any character recognition system are highly dependent on the preprocessing and segmentation stages. An efficient and better segmentation technique will result in a better and efficient recognition system. This step is carried out before the feature extraction and classification stages and the output of the segmentation process will be the input of the feature extractor. Technically segmentation process involves three sequential steps which are line, word, and character segmentation. Primarily, in the segmentation stage, every line of the image is segmented row by row. From each individual row, every word will be segmented column by column. Finally, from the words, each character will be segmented which will make each character ready for the feature extraction stage. However, in some handwritten recognition system, these three segmentation steps might not be required. For example, in English writing styles cursive writing is a usual writing style. This kind of writing system cannot be recognized by a character recognition system basically, they are considered as word recognition system. Thus, for word recognition systems character segmentation is not required, but in many cases, line segmentation is usually done. In any character recognition system, the crucial segmentation approach and stages are the character segmentation step. The character segmentation will define the accuracy of the recognition system. A poor character segmentation result will lead to an incorrect classification of characters. Many segmentation algorithms use vertical and horizontal projections for the segmentation process. Horizontal projection is used for extracting each line from the binarized image and for both word and character segmentation vertical projection is used to extract them accordingly. Usually, there are three kinds of character segmentation techniques according to the paper (Choudhary, 2014) and they are listed as follows:

- Explicit segmentation,
- Implicit segmentation,
- Holistic segmentation,
- Hybrid segmentation.

Explicit segmentation technique is based on a vertical projection method, it first scans the image from the top to the bottom and saves the position of each column. By using those columns, the algorithms divide the word images into individual character images and make them ready for the feature extractor stage. Also, this technique is sometimes called a dissection because, the method follows a character cutting technique from the words (Choudhary, 2014). Whereas, the implicit segmentation approach carries the process of segmentation with the recognition side by side. It just finds for the character from the image directly and it serves as a replacement for the explicit method. It is usually used in HMMs based recognition systems. Moreover, the holistic approach directly deals with the words rather than the character. It's also called a segmentation free technique since it doesn't care about segmenting the characters from the words. This holistic technique is highly dependent on the predefined set of lexicons of words. Therefore, mostly this approach is not suitable for character recognition systems rather its suitable for word recognition systems. Lastly, the hybrid approach has been proposed and presented by many researchers to improve the segmentation process through the application of methods like linear searching and contextual knowledge. Also, some researchers introduce a new hybrid segmentation method in their research papers and improved the accuracy of the segmentation (Blumenstein and Verma, 2001) or (Khan and Mohammad, 2008).

4.2.4. Feature extraction

Once all of the previous processes are completed the feature extraction step starts. Usually, in any character recognition systems, the output of the segmentation stage is a set of a character image but in some cases, the output might be a set of words. These set of characters will act as an input to the feature extraction stage. The main task of the feature extraction step is to extract a unique and diversified feature of each input image or character (Najafiragheb et al., 2016). Thus, in feature extraction, we will extract and obtain the characteristics of the images accordingly. Also, sometimes the character images as itself can be taken as a feature and can be passed to the classifier. (Najafiragheb et al., 2016) There are different feature extraction techniques and we have explained some of them are listed as follows:

- Zoning,
- Momments,

- Structural analysis and statistical features.

As the name zoning indicates the approach divides the entire character pixels into a number of zones and calculates the concentration of the pixels or densities of the pixels on each zone. Then, the calculated densities of each zone will be taken as a feature of the characters. Momments techniques are usually used for object movement detection (Saif et al., 2015). It basically works by detecting the edge of the objects so, in the same way, we can use it to detect the edge of each character and use that result as the extracted feature of the characters. But this technique might not be suitable for OCR systems since its suitable when there is a movement. Furthermore, the third method is considered as a geometrical model and it works based on the topology of each character. This method will extract the features of the character based on their strokes, circle/loops and other shapes that can uniquely describe the character (Najafiragheb et al., 2016).

4.2.5. Classification

Usually, classification is the final step of most character recognition systems however some systems also conducts postprocessing techniques for the purpose of improving the results of the classification stage. Classification involves accepting the features of each character or words in case of word recognition and classifying those features to predict the true classes or labels of the images. Therefore, classification is mainly focused on the decision making of the system. Basically, the classification stage involves two separate and essential process which are: training the system and testing the system with the prepared dataset. As we know it the final aim of any character recognition system is to map the inputted images to get the class label of each character patterns. At this stage, the inputted image characters are classified or converted into their classes or labels that can represent them using ASCII or Unicode. There are many different approaches to a character classification system that are discussed in Section 4.3.

4.2.6. Postprocessing

After the characters are recognized or classified through the implemented classification technique the classifier may not recognize the characters accurately. Errors might happen as a result of poor classification or preprocessing techniques so; these errors should be eliminated as much as possible. The main objective of the postprocessing approaches is to improve the accuracy of the classifiers through the application of word correction approaches. Nevertheless, in some books when they are listing the general steps of character recognition system postprocessing is not included (Cheriet, 2007). Postprocessing techniques will try to correct the output of the classifier in the same way autocorrection works in most editing applications. There are different

postprocessing techniques which can be used for improving the accuracy of the recognition systems. The most common postprocessing techniques are manual, dictionary, and context-based error correction methods (Doush et al., 2018). Furthermore, Natural Language Processing (NLP) can also be used as a postprocessing approach. Unfortunately, any of these postprocessing techniques will not be implemented for the proposed system. However, we can recommend and suggest it as future work for other researchers.

4.3. Approaches of Handwritten Character Classification

Obtaining a better and efficient classification accuracy is the main goal of any handwritten character recognition systems. However, the accuracy of the system may vary depending on the technique or approach of classification used. There are many different kinds of classification methods which have been proposed by researchers (Kavallieratou et al., 2003), (Nasien et al., 2010), and (Pradeep, 2012). Every approach has its own advantages and disadvantages so, it is difficult to select a general, better and efficient classification approach. The most common approaches of character classifier are grouped into five groups which are: statistical, support vector machine, structural pattern recognition, artificial neural network, and hybrid or combined classification systems (Cheriet, 2007).

4.3.1. Statistical methods

Statistical techniques are mostly used in data mining and big data analysis fields. This statistical technique is based on the application of the Bayes decision theory which makes a decision based on the use of probability and costs (Cheriet, 2007). Let's say, from the inputted pattern, an *f*-number of feature measurements $\{x_1, ..., xf\}$ have been extracted. Then the inputted pattern can be represented as *f*-dimensional feature vector $\mathbf{X} = [x_1, ..., xf]$ and X is one of the previously *Z* defined set of classes or labels $\{x_1, ..., xz\}$. Then through the Bayes theorem, we can calculate the probability of each class $P(x_i)$. The statistical methods are most known and use a loss matrix to draw out the probability and result of the misclassified patterns. Also, the loss matrix will tell us the expected loss or the conditional risk of the patterns to be classified. In general, for any character image that can be represented using any dimensional vector, the statistical methods are suitable and can be applied. There are different kinds of methods that use statistical methods such as the K-Nearest Neighbor Method (KNN) and Parzen Window Method (PWM).

4.3.2. Support vector machines

The support vector machine (SVM) technique is ultimately chosen by many researchers like (Nasien et al., 210) and (Malanker and Patel, 2014) due to its accuracy and small usage of

computational power. The main objective of the SVM is to find out the hyperplane from the total number of input features that can uniquely classify the features. Hyperplanes are a decision-making line that will be used to clearly classify the features based on the location where the data lies on the plane using the line. The dimension of the hyperplane highly depends based on the amount of the input features. Basically, the dimension of the hyperplanes is three because most of the time the number of the input features are more than two. However, in some cases the number of the input features are less than three then the hyperplane will be a two-dimensional plane. The decision line is highly affected and depends on the support vectors which are close to the hyperplane. Also, the margin between the support vectors and the hyperplane should be maximized to meet the main goals of the SVM model (Cheriet, 2007).

4.3.3. Structural pattern recognition

Structural pattern recognition is usually used for online handwritten recognition systems rather than offline recognition systems (Cheriet, 2007). These structural techniques use syntactic grammar for uniquely identifying the structure symbols or objects (Ajmire et al., 2012). For this reason, they are sometimes referred to as a syntactic classification approach. In general, this structural or syntenic approaches requires deeper analysis to generate the morphological structure of the objects or the characters. The structure technique records the sequence of the character stroke and assembles these strokes to create the correct structure of each character. The classes that are used to classify the characters are structural templates. In which each character can be represented with one or more templates and the inputted pattern is compared with the previously obtained structural templates. Furthermore, this syntactic technique doesn't provide a general similarity pattern that can identify the character as a whole but it rather provides a similarity index for each stroke or the components that make a character. But pattern or feature extraction in this approach might be difficult.

4.3.4. Artificial neural network

Artificial neural networks (ANNs) are intended for creating a simulation of the human brain. An artificial neural network is composed of a number of neurons which are connected to each other. Neural networks can be classified based on their interconnection as a feedforward network, feedback networks, recurrent networks, LSTMs and etc. Any NN system is composed of a number of inputs, hidden and output neural nodes. There are many different methods of training a neural network. Such methods can be supervised, unsupervised and reinforcement training methods.

A neural network architecture having a single hidden layer is called a single-layered neural network and architecture having more than a one hidden layer is called a multilayer neural network or deep neural network. The efficiency of a neural network is controlled by the different parameters of the network such as learning algorithms, the number of hidden neurons and the number of epochs (Siranesh and Menore, 2016). As we can see it in Figure 1.4 every neuron is connected with another neuron and the input layer is connected with the hidden layers and the hidden layers are connected with the output layers. Therefore, an isolated neuron will not exist in any neural network design. In neural network-based character classifier, the number of classes or labels and the number of output neurons is similar. Generally, the input for the input layer of the neural network is the features vector of the characters. The following Figure 4.5 shows the model of a neural network.

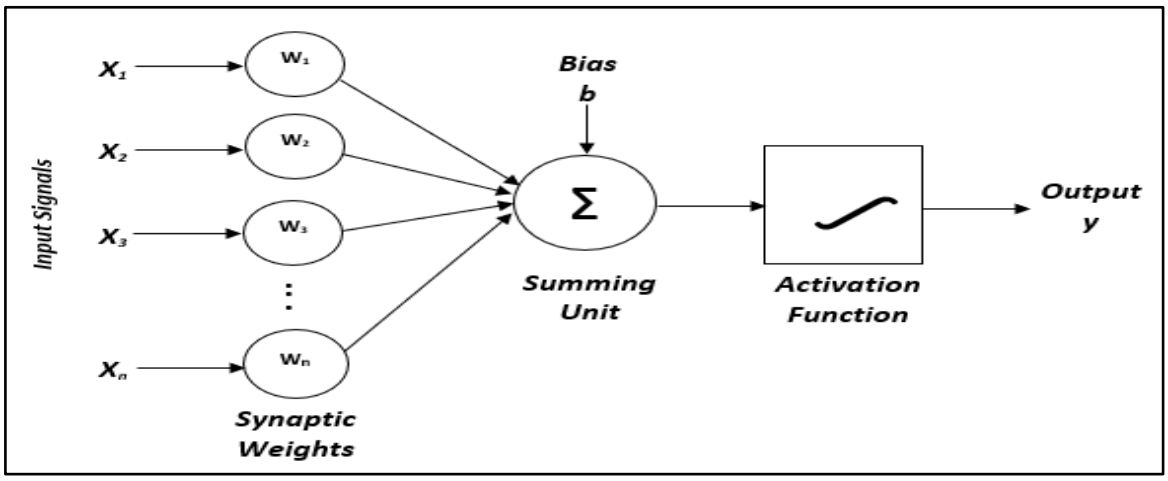


Figure 4.5: Neural model

As the image above describes it clearly a neural network is composed of a serious of input signals or values denoted by X= {x_1, x_2, ..., x_n}. Then the input signals are processed with the weights of the input node which are denoted by *W*= {w_1, w_2, ..., w_n}. Before passing the result to the activation function the input signals are summed with the summing function ∑, the mathematical formula of the summing function is expressed in equation 4.1 where U is the result of the summing. Furthermore, the activation function is responsible for defining or expressing the output of the neural network. In which the activation function decides that the neuron should fire or not. Finally, the result or in our case the character class of the inputted image is returned by *y* after passing through the activation function.

$$U = \sum_{i=1}^{n} WiXi + b \tag{4.10}$$

Activation functions serve as an indicator for the neural network in order to make a decision based on the inputted sum from equation 4.11. There are four different kinds of activation functions which are linear, logistic (sigmoid), hyperbolic tangent, and rectified linear units (ReLU) and they are expressed in the following equations respectively. Among the three activation functions, the ReLU function is the most commonly used and considered as an efficient one for deep neural networks (Eckle and Schmidt-Hieber, 2019). There is another activation function called leaky ReLU which is a variant of the rectified linear unit (ReLU). In which its function can be represented using $g(x) = \max(\alpha x, x)$. Also, other variants of the above listed activations have been proposed by new researches (Gomes et al., 2011). In Equation 4.14 we have shown the ReLU activation function followed by the leaky ReLu function in which it adds a smaller constant value α to the equation.

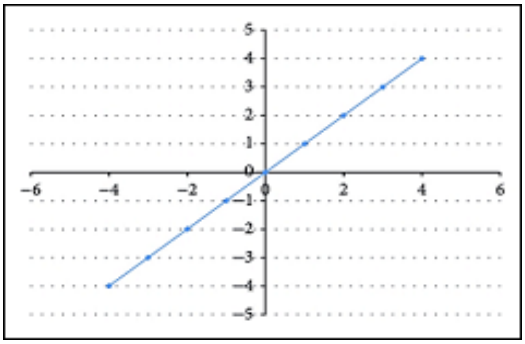
$$g(x) = x \tag{4.11}$$

$$g(x) = \frac{1}{1 + \exp(-x)} \tag{4.12}$$

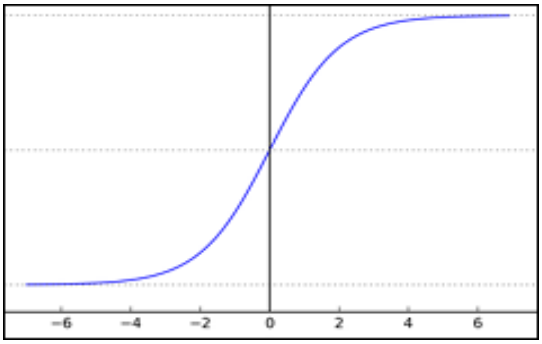
$$g(x) = \frac{\exp(2x) - 1}{\exp(2x) + 1} \tag{4.13}$$

$$g(x) = \max(0, x)$$
$$g(x) = \max(\alpha x, x) \tag{4.14}$$

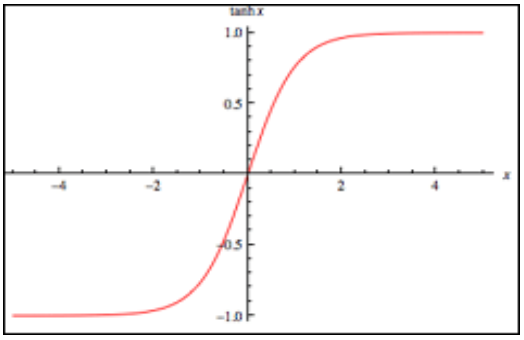
Whereas α is the smallest constant number usually 0.001. However, we can't assign 1 for α because the model will be a linear model. The α constant is used in order to fix the neuron dying problem of the ReLU activation function.



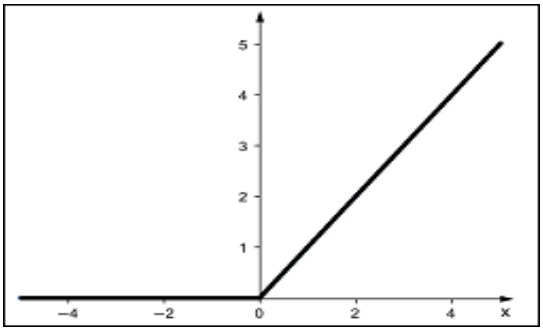
(a) Linear activation function



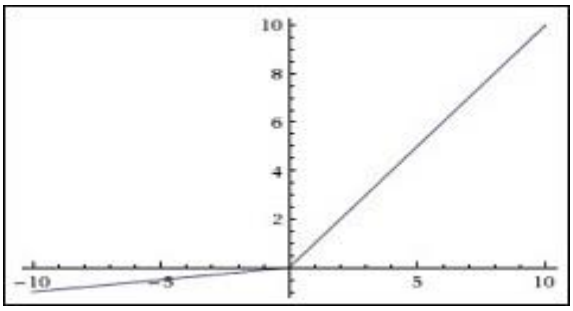
(b) Sigmoid activation function



(c) Hyperbolic tangent activation function



(d) ReLU activation function



(e) Leaky ReLu activation function

Figure 4.6: The most commonly used activation functions of a neural network

4.3.5. Combined or hybrid classifier

These combined or hybrid methods are the combined form of the different classification technique. Character recognition based on multiple classification techniques has been studying by many researchers starting from early 1990 (Cheriet, 2007). Researchers have introduced different kinds of method for combining multiple classifiers into a single system (Shi et al., 2002) and (Liu et al., 2004). These studies have implied that multiple classifiers are better than a single classifier.

4.4. Deep Convolutional Neural Network

A deep neural network is a neural network architecture that has more than one hidden layer. Thus, when we say deep convolutional neural network, we mean a neural network having more than one

convolutional layer (Fukushima, 1980). Convolutional neural networks were originally introduced in late 1980 and they are sometimes abbreviated to as ConvNet or CNN (Cheriet, 2007). The main intention of CNNs when they were originally designed was for the purpose of spatial structures. However, nowadays CNNs have been used for many other pattern recognition fields like speech recognition. The word convolution means a sliding window which slides on the inputted image to retrieve the relevant patterns that express the images. A CNN is compassed of different components which are described below.

4.4.1. Filter windows

Filters in CNNs are small sized windows or in more technical terms two-dimensional vector which is initialized with a random value. The main power of convolutional neural networks lies with detecting the patterns of the images by using the filters. So, we need to define filters for each convolutional layer of the neural network architecture. The movement or sliding of the filters on the original image is called convolving. The filters do the convolving on the original image and calculate the dot product of the filter matrix and the area of the original image in which the filters were convolving. After the filters are convolved on the entire area of the inputted image, we will have a new form of the inputted image matrices which is made from the obtained dot products of each convolving process. Then the new form of the inputted image is stored on the output channel of the network which is called a feature map.

4.4.2. Convolutional layers

The convolutional layers are the hidden layers of the CNNs models, that are responsible for performing all the filtering operation. The convolutional layer is just like the other hidden layer they accept inputs and then performs some operation and pass it to the next layers. The operation performed on each convolutional layer is called a convolution operation also they are known as a cross-correlation operation mathematically. Each convolution layer is responsible for retrieving patterns from the inputted image using the filter window. The patterns that are retrieved using the convolution layers can be strokes, shapes, borders and so on. But when the neural network goes into deeper and deeper layers the patterns that are going to be recognized using the layers become much more complicated. Just like eyes, ears, and an entire object such as faces, characters, cars, dogs and so on. Moreover, when the convolution layers become deeper and deeper the filters become more complicated.

4.4.3. Pooling layers

The main purpose of the pooling layers is to reduce or condense the data collected from the convolutional layers. Although, pooling is applied after each convolutional layer of the network and after each convolutional layer we can define different pooling techniques. There are two common types of pooling techniques which are average and max-pooling (Zha et al., 2015). The max-pooling technique takes a two-dimensional matrix and selects the maximum number from the matrices and condenses the entire 2D matrices into the selected single maximum value. So, the entire matrix is reduced into the maximum single value of the matrix itself.

Similarly, the average-pooling methods take a two-dimensional matrix and condense it into a single value in which this single value is the average of the values of the entire matrices. However, between the two common pooling techniques, max-pooling is better in many cases. Not that when we are saying two-dimensional matrices, we are considering that the images are binary images and they are repressed using matrixes. We can see an example of both average and max-pooling methods in Figure 4.7. The image shows how to reduce the original matrix using two by two pooling matrix while convolving on the inputted image with two strides. Strides are used to define how many pixels at a time the filter move while sliding or convolving for computing the pooling values on each layer. The numbers in Figure 4.7 are values that are used for a representation of the color density of the images. The pooling operation is done by convolving on the input image with a 2*2 filter by sliding each time with 2 pixels sideward.

5.1	3.6	4.1	1.2	2.5	8.2
3.2	7.3	3.2	5.3	8.2	9.3
9.5	6.2	4.7	3.4	5.4	8.2
1.1	3.1	7.6	8.5	4.3	7.6
5.2	2.8	4.4	3.7	8.1	9.1
3.2	6.3	9.3	7.8	5.2	4.1

(a) Originally inputted image

7.3	5.3	9.3
9.5	8.5	8.2
6.3	9.3	9.1

(b) Max-pooling result

4.8	3.45	7.05
4.975	6.05	6.375
4.375	6.3	6.625

(c) Average pooling

Figure 4.7: Image preprocessing stages of the handwritten document

4.4.4. Fully-connected layers

The full-connected layers are sometimes referred to as dense layers in which every input neuron is connected with every neuron of the next layers. The fully connected layers have a SoftMax activation function that is used to predicate a probability of classification for the inputted image

with a range of 0 to 1. Basically, the fully connected layer is placed after the last convolutional or pooling layers. The main purpose of the fully connected layers is to accept the features which are extracted from the convolutional layers. Then it performs the classification process based on the inputted features sometimes. In which their main aim is to classify the inputted images based on the features extracted from convolutional layers.

4.4.5. Training CNNs

The process of training a neural network is simply the process of updating the weights of the neural networks. Unlike, the Multi-Layer Perceptions (MLP) the CNNs neurons' share the same weights among them and also, they are followed with pooling layers. In which the sharing of weights among the neurons and pooling layers will help to decrease the overall weights of the neural network and computational power. Also, an activation layer such as sigmoid and ReLU is placed between the convolutional and pooling layers. Once the image passes these processes the features will be extracted. This extracted feature is flattened into a 1D vector and used by the dense layers for the classification purpose. Let's say I is a two-dimensional image vector and K is filtering window with the size of $w*h$ the convolution process can be given as:

$$(I * K)_{ij} = \sum_{m=0}^{w-1} \sum_{n=0}^{h-1} Km, n * Ii + m, j + n + b \qquad (4.15)$$

The convolution process is just the same as the correlation operation which shown in Figure 4.8 with a slight change which is the filter kernel matrices is flipped 180^0 horizontally. The convolution operation is the overall operation of the convolutional neural network which is used to calculate the weights of the architecture. Then, for each pixel value of the convolution can be calculated using the equation in 4.17. In which $F= \{F_{11}, ..., F_{22}\}$ is the filter vector and $X= \{X_{11}, ..., X_{33}\}$ is the input feature vector and $C= \{C_{11}, ..., C_{22}\}$

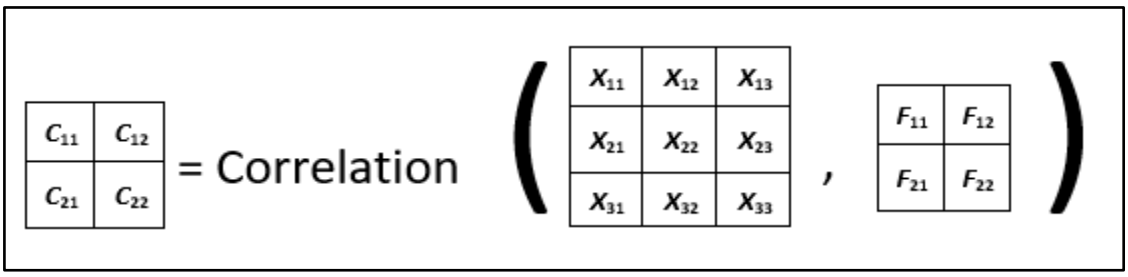


Figure 4.8: Correlation operation of the filter and feature vector

$$
\begin{aligned}
C_{11} &= X_{11}\,F_{11} + X_{12}\,F_{12} + X_{21}\,F_{21} + X_{22}\,F_{22} \\
C_{12} &= X_{12}\,F_{11} + X_{13}\,F_{12} + X_{22}\,F_{21} + X_{23}\,F_{22} \\
C_{21} &= X_{21}\,F_{11} + X_{22}\,F_{12} + X_{31}\,F_{21} + X_{32}\,F_{22} \\
C_{22} &= X_{22}\,F_{11} + X_{23}\,F_{12} + X_{32}\,F_{21} + X_{33}\,F_{22}
\end{aligned}
\tag{4.16}
$$

For the convolution operation, we flip the filter kernel 180^0 horizontally and we can obtain the following convolution operation as shown in Figure 4.9.

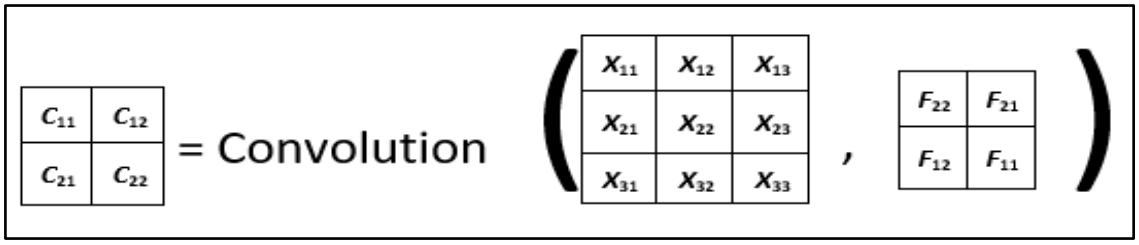


Figure 4.9: Convolution operation of the filter and feature vector

$$
\begin{aligned}
C_{11} &= X_{11}\,F_{22} + X_{12}\,F_{21} + X_{21}\,F_{12} + X_{22}\,F_{11} \\
C_{12} &= X_{12}\,F_{22} + X_{13}\,F_{21} + X_{22}\,F_{12} + X_{23}\,F_{11} \\
C_{21} &= X_{21}\,F_{22} + X_{22}\,F_{21} + X_{31}\,F_{12} + X_{32}\,F_{11} \\
C_{22} &= X_{22}\,F_{22} + X_{23}\,F_{21} + X_{32}\,F_{12} + X_{33}\,F_{11}
\end{aligned}
\tag{4.17}
$$

Backpropagation: is the most commonly used neural network training methods. This method is considered to be the most efficient and fastest training algorithm for neural networks. The process of training a neural network requires a greater computational power so, we need to make our design and training process as efficient as possible. The process of training CNNs can be also called convolution operations which is much similar correlation operation. The process of training a neural model is simply the operation of updating the weights and biases. In the case of backpropagation, the weights are not the only things to be updated also the deltas should be updated. Then, we can compute the updates of the weights which is expressed $\frac{\partial E}{\partial w^{l}_{m',n'}}$, whereas the change in single a pixel $w_{m',n'}$ can affect the loss function *E*. The convolution between the input feature having a size of k_1*k_2 and the filter kernel having a shape of *W***H*, have an output feature map of *z***y* whereas *z* and *y* are given using the following equation. We need to perform much calculus to explain this training operation but we will not be doing that.

$$x = (H - k_1 + 1) \tag{4.18}$$

$$x = (W - k_2 + 1) \tag{4.19}$$

Then we can obtain the gradient component of each weight using a chain role and we can have the fo1lowing final equation.

$$\frac{\partial E}{\partial w^l_{m',n'}} = \sum_{i=0}^{H-k_1} \sum_{j=0}^{H-k_2} \delta^l_{i,j}\, o^{l-1}_{i+m',j+n'}$$
$$\frac{\partial E}{\partial w^l_{m',n'}} = rot_{180^0}\{\delta^l_{i,j}\} * o^{l-1}_{m',n'} \tag{4.20}$$

The summation represents all the sum of the gradients from $\delta^l_{i,j}$ coming from the output layer l in which the double summation is for the shared weights of the filter kernel. The chaining role used above is used for the optimization of the backpropagation training process. The major thing we need to know on the backpropagation training method is that, it uses a chain role to optimize the training process of the CNN architecture. Backpropagation is the tool that Stochastic Gradient Descent (SGD) uses in order to calculate the gradient or derivative of the loss function. After the neural inputs calculated using forward propagation, the output is used backward to update the weights based on the loss function. For the total of *P* prediction classes, the network outputs *y* for which the true class or output is t can be calculated using the mean squared error equation or using cross entropy equation shown in Equation 4.21 and 4.22 respectively.

$$E = \frac{1}{2}\sum_{p}(t - y)^2 \tag{4.21}$$

$$E = -\sum_{p} y \log(t) \tag{4.22}$$

4.5. Summary

In this chapter, we have covered the different steps of character recognition systems starting from the image collection to the classification stages. Also, we have discussed the ancient geez documents what they look like, the writing styles and materials of the ancient Ethiopian scripts and what makes them different from the modern literary. Furthermore, we have discussed the challenges in ancient document recognition specifically the problems in Ethiopic ancient document recognition. Also, we have explained the different types of character classification approaches such as structural, statistical, Support Vector Machines (SVM), and Artificial Neural Network. Among the most commonly used classification methods, we have discussed the

Convolutional Neural Network Classifier. Finally, we have seen how CNNs work their magic and how they are trained using backpropagation algorithm efficiently. Also, we have discussed what filter, convolution, correlation, pooling, dense layer means in detail. The advantages of adding pooling layer into the CNN architecture and the types of pooling techniques used so far have been discussed. How to add the dense layer after the last convolution layer and what they are used or why we need to add a dense or fully connected layer in CNN are discussed in this chapter. In the next chapter, we will explain the design of the proposed system.

CHAPTER 5
DESIGN AND DEVELOPMENT OF THE PROPOSED SYSTEM

5.1. Overview of the Proposed System

The proposed ancient document recognition system constitutes processes such as preprocessing, segmentation, feature extraction, character classification, dataset preparation, and training the designed system. Then, the trained character classifier, which is a deep convolutional neural network (Deep CNN) is used to recognize the base characters and some of the punctuation marks of Geez. Essentially, the convolutional neural network is not responsible for character classification, its main purpose is to extract the features of the characters from the image. Therefore, we will have a dense layer which is responsible for accepting the features extracted from the convolution layers and use them to predict the characters. The dense layer uses the extracted features for predicting the true classes.

The proposed system will accept a colored image of a document which is captured either through digital cameras or scanner. Then the system converts the colored image into a grayscale version of the image which will make the preprocessing process much easier. Since colors don't have an effect in character classification, we can convert the RGB image into grayscale. Also, by converting the colored images into the grayscale image we will be advantages on reducing the computational power required to compute the required operations. Colored or RGB images are stored as three-dimensional vectors which means for each color, we need one-dimension. However, grayscale images are stored as a one-dimensional image which reduces the space and other computational powers required three times.

Afterward, the grayscale image is preprocessed for the purpose of eliminating noises which are unwanted pixels or dots on the image. Noise will highly affect the feature extraction and classification process. Although, since the system is intended to deal with an ancient and degraded documents noise reduction is a critical process. Furthermore, at this stage, the characters should be separated from the background of the document this process is binarization. In which the grayscale image is converted into a binary format that can represent the foreground and background of the image more clearly. As we know it, the word binary means two, so the binarization process will convert each pixel of the image either into black or white. The conversion is done by using a threshold value *t* in which every pixel point having value more than that

threshold value will be converted into white and the rest is converted into a black color. Therefore, in the binarized image, we only have an image which is composed of two colors, a completely white (255) and black (0). Usually, the foreground of the image is represented using black color and the foreground or the characters are repressed by white color.

Then the preprocessing stage is followed by a segmentation process which involves extracting every lines, words, and character of the input document. The main aim of this phase is to extract the character from the image with the original order so that the features can be extracted from them. Then the extracted characters are passed to the convolutional layers so that the relevant features that can uniquely represent the character will be extracted from them. The input for the first convolutional layer is a two-dimensional vector form of the extracted characters. Then, the convolutional layer uses these input vectors and perform a convolution operation with the filter kernel. Through the repetitive process of convolving and pooling the features of the characters are extracted. Then the dense layer of the network uses these features to classify the characters.

5.2. System Modeling

As we have tried to explain in the previous chapters many character recognition systems might follow different steps and processes. However, there exist some processes that are common and should be common. The following Figure 5.1 shows the common steps and process that the proposed system follows. The proposed recognition system doesn't include any postprocessing phase hence, it's out of the scope of the thesis. The model of the proposed system encompasses the most important and recommended process of handwritten character recognition system starting from data acquisition to the classification stage.

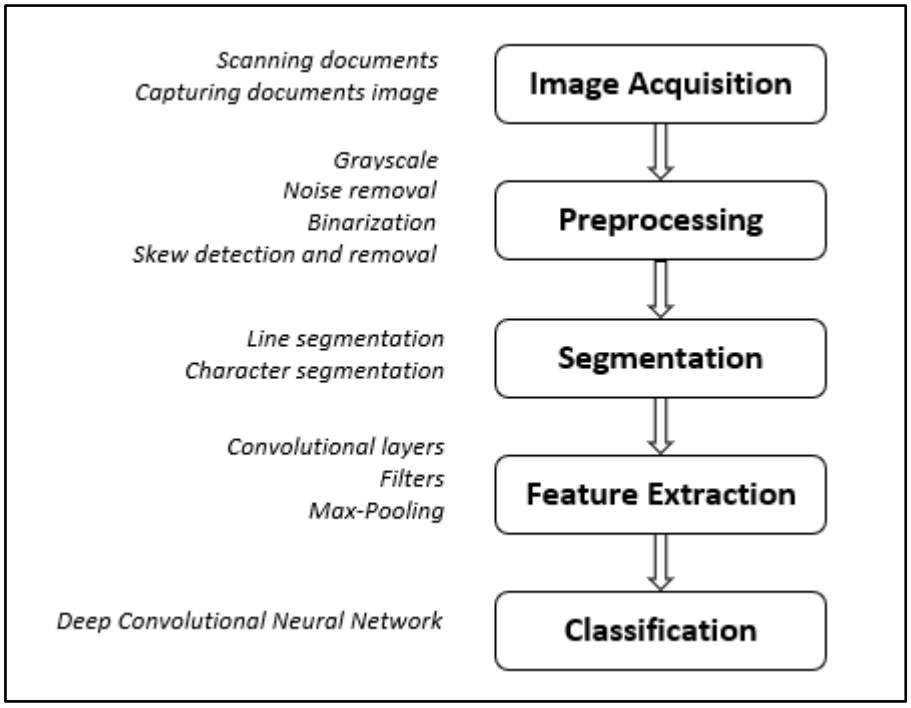


Figure 5.1: Model of the proposed system

5.3. Image Acquisition

The majority of the images are collected from EOTCs and libraries. Most of the images are scanned using a scanner. We have collected around 200 pages of images from all sources. More than 85% of the collected images are gathered from documents which are found in EOTCs and the rest are collected from libraries, private books, and the internet. We have gathered 160 pages of 15 different books from ancient documents found in EOTCs. Also, 40 pages of images are captured from books found in libraries and private. But most of the books found in the libraries are not that much aged and degraded documents. This might affect the recognition system.

Hence, the proposed system is mainly intended ancient and degraded document recognition we need to collect aged and degrade document. Eventually, the documents found in EOTCs are suitable for the proposed system but the permission to access them is limited. Also, there are pages which are collected from the internet. Even though more than 160 pages are collected from EOTCs and libraries, the repetition of the base characters on those documents is not as much as we need. In total around 208 pages are collected for creating the dataset.

5.4. Preprocessing

Once the images are collected from the different places which are discussed above, they should pass through some preprocessing techniques. The aim of this stage is to eliminate unwanted pixels

so that a better form of the images can be obtained. Usually, the preprocessing stage involves different subprocesses. The preprocessing stage of the proposed system involves the following step by step processes. Note that every image is scaled down to 300dpi * 600dpi.

5.4.1. Grayscale conversion

The images collected are colored or RGB images after they are scaled down, they are converted into grayscale images using the following Equation 5.2 for each and individual pixels of the original image. Which works by multiplying the red, green and blue values of the colored image with some predefined constants and summing them. The final sum of the multiplied values of each colored is a single value which represents how much a single pixel is bright. In order to find the brightness of each pixel in the inputted image, we can use Equation 5.2 whereas *i* and *j* represent the column and row of the image vector respectively.

$$gray = 0.299 * R + 0.587 * G + 0.114 * B \quad (5.1)$$

$$gray_{(i,j)} = 0.299 * R_{(i,j)} + 0.587 * G_{(i,j)} + 0.114 * B_{(i,j)} \quad (5.2)$$

5.4.2. Noise reduction

Noise reduction is an essential process while dealing with handwritten document recognition especially while conducting ancient document recognition since the documents are highly noise and degrade. In the proposed system we have used a noise reduction algorithm called non-local means denoising which was originally proposed in a research paper by (Buades et al., 2011). It works by finding the average of similar pixels and updating the pixel value with the calculated average pixel value. The whole image is scanned for finding pixels having similar intensity with the pixel to be denoised. Denoising is basically done by computing this average value of the pixels having similar values. The similarity is obtained by Equation 5.3.

$$NLu(p) = \frac{1}{C(p)} \int f(d(B(p), B(q))u(q)dq) \quad (5.3)$$

Where $d(B(p), B(q))$ is an Euclidean distance between p and q in which they are patches of the image. Also, $C(p)$ and f are the normalization and decreasing functions respectively. Furthermore, the denoising of a pixel p on image u is given by the following Equation 5.4. Where $B(p, r)$ is representing neighborhood centered at p and a pixel size of $(2r + 1) \times (2r + 1)$. The results of this noise reduction algorithm is given in the next chapter.

$$u(p) = \frac{1}{C(p)} \sum_{q \in B(p,r)} u(q)w(p,q), \quad C(p) = \sum_{q \in B(p,r)} w(p,q), \tag{5.4}$$

There is no common and standardized noise reduction algorithm that can be suitable for any process. As a recommendation, if we can let the user determine the amount of noise on the run time, we can easily apply a well-suited noise reduction technique based on the degree of noise.

5.4.3. Binarization

Once the images are converted to grayscale and the irrelevant noises are removed from the images then the binarization phase takes place. In which it converted the grayscale image into binary images so that the image is composed of pixels having a color value of either black or white. In the proposed system we have used inverted Otsu thresholding algorithm to convert the images into binary format. The inverted method converts the background with black color and foreground or characters with white color. The Otsu method is intended to find a threshold value that can separate the foreground of the image with background ie. the black pixels with white pixels. So that it can minimize the overlap that occurs between the white and black pixels. The Otsu algorithm uses the threshold value and increase the distribution of either of the pixels and decrease the other left pixel. Using Equation 5.3 we can find the within class variance $\delta_{wc}^2(T)$.

$$\delta_{wc}^2(T) = n_B(T)\delta_B^2(T) + n_O(T)\delta_O^2(T) \tag{5.5}$$

In which $\delta_B^2(T)$ is the variance pixels having an intensity value of less than the threshold value or in easier terms variance of the background. Also, $\delta_O^2(T)$ is the variance of the foreground pixels or pixels having an intensity value greater than the threshold. And we can find $n_B(T)$ and $n_O(T)$ in which the range of intensity is given by *{0, 1, ..., N-1}* based on the Equations given at 5.4 and 5.5 respectively.

$$n_B(T) = \sum_{i=0}^{T-1} p(i) \tag{5.6}$$

$$n_O(T) = \sum_{i=T}^{N-1} p(i) \tag{5.7}$$

And by using the following equations the algorithm finds the between-class variance. It can be easily calculated by subtracting the within class variance from the total class variance.

$$\delta_{bt}^2(T) = \delta^2 - \delta_{wc}^2(T) \tag{5.8}$$

$$\delta_{bt}^2(T) = n_B(T)[\mu_B(T) - \mu]^2 + n_O(T)[\mu_O(T) - \mu]^2 \quad (5.9)$$

$$\delta_{bt}^2(T) = n_B(T)n_O(T)[\mu_B(T) - \mu_O(T)]^2 \quad (5.10)$$

Here $\mu_B(T)$ and $\mu_O(T)$ are not defined yet, they are computed using a simple recurrence operation the algorithms compute them using Equation 5.9 and 5.10 respectively.

$$\mu_B(T+1) = \frac{\mu_B(T)n_B(T) + {n_T}^T}{n_B(T+1)} \quad (5.11)$$

$$\mu_O(T+1) = \frac{\mu_O(T)n_O(T) + {n_T}^T}{n_O(T+1)} \quad (5.12)$$

Whereas $n_B(T+1)$ and $n_O(T+1)$ are as follows

$$n_B(T+1) = n_B(T) + n_T \quad (5.13)$$

$$n_O(T+1) = n_O(T) + n_T \quad (5.14)$$

Therefore, we can see how Otsu thresholding works. Simply it calculates the thresholding value and separates the entire image pixels in two groups and then it finds the mean of each group. Then it squares the result of the subtraction of the mean of the two groups and finally multiplies the number of pixels in the two groups. The result we have after applying the inverted Otsu thresholding algorithm is in the opposite format. In which the pixels of the character is represented with white color and the background is represented with black color. As a recommendation, there is no efficient thresholding algorithm that can be suitable for every problem. Therefore, it's recommended to include more than one thresholding procedure and leave the choice for the system users to apply the efficient algorithm based on their choice at the run time.

5.4.4. Skew detection and correction

Once the image is denoised and binarized the skew detection and correction method is applied to the binary image. Initially, the algorithms find all the pixel coordinates, that are part of the foreground. The foreground pixels are all the points having a pixel color of white. Then the rotation of the rectangle is calculated based on the collected pixel coordinates. The rotation angle is which will be returned is between -90 and 0. By using the angle we will find the rotation matrix with respect to the center coordinates of the image through Equation 5.15. Finally, using the rotation

matric obtained we rotate the inputted binary image using the following Equation 5.18. Where *src* is the threshold image and *img* is the rotated image.

$$M = \begin{bmatrix} \alpha & \beta & (1-\alpha) * center.x - \beta.center.y \\ -\beta & \alpha & \beta.center.x + (1-\alpha).center.y \end{bmatrix} \quad (5.15)$$

Where α and β are as described below which r is the rotation angle.

$$\alpha = \cos r \quad (5.16)$$

$$\beta = \sin r \quad (5.17)$$

$$img(x,y) = src(M_{11x}, +M_{12y} + M_{13}, M_{21x+M_{22y}} + M_{23} \quad (5.18)$$

5.4.5. Morphological transformations

Furthermore, we have also used some other transformations that are used to remove the irrelevant pixels that happen during the task of binarization and skew correction. The first transformation operation applied is opening that will remove the boundaries of the foreground or the character so that the character becomes more visible. Secondly, we have applied a transformation operation called dilation. The dilation technique adds more white pixels to the foreground of the image. This dilation transformation is useful to improve the characters that are too thin by adding pixels on the boundaries of the character. Then finally closing transformation is applied, which will help in filling the openings that happen inside the foreground of the images. All of these morphological operations provide us with the ability to reduce noises that are missed in the noise removal stage and noises that happen during the execution of the other stages. Thus, through thinning and filling some gaps that occur between characters we have improved the preprocessing phase basically the noise reducing stage. However, these operations have drawbacks, they might end up with resulting additional noises.

5.5. Segmentation

Segmentation is a critical step in the handwritten recognition process and it highly affects the recognition process. Usually, the segmentation process involves three steps which are line segmentation, word segmentation, and character segmentation. For each segmentation process, contour analysis plays a greater role in the proposed system. Basically, for line segmentation and character segmentation, an especially technique of adding additional foreground colors horizontally is used. In general filling additional pixel which is also called dilation, the technique is used for line segmentation. However, the word segmentation is not conducted in the

segmentation process because of the nature of Geez documents. The methods used in the proposed system is described in the following subsections.

5.5.1. Line segmentation

Text line segmentation is an essential step in which every line is cropped out of the image and stored in a sorted manner. Thus, using those lines words and characters can be extracted from each and every line with the original order. We have used contour locating algorithm for finding contours in both line and character segmentation processes which is described in Section 5.5.4. The steps used in the proposed system for line segmentation is as follows:

1. Accept the binarized image that was processed with the preprocessing techniques.
2. Copy the binarized image because the image will be updated while finding lines.
3. Find all the character pixels or white pixels from the copied image i.e. after binarization white color is the representation of characters and black is for the background.
4. Add an additional 50 pixels to each pixel horizontal so we could connect each character horizontally i.e. 50 pixels is the maximum gap between characters that need to be filled
5. Now we have the original threshold image and a copied image that has the lines as a horizontal connected character that forms a line.
6. Then we have applied a contour-finding technique to find out the text lines.
7. Contour searching algorithm returns coordinates of each line.
8. Then we cropped out the lines from the images using the coordinates.

The contour analysis algorithm finds the borders of each line and returns a list of coordinates for the contours. The coordinates are the outer most pixels of the lines from the image. A summary of the steps used by the algorithm for finding contours is discussed in Section 5.5.4.

5.5.2. Word segmentation

In the proposed system the way we have achieved word segmentation is a little bit tricky. In Geez, literary words are not separated with spaces, rather with a "፡" punctuation mark. The word separator has a colon like structure, so each word is separated with a colon from another word. Since the system can recognize some of the punctuation marks the word separator will be recognized in the classification stage. One of the aims of recognizing some of the punctuation marks was for this kind of reasons. Thus, there is no need for word segmentation.

5.5.3. Character segmentation

Once the lines are segmented successfully character segmentation is carried out. In the proposed system for the character segmentation, we have used a similar technique with the line segmentation method described in section 5.5.1. In order to segment or crop the character, we have used contour analysis. The steps followed for character segmentation is described below

1. Accept the list of lines which are returned from the line segmentation stage.
2. Find all contours/characters on the line and return a list of coordinates for each contour.
3. Crop the characters from the threshold text line images using the returned coordinates.
4. Iterate through each line.

5.5.4. Finding contour

Basically, the contour analysis algorithm we used was proposed in a research paper (Suzuki and Abe, 1985). In which the algorithm uses border tracing technique to find all contours or objects on the image. The algorithm finds the borders of each contour and returns their location coordinates. The coordinates returned are the points of the top-left, top-right, bottom-left, and bottom-right corners of each contour. Then the coordinates are used to process the contours or objects. The steps that the algorithm follows for finding contours are listed below.

1. Scan the binary image while updating P. Where P is a counted sequential number of the lastly found outer most border.
2. Represent each hole by shrinking the pixels into a single pixel based on a given threshold value t. (where t is a minimum perimeter of a character or line in line segmentation).
3. Represent the outer border by shrinking its pixels into a single pixel based on a given perimeter or threshold value t.
4. Place the shrank outer border pixel to the right of the shrank hole pixel.
5. Extract the surrounding relations connected components i.e. holes and outer borders.

5.6. Feature Extraction and Classification

Once the segmentation process is completed the system has all of the isolated characters and it's ready to extract the features that will uniquely represent the characters. Convolutional Neural Networks are composed of two parts. The first part contains a number of convolutional layers which are a special kind of neural network that is responsible for extracting features from the inputted character image. And the second part includes dense neural layers which are required to classify the inputted character based on the features extracted from the convolution layers. On the proposed system we have used 3 convolutional layers for feature extraction and 2 dense layers for

classification purpose. The architecture of the proposed system is shown in Figure 5.2. The main goal of any neural network is to decrease the loss function which can be given either by cross-entropy shown in Equation 5.19 or means squared error shown in Equation 4.21.

$$\mathcal{L} = -\sum_{i=1}^{N} Z_{x,c} \log(P_{o,c}) \tag{5.19}$$

Whereas N is the total number of classes, Z is an indicator which is either 0 or 1 if the label c is a right classification for input x, and P is the predict probably for input x.

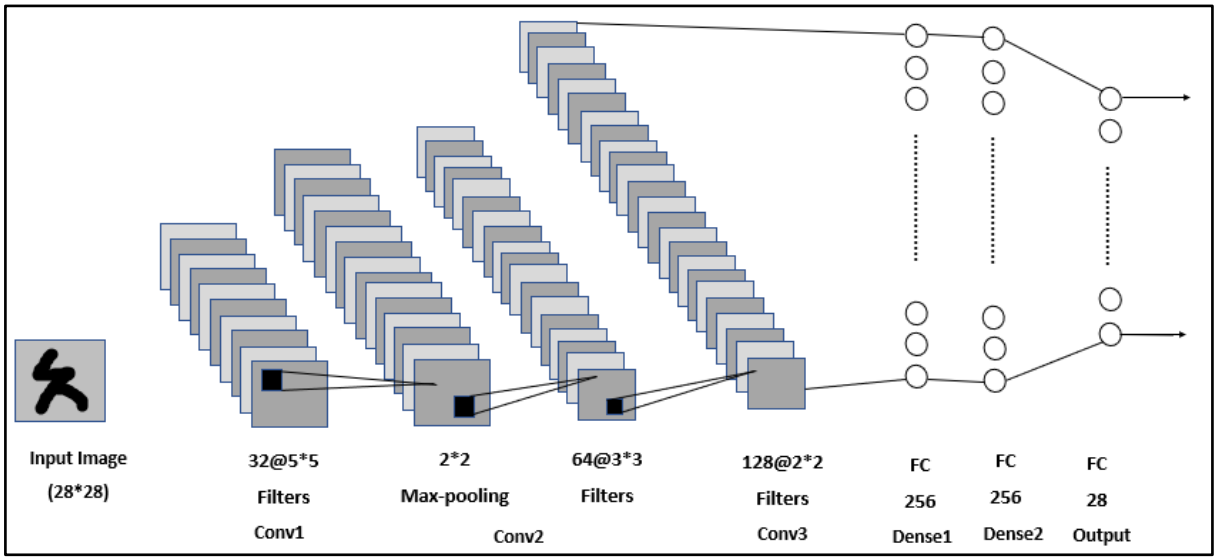


Figure 5.2: Convolutional neural network architecture of the proposed system.

The designed convolutional neural network architecture has a number of convolutional layers, filter kernels, activation function, fully connected layers, and pooling specifications as specified in the following Table 5.1.

Table 5.1: Convolutional layer specifications of the system.

Layers	Input shape	Filters	Filter size	Pooling	P-size	Dropout	Activation
Conv1	28*28	32	5*5	Max-pooling	2*2	0.25	ReLU
Conv2		64	3*3	Non	Non	0.2	ReLU
Conv3		128	2*2	Non	Non	Non	ReLU

The output of the second convolutional network is the features and they are flattened. Thus, they can be passed to the fully connected or dense layers. In the following Table 5.2, we can see the specifications of the dense layers that are responsible for using and classifying the features extracted from the convolutional layers.

Table 5.2: Deep layer specifications of the system.

Layers	Dropouts	Input shape	Activation function
Dense1 layer	0.2	256	ReLU
Dense2 layer	0.2	256	ReLU
Output layer	Non	28 (number of classes)	SoftMax

5.6.1. Training the designed architecture

The neural network architecture of the proposed system is trained using a supervised learning algorithm. The network is trained with three different epoch values and the default batch size which is 128 with the optimizer function of Adam. Adam is the best gradient descent optimization technique for optimizing deep neural network architectures. We have decreased the batch size even if higher batch size is better in decreasing the time required to train the model. However, using a smaller batch size decreases the computational power required to train the neural network.

5.7. Preparation of Dataset

For training and testing the proposed system, we have preparade our own dataset that contains the base characters and some of the punctuation marks of Geez alphabets. The dataset is prepared from a total of 208 pages collected from EOTCs, libraries, and private books. The dataset contains 22,913 binary image characters which are normalized into 28*28 sized images. It was really difficult to find the same frequency of character images from the images, thus, each of the characters doesn't have an equal frequency of repetition. The frequency varies from character to character the minimum number of images per class is around 400 and the maximum is 1700. Finally, the prepared dataset is separated into three parts for training, testing, and validation sets. In which the dataset used for training contains 70% (16,038) of the entire images and 20% (4,583) is used for testing, then the reset 10% (2,292) is used for validating the proposed system. Furthermore, the results for the process of training and testing the proposed system is discussed in the next chapter.

5.8. Prototype Implementation

The prototype of the proposed system is designed using different supportive libraries on PyCharm, which is a python editing tool. Among those supportive libraries Keras, OpenCV, and TensorFlow are the most import libraries used. Almost all techniques used for preprocessing of the collected image in order to remove noise, skew correction, image reading, and other relevant task are done using functions which are provided by OpenCV. We have used Keras API for developing the neural network architecture and training. Also, Keras has its own graphical representation of the processes of training the neural network. The specifications of the programming tools, supportive libraries, and execution environment are discussed below.

Programming language: Python 3.6 used for developing the proposed system.

Libraries: Some of the relevant and commonly used libraries used during implementation of the proposed system are listed below.

1. **OpenCV:** For preprocessing and other relevant tasks.
2. **Keras:** For implanting the CNN architecture in a much easier way.
3. **TensorFlow:** For normalizing image vectors and as a supporting library for Keras.
4. **NumPy:** For converting the images into integers and storing them as a 2D vector.

Execution environment: The specifications of the execution environment used while implementing, training and testing the proposed system are shown in the following table.

Table 5.3: Execution environment specifications

No.	Type	Specification
1	Processor	Intel (R) Core (TM) i3-6100U CPU @ 3.30GHz
2	RAM	4 GB DDR4
3	Storage	1TB HDD
4	Architecture	64bit
5	Operating System	Windows 10 professional

CHAPTER 6
EXECUTION RESULTS OF THE PROPOSED SYSTEM

6.1. Results of Preprocessing

The results of each of the preprocessing stage of the proposed system are described in the following subsections.

6.1.1. The result of the grayscale conversion

The first step for the preprocessing stage in the proposed system is the conversion of the colored (RGB) image into a grayscale image. We used OpenCV method named cv2.cvtColor with a specific constant parameter cv2.COLOR_BGR2GRAY to converted the colored image into a grayscale image. The result of the grayscale conversion method is shown in Figure 6.1.

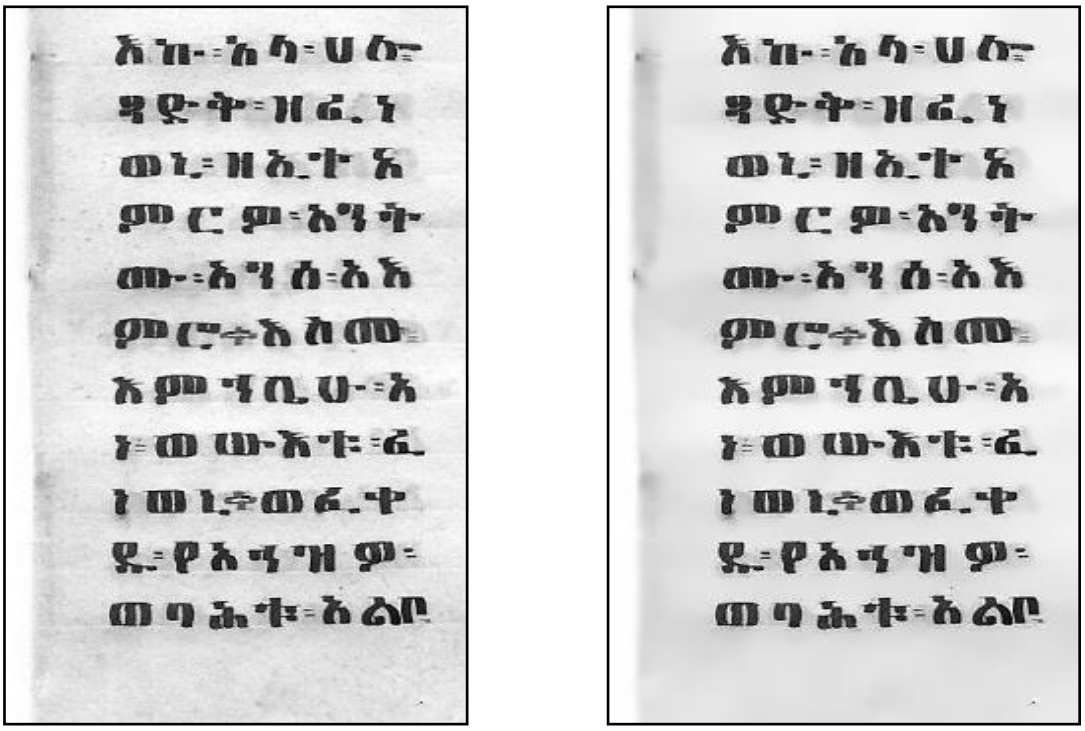
(a) Input colored image (b) Grayscale image

Figure 6.1: Results of grayscale conversion

6.1.2. The result of noise reduction

The next step after grayscale conversion is noise reduction in our proposed system. The noise reduction process accepts the grayscale image from the grayscale conversion and reduces the noise from the image. The OpenCV method named cv2.fastNlMeansDenoising is used for

reducing the noises from the grayscale images. The result of the noise reduction operation for a noisy and non-noisy image is shown in Figure 6.2.

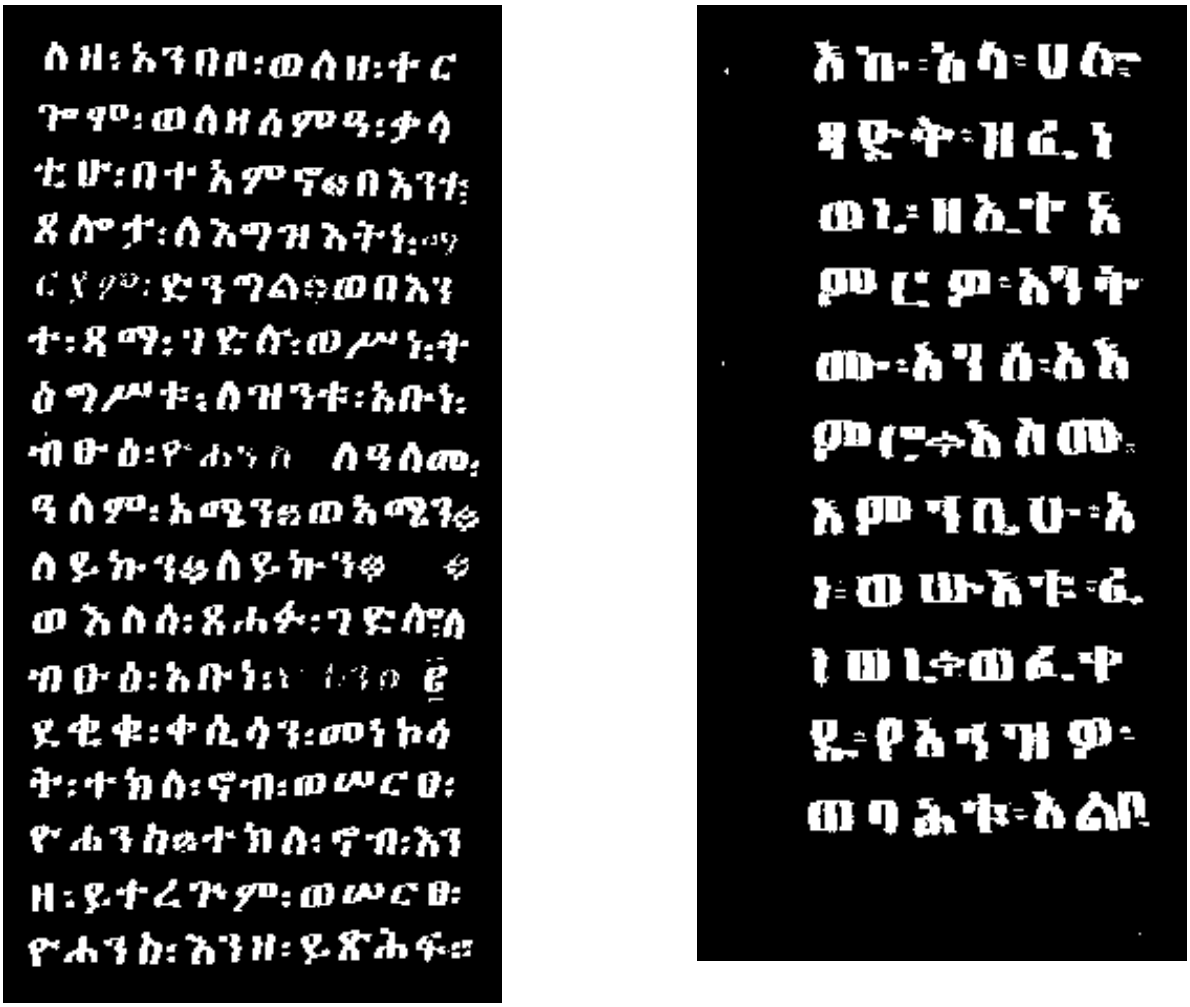


(a) Input grayscale image (b) Denoised image

Figure 6.2: Results of noise reduction

6.1.3. Result of binarization

Once the grayscale conversion and noise reduction stages are done the image is passed to the binarization step so that by using Otsu binarization technique the image is binarized. By using OpenCV method named cv2.threshold with parameters specifying the binarized image should be inverted and Otsu binarization technique should be used. The results of the Otsu binarization method is shown in the following Figure 6.3.



(a) Binarized image of Figure 6.1

(b) Binarized image of Figure 6.2

Figure 6.3: Results of Otsu binarization conversion

6.1.4. The result of the skew correction

The skew detection and correction process are undertaken after the image is binarized from the grayscale image. We have used cv2. getRotationMatrix2D method for getting the rotation matrix based on the calculated degree and cv2.warpAffine is used to rotate the matrix using the rotation matrix returned. The result of the skew correction is shown in Figure 6.4 below.

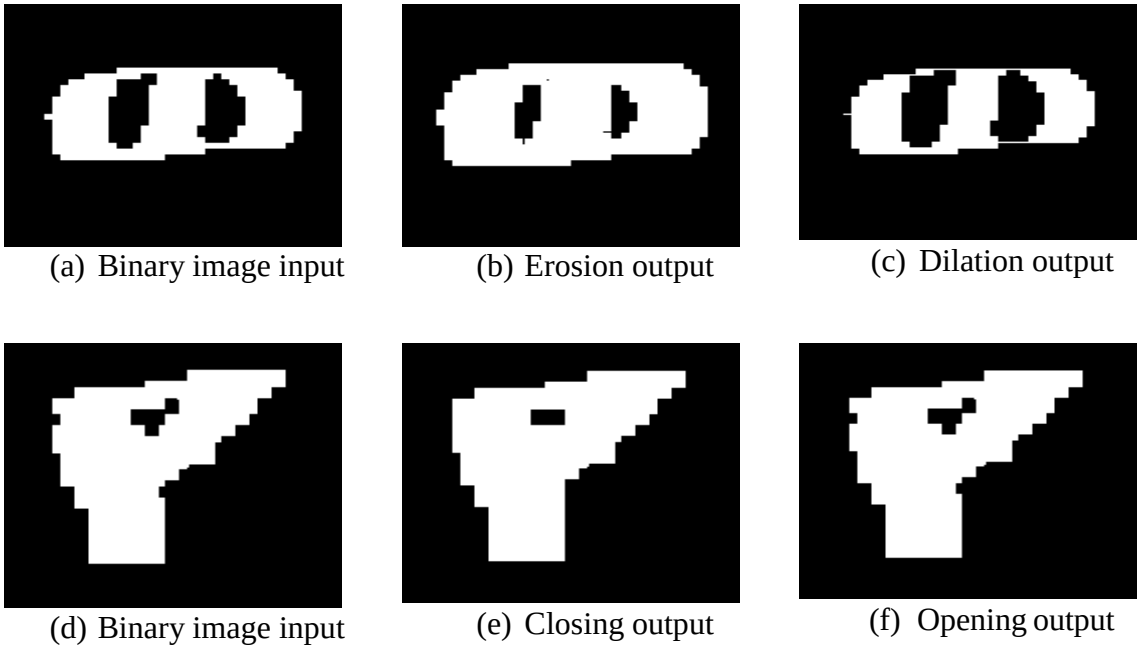
(a) Skewed image

(a) Skew corrected image

Figure 6.4: Results of skew correction

6.1.5. The result of morphological transformations

These morphological operations are the last step of the preprocessing phase of the proposed system. We have performed four kinds of morphological operation on the most recent image. These operations are opening and closing using the methods provided by the OpenCV API. Whereas opening is done through apply erosion first and the dilation next also, the reverse is for closing first dilation then erosion. Some of the results of these morphological operations are shown in bellows Figure 6.5. The samples shown below are some of the Geez base characters and they are zoomed to make them visible so that the effects of the morphological transformation methods are clear and distinguishable.



(a) Binary image input (b) Erosion output (c) Dilation output

(d) Binary image input (e) Closing output (f) Opening output

Figure 6.5: Results of the morphological transformations.

We have applied an efficient preprocessing technique for each stage and we have obtained a good classification accuracy. But, in some conditions when there is a higher degree of noise on the documents some of the preprocessing techniques have failed. Some of the errors that have happened during the preprocessing phase of the proposed stem can be found in Appendix A.

6.2. Results of Segmentation

The proposed system involves three steps of segmentation which are line, word and character segmentation. However, word segmentation is not performed on the images rather, the final out of the character classifier is grouped into words. Therefore, we can say in the proposed system word segmentation is performed implicitly after the recognition process is completed.

6.2.1. The result of line segmentation

For lines segmentation, the algorithm we have applied accepts the binary images that have passed through all the preprocessing stages. The algorithm has been tested by a number of documents images and its accuracy was as expected. However, in some cases when there is a connection among the lines the algorithm was unable to segment the lines accurately. Basically, the connection that exists among the lines was supposed to be removed on the noise removal and morphological transformation stages. Some errors of line segmentation can be found in Appendix A. Figure 6.2 shows the sample result of the line segmentation stage.

(a) Binary image input (b) The result of line segmentation

Figure 6.6: Results of line segmentation

6.2.2. The result of character segmentation

Once the lines are extracted successfully through iterating each extracted line segments the algorithm locates the contours or characters on each line. The proposed system has been tested with a number of documents and it has successfully segmented most of the characters as shown in Figure 6.7. However, in some cases, the results were not as expected or as they should be. On some documents, the character segmentation algorithm has detected more than two characters as a single character. Also, some character has not been recognized as a character and they are jumped see Appendix A. Some of the reasons for these problems are listed below:

1. There exists a connection between the characters i.e. there is no space b/w the characters.
2. The erosion or thinning method we used was unable to separate the connected characters.
3. Some of the punctuation marks are too small naturally thus, they can't meet the minimum requirement of the contour finding algorithm i.e. minimum contour area.

(a) List of binary line images (b) Character segmentation results

Figure 6.7: Character segmentation results

6.3. Results of Training and Classification

The prepared dataset containing 22,913 binary images which are centered and normalized to a size of 28*28 is used to train and test the prepared convolutional neural network. Out of the total dataset images, 16,038 is used for training, 4,583 is used for testing and the left 2,292 is used for validating the designed neural network architecture. The result of the training and testing of the proposed convolutional neural network architecture is presented in the following sections. The proposed neural network system is trained with the prepared dataset according to the specifications presented in Table 6.1. Which shows us that when the epoch value increases and also the loss value decreases. Also, the frequencies of each class are displayed in Table.

Table 6.1: Training and classification results of the proposed CNN architecture

Batch size	Epoch	Test accuracy	Test loss	Validation accuracy	Validation loss
128	10	0.992363	0.04781	0.9895	0.0756
128	15	0.993890	0.04442	0.9895	0.0773
128	20	0.991708	0.05239	0.9887	0.0508

As the results shown in the above table when the epoch value increases the accuracy of the system is improved, however when the epoch value is raised to 20 the model is not showing any improvement. We can see the loss and accuracy of the model for the three different epoch values from the graphs. The first two Figures 6.8 and 6.9 shows the accuracy and loss graphs of the training and classification result trained using 10 epoch values respectively. And Figure 6.10 and

6.11 shows the accuracy and loss graphs of the model which is trained using 15 epoch value respectively. Finally, Figure 6.12 and 6.13 shows the accuracy and loss graphs of the CNN model which is trained with 20 epoch values respectively. Furthermore, the confusion matrix for the CNN model trained with 15 epoch value is shown in Figure 6.14. The confusion matrix will help us to easily identify which classes are miss-classified and how many times. As we can see it from Figure 6.14 the character "ቀ" is 4 times miss-classified as "ተ", "ጸ" is 3 times miss-classified as "ደ", "ሰ" is 2 times miss-classified as "ለ", "ፀ" is miss-classified 2 times as "ዐ", and "ዐ" is miss-classified 2 times as "ፀ". These are the maximum misclassified classes from the training set. The total number of misclassified classes out of the total 4,583 testing images is 28.

Table 6.2: Image frequencies of the dataset for each character.

Characters	**Frequency**	**Characters**	**Frequency**	**Characters**	**Frequency**
ሐ	637	ፈ	458	ሠ	632
ሰ	704	ገ	667	ተ	738
ጎ	575	ሀ	1091	ወ	899
ዐ	1396	ከ	668	የ	613
ጠ	619	ለ	872	ዘ	1356
ጸ	569	መ	831	፡	1692
ፀ	1108	ነ	670	፤	224
አ	1613	ጰ	556	።	846
በ	1005	ቀ	619	**Total =**	22,913
ደ	623	ረ	632	**Training =**	16,038 (70%)
				Testing =	4,583 (20%)
				Validations =	2,292 (10%)

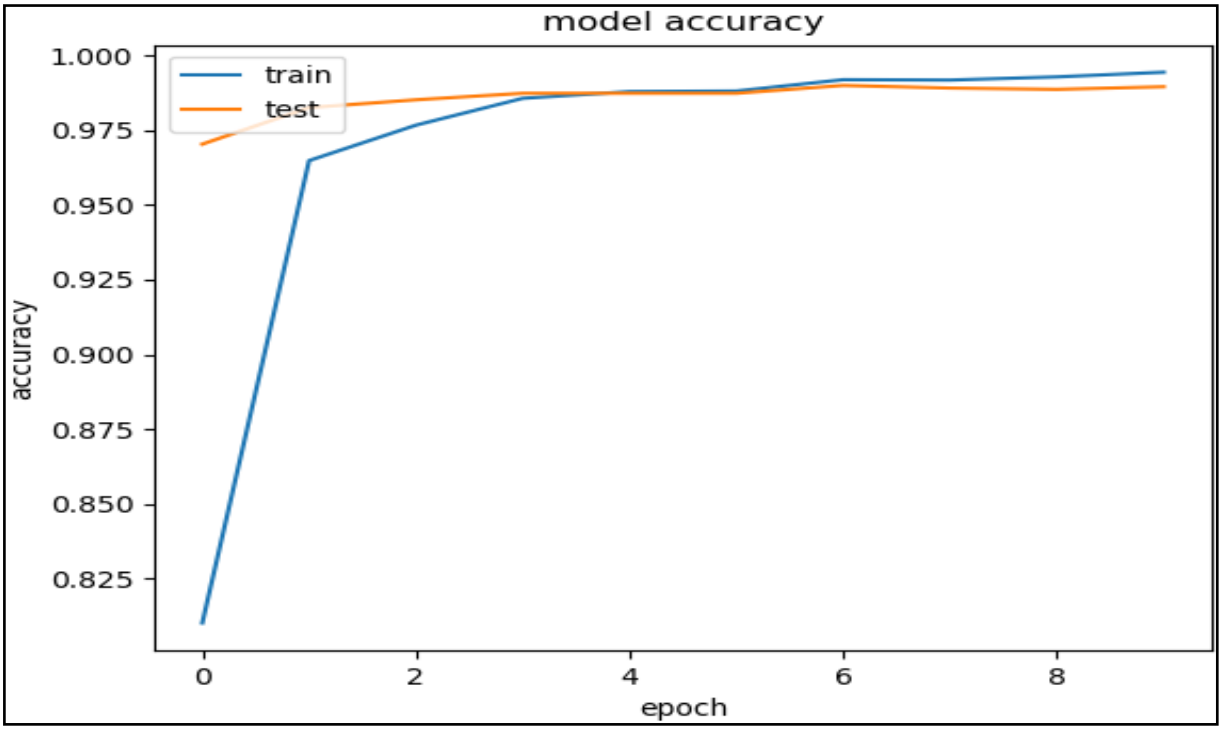


Figure 6.8: Accuracy of the proposed CNN model with epoch value of 10.

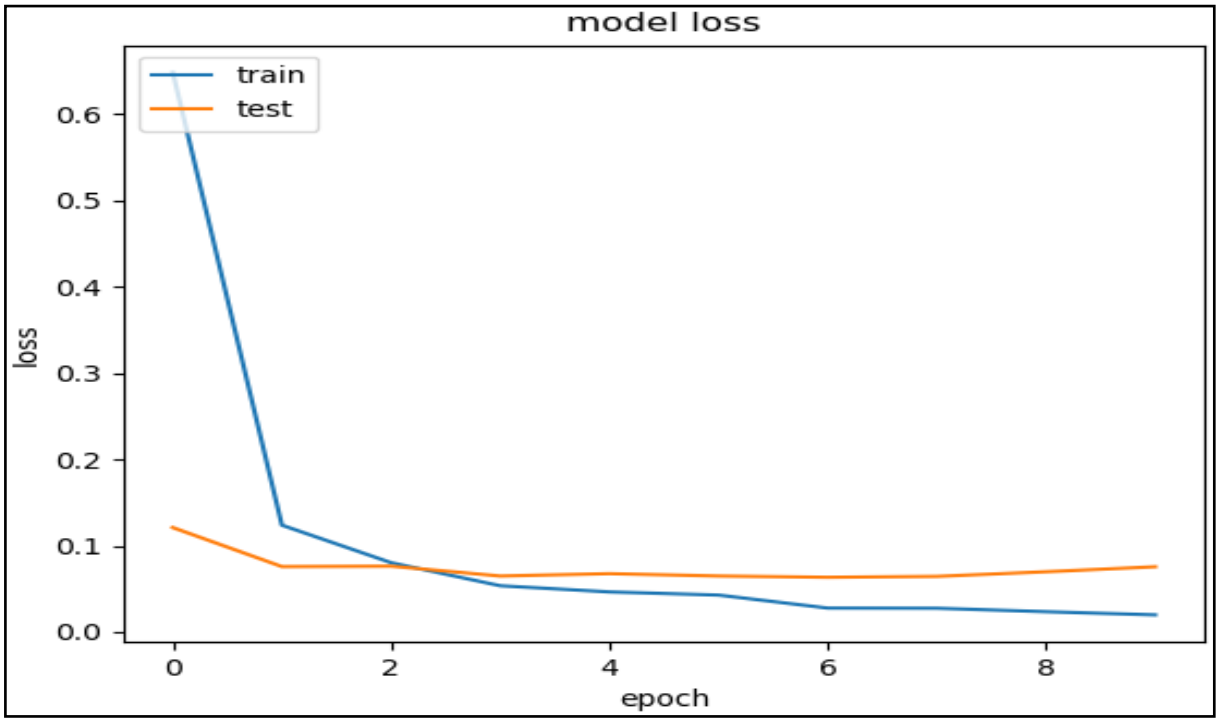


Figure 6.9: Loss of the proposed CNN model with epoch value of 10.

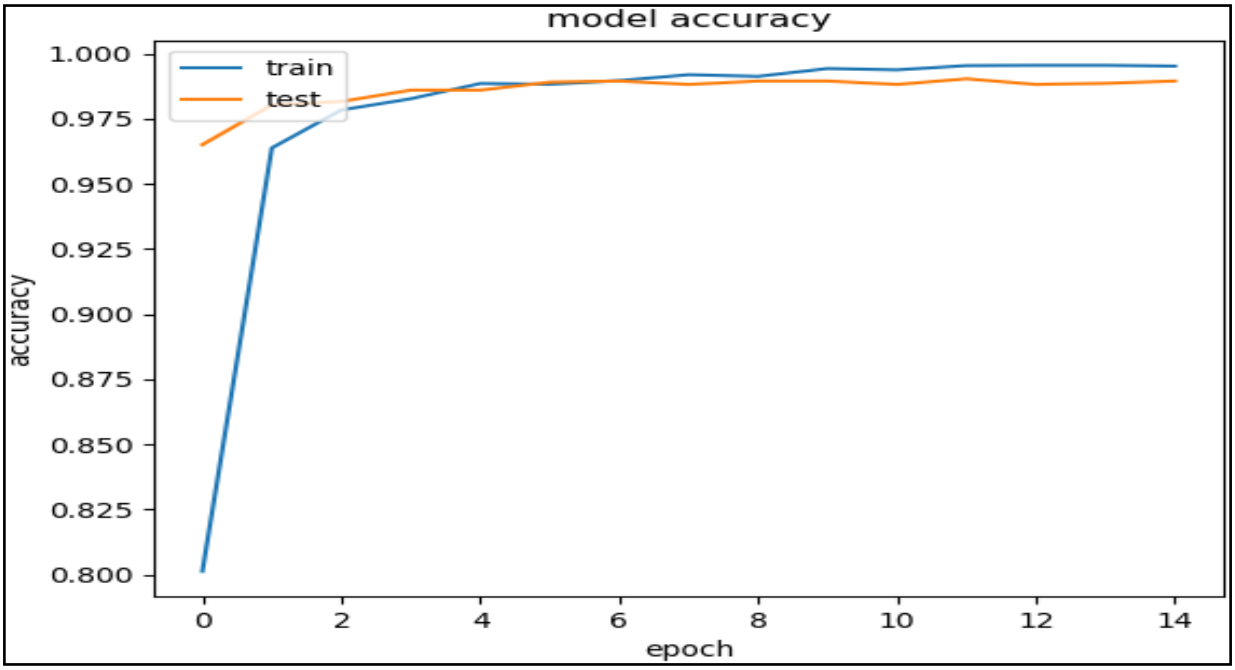


Figure 6.10: Accuracy of the proposed CNN model with epoch value of 15.

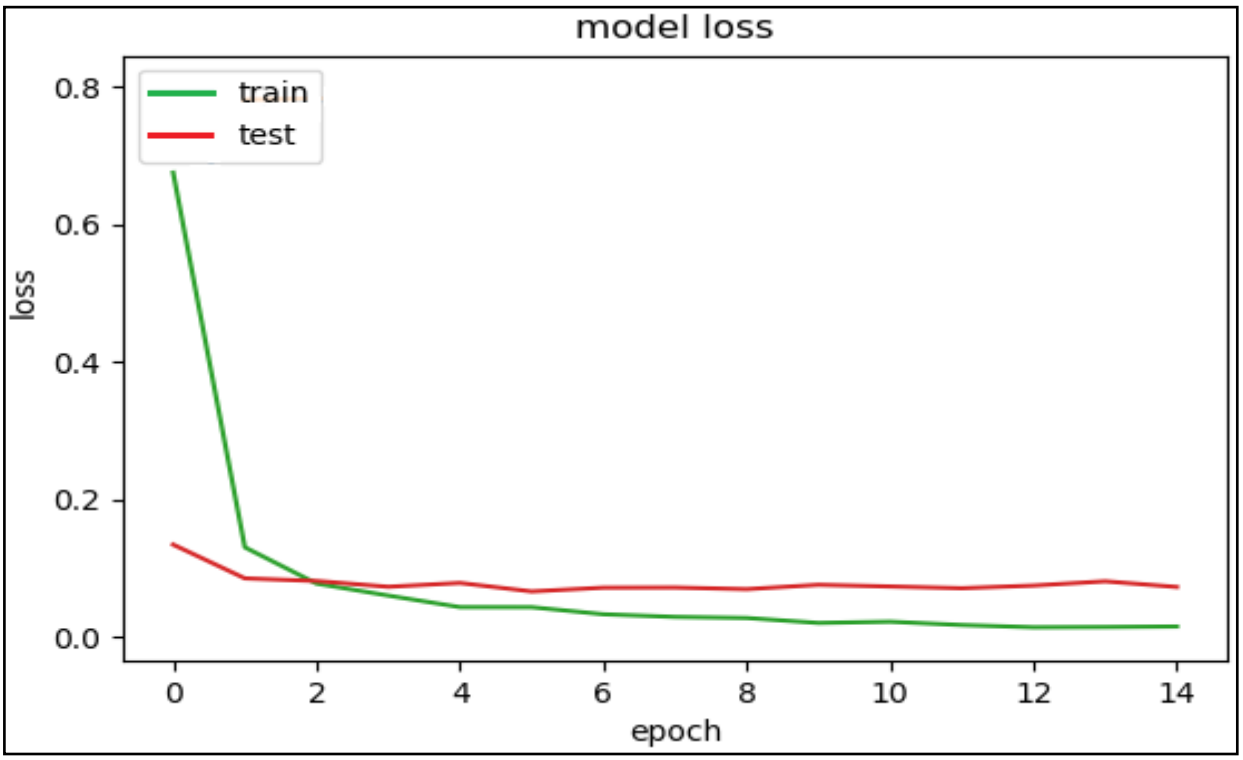


Figure 6.11: Loss of the proposed CNN model with epoch value of 15.

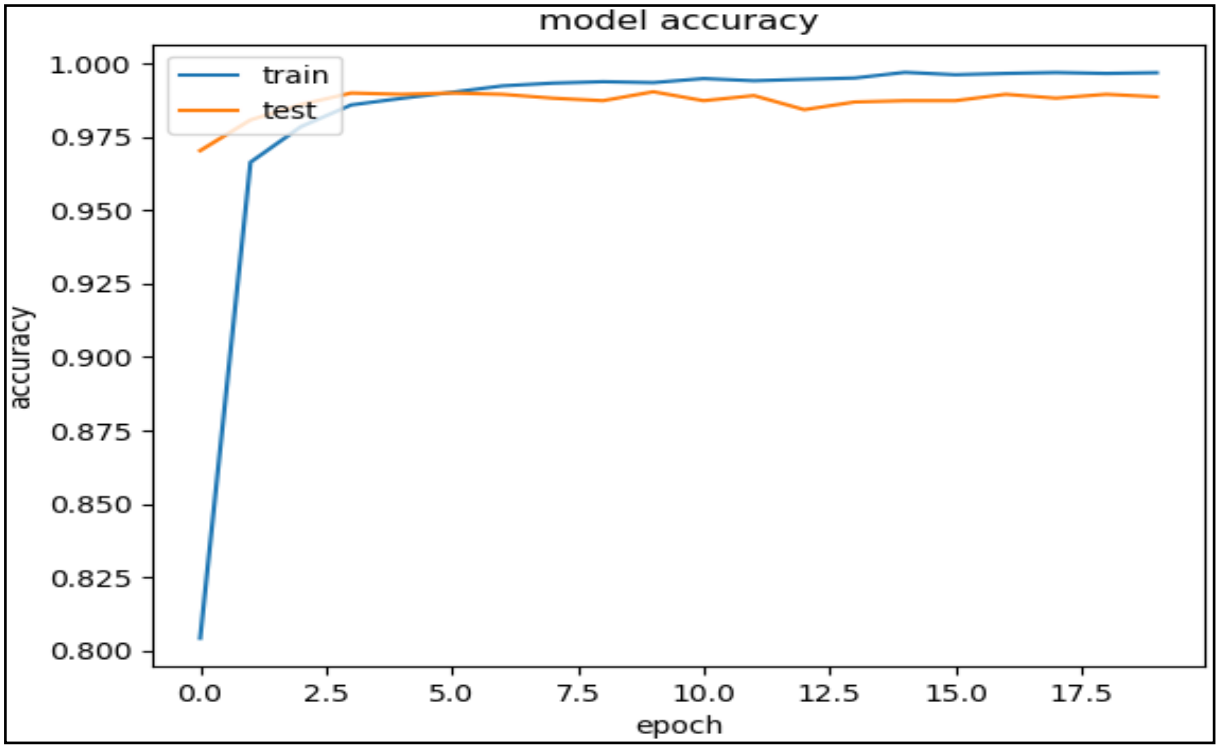


Figure 6.12: Accuracy of the proposed CNN model with epoch value of 20.

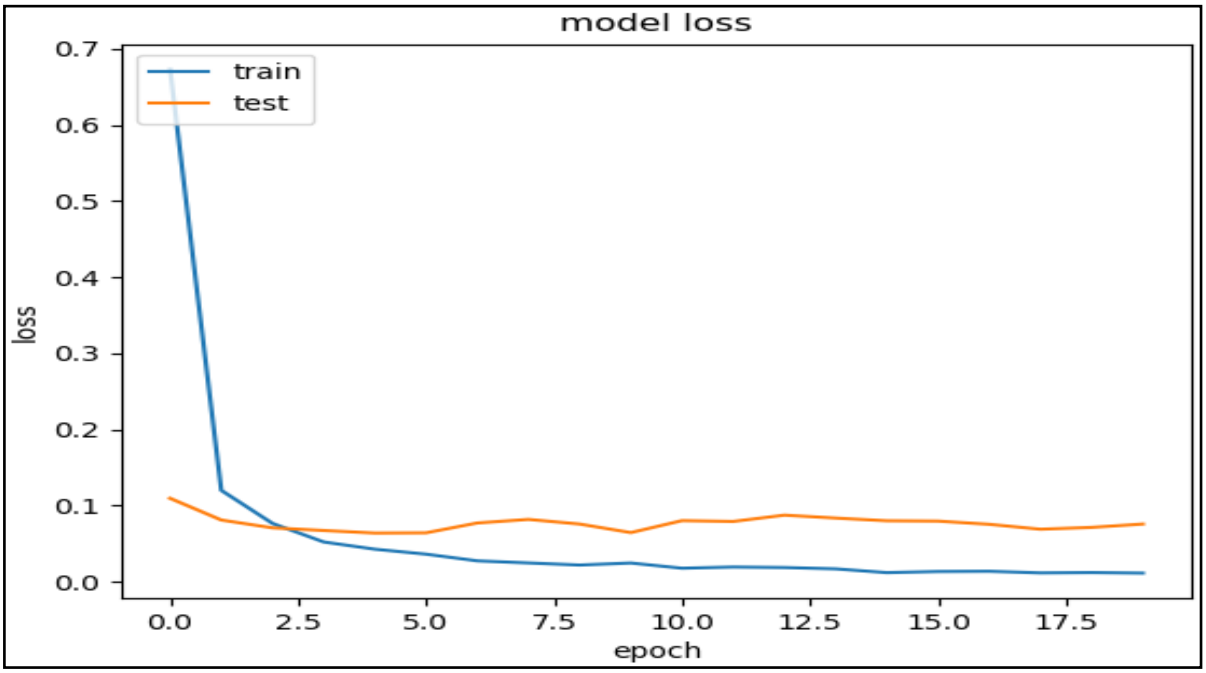


Figure 6.13: Loss of the proposed CNN model with epoch value of 20.

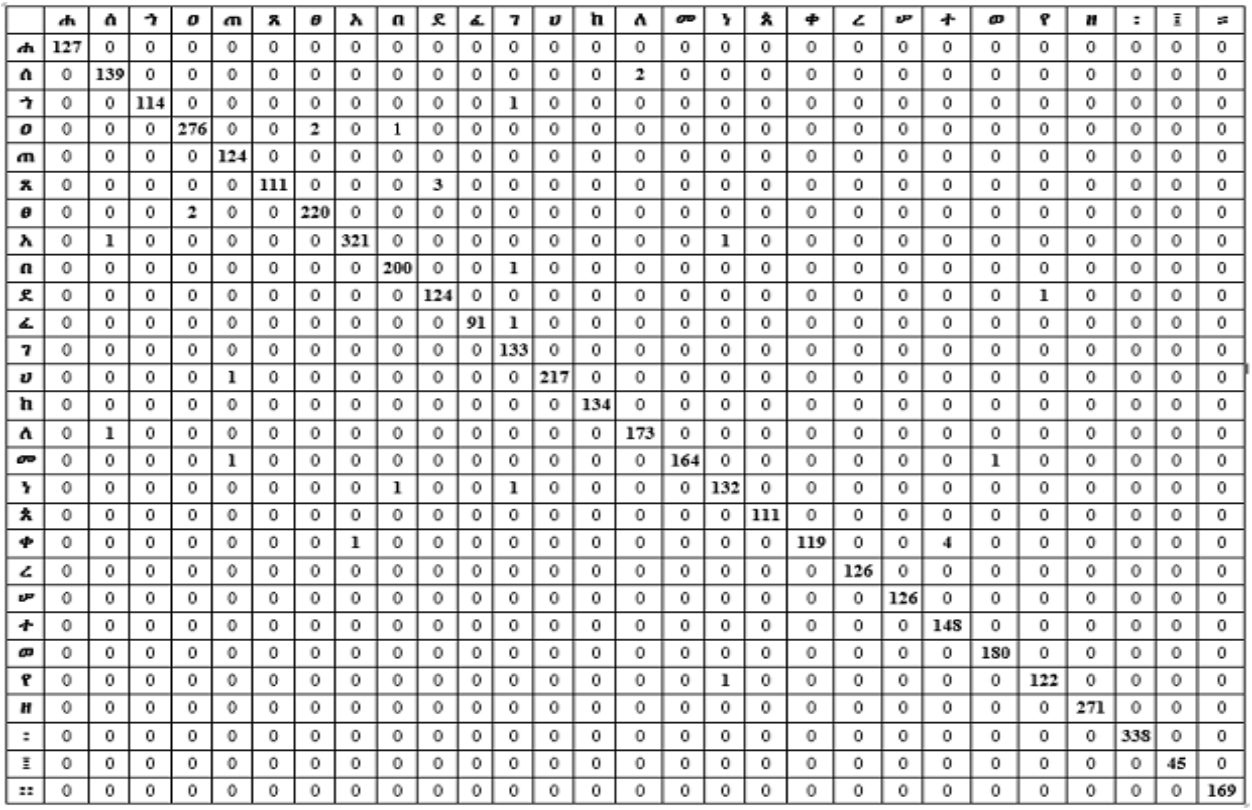


Figure 6.14: Confusion matrix for of the proposed CNN model with the testing dataset

6.4. Comparison between Classification Results of CNN and Deep Neural Network

As we have tried to explain in the previous chapters for the purpose of comparing and proving that the proposed model is better than the other commonly used neural models, we have designed a deep neural network. The designed neural network is trained with the same dataset which was used to train to the proposed convolutional neural network (CNN). We have used three kinds of deep convolutional models which have 1, 2 and 3 hidden layers respectively for getting the best among them and compare it with the proposed convolutional neural network. The training and classification results of the three deep NN models are described in the following Table 6.3.

Table 6.3: Training and classification results of the deep neural network

Hidden Layer	Epoch	Nodes	Test accuracy	Test loss	Val accuracy	Val loss
1	20	256	0.97556	0.10462	0.9773	0.0788
2	20	256	0.98188	0.09504	0.9817	0.0681
3	20	256	0.97316	0.11242	0.9817	0.0737

As we can see it from the above table, the deep neural network model having two hidden layers has obtained a better recognition accuracy and a smaller test loss than the other deep neural

network models. However, when we compare the best result of the deep neural network model with the worst results of the deep convolutional neural network model, we can see that the proposed CNN model for ancient Geez document recognition is much better. As we can see it in the above sections the best recognition accuracy achieved by the proposed deep CNN model is 99.38% with a loss of 0.04442 and the best recognition accuracy obtained from the deep neural network is 98.18% with a model loss of 0.09504. Furthermore, the worst recognition accuracy obtained from the deep CNN is 99.17 and the worst recognition accuracy obtained from the deep neural network is 97.316%. Also, the worst model loss for the proposed system is 0.05239 and the worst model loss for the deep neural network is 0.11242.

Additionally, in research made on ancient Geez character recognition, they proposed three different kinds of neural network models (Siranesh and Menore, 2016). The models have 1, 2 and 3 hidden layers with 100, 200, and 300 number of nodes on each layer. The results obtained by their system is compared with the results of the proposed system in the following Table 6.4.

Table 6.4: Comparison results.

	Testing accuracy	**Testing loss**	**Epoch**	**Neural Model**
Best of the Proposed system	99.389 %	0.044	15	Deep CNN
Best of Siranesh and Menore	93.75 %	0.062	150	Deep NN
Worst of proposed system	99.17 %	0.052	20	Deep CNN
Worst of Siranesh and Menore	92.0 %	0.218	50	Deep NN

6.5. Results of CPU Time

The results listed in Table 6.4 includes the CPU time taken to for grayscale conversion, for binarizing the grayscale image, for denoising, and time taken for skew detection and correction. The results in Table 6.5 are the results of 2 different executions of the stages with similar variables. Hence, the CPU time varies from one execution to another it's essential.

Table 6.5: CPU time results of the processing steps.

No	**Grayscale conversion**	**Denoising**	**Binarization**	**Skew correction**
1	0.000996828079 sec	0.5277755260 sec	0.000997066 sec	0.0089755058 sec
2	0.000998258590 sec	0.5316498279 sec	0.000997391 sec	0.0079770088 sec

Additionally, the following Table 6.5 shows the time taken for the line and character segmentation in seconds required for a similar image which was used in the preprocessing stages. The table displays the results of two different execution with the same specifications.

Table 6.6: CPU time results of the segmentation stages.

No	Line Segmentation	No of lines	Character Segmentation	No of Characters
1	0.004000186920 sec	17	0.012000083923 sec	220
2	0.004002094268 sec	17	0.011999845504 sec	220

Also, in the following Table 6.6 we have displayed the time take for training each of the three proposed deep CNN models based on their epoch value.

Table 6.7: Results of CPU time for training the proposed deep CNN

Epoch	Training Time	Training set	Testing set	Validation set
10	885.5399725437164 sec	16,038	4,583	2,292
15	1178.7913699150085 sec	16,038	4,583	2,292
20	1944.076278924942 sec	16,038	4,583	2,292

The results displayed in the following Table 6.7 are results of CPU time taken for training the deep neural network model according to the number of hidden layers they have.

Table 6.8: CPU time for training the deep neural network models

Hidden Layer	Training Time	Training set	Testing set	Validation set
1	26.06837391853 sec	16,038	4,583	2,292
2	28.89964437484 sec	16,038	4,583	2,292
3	35.80987024307 sec	16,038	4,583	2,292

In the following two tables we have shown the time taken to train and test the proposed system but, with different ratio of training, validation and testing sets. We have divided the dataset into

50%, 30%, and 20% for the training, testing, and validation sets respectively. And the results obtained are displayed it the following two tables.

Table 6.9: Results of CPU time for the deep CNN with different dataset ratio

Epoch	Training Time	Training set	Testing set	Validation set	Testing Accuracy	Testing Loss
10	658.0104596 sec	11,455	6,875	4,583	0.9892	0.0549
15	1062.839679 sec	11,455	6,875	4,583	0.9866	0.0683
20	1462.808309 sec	11,455	6,875	4,583	0.9889	0.0547

6.6. Summary

The proposed convolutional neural network architect is much suitable for the ancient Geez document recognition according to the results obtained from training and testing the proposed model. Also, the preprocessing and segmentation stages results were great and encourageable. However, some of the preprocessing stages are not as efficient as expected as a result of the very noisy and degraded documents. Additionally, the segmentation stage was efficient in many cases but, the character segmentation stage was falling to segment the characters successfully. These character segmentation errors happen because some of the characters are connected to each other and some of them are too small in size. Sometimes these small characters are considered as noise and also segmented as two characters. However, the overall geez document recognition system is successful and we believe we have met our general and specific objectives by developing the proposed system.

CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1.Conclusion

This thesis focusses on developing ancient Geez document recognition system using a deep convolutional neural network. The main aim of a document recognition system is converting scanned papers or images into digital formats i.e. ASCII or Unicode. The development of the proposed character classifier comprises all the necessary process of any character recognition system. The proposed system involves preprocessing, segmentation, feature extraction, and classification stages. Although, for the proposed system we have prepared a dataset that is used for training and testing the proposed system. Hence, there isn't a common dataset that can be used for Geez recognition systems, it was a must to prepare the datasets. The designed system was trained and tested with the images of the 26 base characters and 3 of the punctuation marks of Geez language.

Even though, the images collected were highly noise and degraded the methods used to reduce the noise from the image were successful. Most of the documents are difficult even to recognize through our own eyes however, the convolutional model obtains a greater accuracy. We can see that if we train the network with much more comprehensive dataset, we will get a better result. As we can see it from the results of the proposed system, the convolutional neural network systems have obtained a better recognition result when they are compared with the deep neural network. Although, the method used for line and character segmentation was successful with some minor mistakes. If we can prepare a better dataset that has like ten times than the one, we prepared the system can be used easily to digitize any ancient documents for good. We have achieved an accuracy of 99.389% and loss of 0.044 even if the dataset is not that much large and comprehensive. The result shows that the proposed system is well suited for ancient geez document recognition. The accuracy achieved by the proposed system was very exciting thus, it can motivate and get the attention of other researchers to participate and come up with a better recognition approach.

7.2.Future work

The accuracy of the proposed system can be taken as a better improvement for geez handwritten document recognition. However, the results of the proposed system can be improved by applying

some additional techniques. There are many techniques that can be used to improve the proposed recognition system. Some of the methods that can improve the recognition accuracy of the proposed system are described in the following sections as future work. The prepared dataset was not as good as it should be in terms of quantity. However, in general, the obtained accuracy of recognition with that much smaller dataset is an encouraging result. The amount and quality of a dataset used to train the neural network highly affect the accuracy of any recognition systems just like the number and quality of sources we used to collect information affects the knowledge we get. Thus, preparing a comprehensive dataset for Geez characters can make easier many researchers work to come up with a better recognition system.

Usually, character recognition systems are followed by some kinds of postprocessing techniques for improving the results of classification. These techniques can be of many types like dictionary based or natural language processing (NLP) based. However, the proposed system doesn't include any kind of postprocessing techniques. But if we can apply some kind of postprocessing technique to the proposed character classifier we will get a better result. Basically, the dictionary-based postprocessing techniques are easier to implement and feasible in terms of time, space and cost. However, the NLLP based is considered to be more efficient than dictionary based postprocessing techniques.

REFERENCES

Ahmad, I., & Fink, G. (2016). Class-Based Contextual Modeling for Handwritten Arabic Text Recognition. *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. doi:10.1109/icfhr.2016.0107.

Ajmire, P., Dharaskar, R., Thakare, V., & Warkhede, S. (2012). Structural Features for Character Recognition System-A Review. *International Journal of Advanced Research in Computer Science,3*(3). Retrieved from http://www.ijarcs.info/index.php/Ijarcs

Alembrehan, B. (2017, October 03). የግእዝ ቁጥሮች(Numbers in Geez). Retrieved March 21, 2019, from http://www.sewasew.com/p/የግእዝ-ቁጥሮች-numbers-in-geez.

Alemu, W. (1997). The Application of OCR Techniques to the Amharic Script; (Masters thesis) School of Information studies for Africa, Addis Ababa University, Addis Ababa

Alemu, W., & Fuchs, S. (2003). Handwritten Amharic Bank Check Recognition Using Hidden Markov Random Field. *2003 Conference on Computer Vision and Pattern Recognition Workshop*. doi:10.1109/cvprw.2003.10027

Alom, M. Z., Sidike, P., Hasan, M., Taha, T. M., & Asari, V. K. (2018). Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks. *Computational Intelligence and Neuroscience,2018*, 1-13. doi:10.1155/2018/6747098.

Anzai, Y. (1992). *Pattern recognition and machine learning*. London: Academic Press.

Arel, I., Liu, C., Urbanik, T., & Kohls, A. (2010). Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems,4*(2), 128. doi:10.1049/iet-its.2009.0070

Arulmozhi, K., Perumal, S., Priyadarsini, C., & Nallaperumal, K. (2012). Image refinement using skew angle detection and correction for Indian license plates. *IEEE International Conference on Computational Intelligence and Computing Research*. doi:10.1109/iccic.2012.6510316.

Assabie, Y., & Bigun, J. (2011). Offline handwritten Amharic word recognition. *Pattern Recognition Letters,32*(8), 1089-1099. doi:10.1016/j.patrec.2011.02.007.

Assabie. Y. (2002). Optical character recognition of amharic text: an integrated approach. *Ph.D. dissertation*, Addis Ababa Universisty, Ethiopia.

Bender, M. (1971). The Languages of Ethiopia: A New Lexicostatistic Classification and Some Problems of Diffusion. *Anthropological Linguistics, 13*(5), 165-288. Retrieved from http://www.jstor.org/stable/30029540

Bigun, J. (2008). Writer-independent Offline Recognition of Handwritten Ethiopic Characters. *ICFHR 2008*.

Birhanu, A., & Sethuraman, R. (2015). Artificial Neural Network Approach to the Development of OCR for Real Life Amharic Documents. *International Journal of Science, Engineering and Technology Research (IJSETR),4*(1). Retrieved January 17, 2019.

Blumenstein, M., & Verma, B. (2001). Analysis of segmentation performance on the CEDAR benchmark database. *Proceedings of Sixth International Conference on Document Analysis and Recognition*. doi:10.1109/icdar.2001.953964.

Britannica, The editors of Encyclopedia. (2015, July 24). Ge'ez language. Retrieved March 21, 2019, from http://www.britannica.com/topic/Geez-language#accordion-article-history.

Bu, X., Rao, J., & Xu, C. (2009). A Reinforcement Learning Approach to Online Web Systems Auto-configuration. *2009 29th IEEE International Conference on Distributed Computing Systems*. doi:10.1109/icdcs.2009.76.

Buades, A., Coll, B., & Morel, J. (2011). Non-Local Means Denoising. *Image Processing On Line,1*. doi:10.5201/ipol.2011.bcm_nlm.

Chaki, N., Shaikh, S., & Saeed, K. (2014). Applications of Binarization. *Exploring Image Binarization Techniques Studies in Computational Intelligence,*65-70. doi:10.1007/978-81-322-1907-1_5

Cheriet, M. (2007). *Character recognition systems: A guide for students and practioners*(pp. 32-34). Hoboken, N.J: Wiley Interscience.

Cheriet, M. (2007). *Character recognition systems: A guide for students and practioners*. Hoboken, NJ: John Wiley & Sons.

Choudhary, A. (2014) A review of various character segmentation techniques for cursive handwritten words recognition. *International journal of Information and Computation Technology*, vol. 4, pp. 559-564

Cowell, J., & Hussain, F. (2003). Amharic character recognition using a fast signature based algorithm. *Proceedings on Seventh International Conference on Information Visualization, 2003. IV 2003.*doi:10.1109/iv.2003.1218014.

Diem, M., & Sablatnig, R. (2010). Recognizing characters of ancient manuscripts. *Computer Vision and Image Analysis of Art*. doi:10.1117/12.843532.

Dimov, D. (2001). Using an exact performance of Hough transform for image text segmentation. *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205),*778-781. doi:10.1109/icip.2001.959161.

Doush, I., Alkhateeb, F., & Gharaibeh, A. (2018). A novel Arabic OCR post-processing using rule-based and word context techniques. *International Journal on Document Analysis and Recognition (IJDAR),21*(1-2), 77-89. doi:10.1007/s10032-018-0297-y.

Eckle, K., & Schmidt-Hieber, J. (2019). A comparison of deep networks with ReLU activation function and linear spline-type methods. *Neural Networks,110*, 232-242. doi:10.1016/j.neunet.2018.11.005

Ethiopian literature. (2018, December 24). Retrieved March 20, 2019, from https://en.wikipedia.org/wiki/Ethiopian_literature

Ethiopian manuscript collections. (2019, March 14). Retrieved March 21, 2019, from https://en.wikipedia.org/wiki/Ethiopian_manuscript_collections.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics,36*(4), 193-202. doi:10.1007/bf00344251

Gomes, G., Ludermir, T., & Lima, L. (2011). Comparison of new activation functions in neural network for forecasting financial time series. *Neural Computing and Applications,20*(3), 417-439. doi:10.1007/s00521-010-0407-3

Halcomb, T., Teklu, A., Tadross, A., Gessesse, K., Demisse, D., & Imbert-Vier, S. (March 21, 2019). Amharic numbers. Retrieved March 21, 2019, from http://www.languages andnumbers.com/how-to-count-in-amharic/en/amh/.

Hassaballah, M., & Aly, S. (2015). Face recognition: Challenges, achievements and future directions. *IET Computer Vision,9*(4), 614-626. doi:10.1049/iet-cvi.2014.0084.

Huang, W., Qiao, Y., & Tang, X. (2014). Robust Scene Text Detection with Convolution Neural Network Induced MSER Trees. *Computer Vision – ECCV 2014 Lecture Notes in Computer Science,*497-511. doi:10.1007/978-3-319-10593-2_33

Jain, A., Klare, B., & Park, U. (2011). Face recognition: Some challenges in forensics. *Face and Gesture 2011*. doi:10.1109/fg.2011.5771338

Juan, A., Romero, V., Sánchez, J., Serrano, N., Toselli, A., & Vidal, E. (2010). Handwritten Text Recognition for Ancient Documents. *Journal of Machine Learning Research - Proceedings Track. 11. 58-65.*,58-65. Retrieved from https://www.researchgate.net/ publication/220320874_Handwritten_Text_Recognition_for_Ancient_Documents.

Kavallieratou, E., Sgarbas, K., Fakotakis, N., & Kokkinakis, G. (2003). Handwritten word recognition based on structural characteristics and lexical support. *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* doi:10.1109/icdar.2003.1227727.

Khan, R., & Mohammad, Z. (2008). A simple Segmentation Approach for Unconstrained Cursive Handwritten Words in Conjunction With Neural Network. *International Journal of Image Processing. 2*.

Kim, G., Govindaraju, V., & Srihari, S. N. (1999). An architecture for handwritten text recognition systems. *International Journal on Document Analysis and Recognition,2*(1), 37-44. doi:10.1007/s100320050035.

Laskov, L. (2006). Classification and Recognition of Neume Note Notation in Historical Documents. *International Conference on Computer Systems and Technologies*. Retrieved January 15, 2019.

Leviney, S., Finny, C., Darrell, T., & Abbeel, P. (2016). End-to-end Training of Deep Visuomotor Policies. *Journal of Machine Learning Research*,1-40. Retrieved from https://arxiv.org/abs/1504.00702.

Liu, C., Nakashima, K., Sako, H., & Fujisawa, H. (2003). Handwritten digit recognition: Benchmarking of state-of-the-art techniques. *Pattern Recognition*,*36*(10), 2271-2285. doi:10.1016/s0031-3203(03)00085-2.

Liu, C., Nakashima, K., Sako, H., & Fujisawa, H. (2004). Handwritten digit recognition: Investigation of normalization and feature extraction techniques. *Pattern Recognition*,*37*(2), 265-279. doi:10.1016/s0031-3203(03)00224-3

Liu, C., Sako, H., & Fujisawa, H. (2003). Handwritten Chinese character recognition: Alternatives to nonlinear normalization. *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*doi:10.1109/icdar.2003.1227720.

Lo, S., Chan, H., Lin, J., Li, H., Freedman, M., & Mun, S. (1995). Artificial convolution neural network for medical image pattern recognition. *Neural Networks*,*8*(7-8), 1201-1214. doi:10.1016/0893-6080(95)00061-5.

Malanker, A., & Patel, P. (2014). Handwritten Devanagari Script Recognition: A Survey. *IOSR Journal of Electrical and Electronics Engineering*,*9*(2), 80-87. doi:10.9790/1676-09228087.

Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016). Resource Management with Deep Reinforcement Learning. *Proceedings of the 15th ACM Workshop on Hot Topics in Networks - HotNets 16*. doi:10.1145/3005745.3005750

Meshesha, M., & Jawahar, C. V. (2007). Optical Character Recognition of Amharic Documents. *African Journal of Information & Communication Technology*,*3*(2). doi:10.5130/ajict.v3i2.543.

Meyer-Bäse, A. (2004). *Pattern recognition for medical imaging*. Amsterdam: Elsevier Academic Press.

Najafiragheb, N., Hatam, A., & Harifi, A. (2016). A Survey of Feature Extraction Techniques in OCR. *2016 1 St International Conference on New Research Achievements in Electrical and Computer Engineering*.

Nasien, D., Haron, H., & Yuhaniz, S. S. (2010). Support Vector Machine (SVM) for English Handwritten Character Recognition. *2010 Second International Conference on Computer Engineering and Applications*. doi:10.1109/iccea.2010.56

Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics,9*(1), 62-66. doi:10.1109/tsmc.1979.4310076

Pal, U., Wakabayashi, T., & Kimura, F. (2007). Handwritten Bangla Compound Character Recognition Using Gradient Feature. *10th International Conference on Information Technology (ICIT 2007)*. doi:10.1109/icit.2007.62

Plamondon, R., & Srihari, S. (2000). Online and off-line handwriting recognition: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence,22*(1), 63-84. doi:10.1109/34.824821.

Pradeep, J. (2012). Neural Network Based Recognition System Integrating Feature Extraction and Classification for English Handwritten. *International Journal of Engineering,25*(2 (B)), 99-106. doi:10.5829/idosi.ije.2012.25.02b.03

Pradeep, J., Srinivasan, E., & Himavathi, S. (2011). Diagonal Based Feature Extraction for Handwritten Alphabets Recognition System Using Neural Network. *International Journal of Computer Science and Information Technology,3*(1), 27-38. doi:10.5121/ijcsit.2011.3103.

Puneet, P., & Garg, N. (2013). Binarization Techniques used for Grey Scale Images. *International Journal of Computer Applications,71*(1), 8-11. doi:10.5120/123208533

Sahle, E. (2015, October 29). Top Ten Fascinating Facts about Ethiopia Media has not told you. Retrieved January 2, 2019, from https://africabusiness.com/2015/10/29/ethiopia-4/

Saif, A., Prabuwono, A., & Mahayuddin, Z. (2015). Moment Feature Based Fast Feature Extraction Algorithm for Moving Object Detection Using Aerial Images. *Plos One,10*(6). doi:10.1371/journal.pone.0126212.

Shafii, M. (2014). Optical Character Recognition of Printed Persian/Arabic Documents. Retrieved December 16, 2018, from https://scholar.uwindsor.ca/etd/5179.

Shi, M., Fujisawa, Y., Wakabayashi, T., & Kimura, F. (2002). Handwritten numeral recognition using gradient and curvature of gray scale image. *Pattern Recognition,35*(10), 2051-2059. doi:10.1016/s0031-3203(01)00203-5

Siranesh. G., & Menore. T. (2016). *Ancient Ethiopic Manuscript Recognition Using Deep Learning Artificial Neural Network*(Unpublished master's thesis). Addis Ababa University.

Suzuki, S., & Abe, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing,29*(3), 396. doi:10.1016/0734-189x(85)90136-7.

Verma, R., & Malik, L. (2015). Review of Illumination and Skew Correction Techniques for Scanned Documents. *Procedia Computer Science,45*, 322-327. doi:10.1016/j.procs.2015.03.151.

Yousefi, M., Soheili, M., Breuel, T., Kabir, E., & Stricker, D. (2015). Binarization-free OCR for historical documents using LSTM networks. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. doi:10.1109/icdar.2015.7333935.

Zha, S., Luisier, F., Andrews, W., Srivastava, N., & Salakhutdinov, R. (2015). Exploiting Image-trained CNN Architectures for Unconstrained Video Classification. *Procedings of the British Machine Vision Conference 2015*. doi:10.5244/c.29.60

Zhou, Z., Li, X., & Zare, R. N. (2017). Optimizing Chemical Reactions with Deep Reinforcement Learning. *ACS Central Science,3*(12), 1337-1344. doi:10.1021/acscentsci.7b00492.

APPENDICES

APPENDIX 1:

RESULTS OF PREPROCESSING STAGE

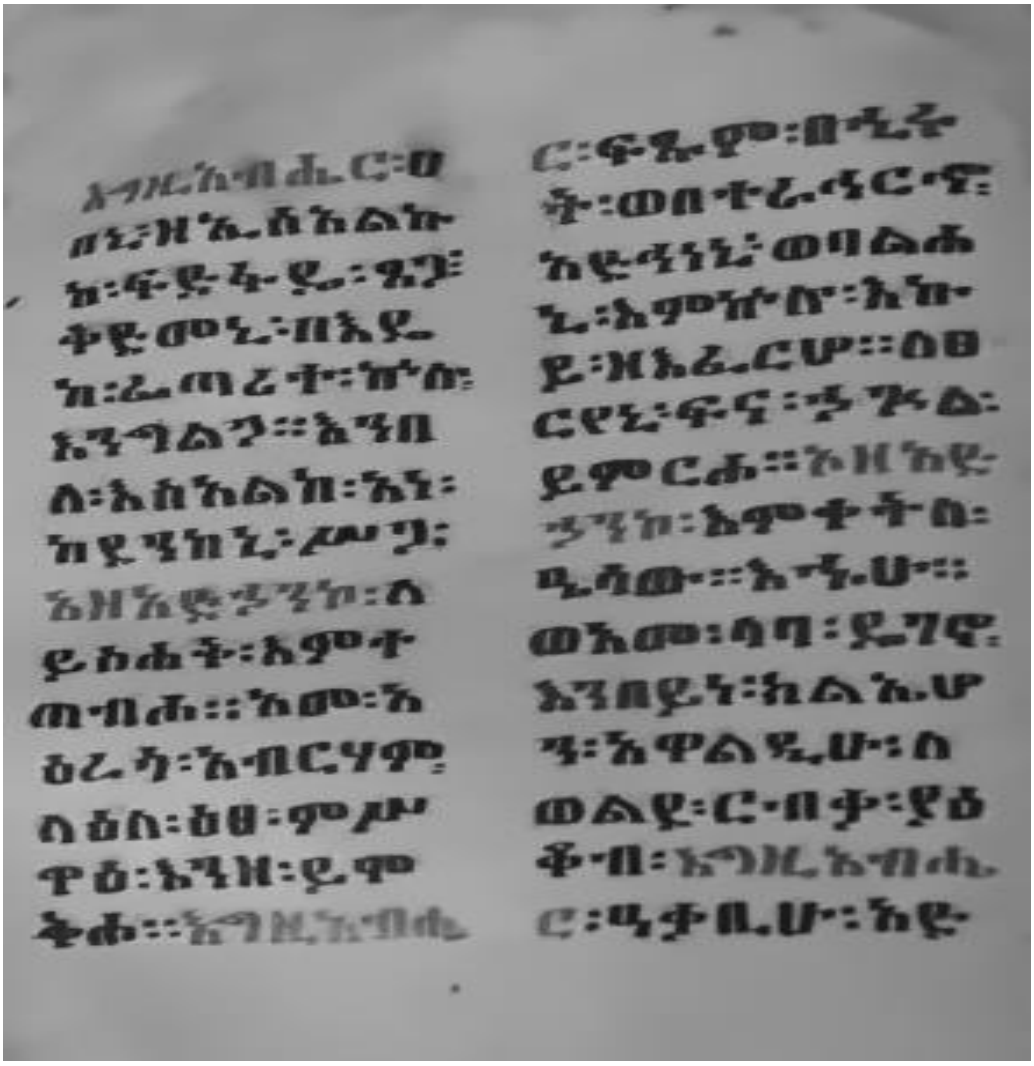


Figure A1.1: Errors of noise reduction

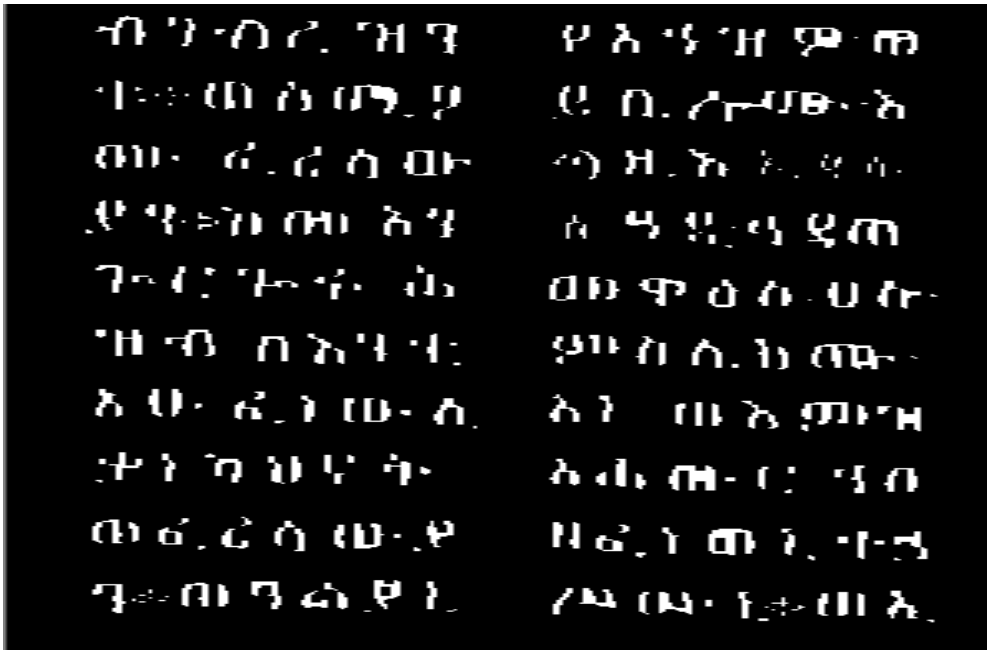


Figure A1.2: Errors of erosion morphological transformations

Figure A1.3: Errors of line segmentation

Figure A1.4: Errors of character segmentation

Figure A1.5: Errors of noise reduction

APPENDIX 2:

SAMPLE IMAGES USED FOR DATASET PREPARATION

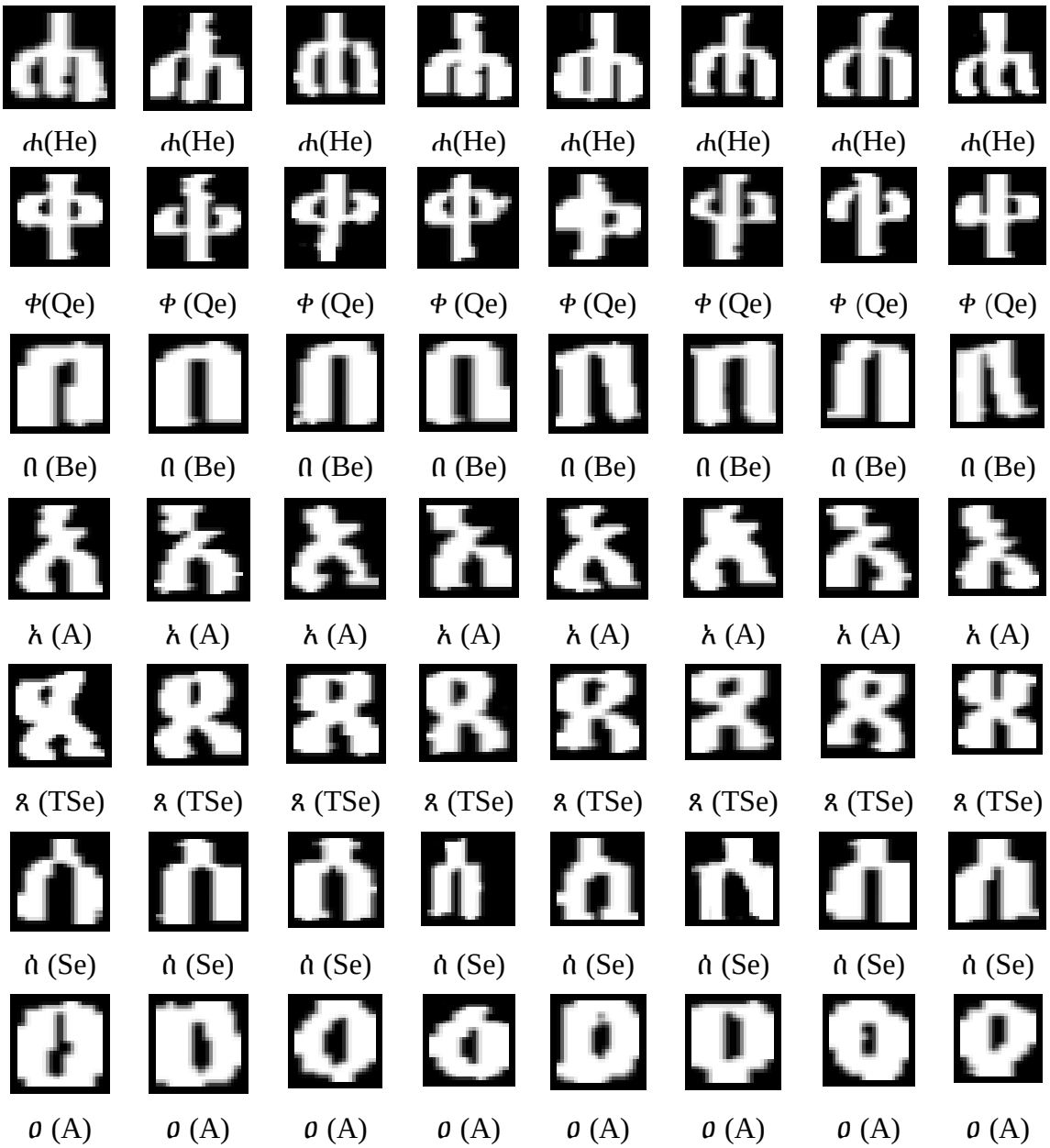


Figure B1.1: Sample character images used for dataset preparation

APPENDIX 3:

SOURCE CODES

1. Preprocessing Method

```
def pre_processing():
        # read in testing numbers image
        original_image_test = cv2.imread("aa_n.jpg")

        # if image was not read successfully
        if original_image_test is None:
                # print error message to std out
                print("error: image can't be found from file \n\n")
                # pause so user can see error message
                os.system("pause")
                # and exit function (which exits program)
                return

        # Gray scale conversion
        grayed_test_image = cv2.cvtColor(original_image_test, cv2.COLOR_BGR2GRAY)

        # Resize image
        grayed_test_image = cv2.resize(grayed_test_image,\
        (400, 600), interpolation=cv2.INTER_AREA)
        original_image_test = cv2.resize(original_image_test,\
        (400, 600), interpolation=cv2.INTER_AREA)

        # Apply mean noise reduction method
        noise_removed_image = cv2.fastNlMeansDenoising(grayed_test_image, None, 10, 7, 21)

        # OTSU threshold
        ret3, threshold_image = cv2.threshold(noise_removed_image,\
        0, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)
        return threshold_image
# end of preprocessing
```

2. Skew detection and correction method

```
def check_for_skew_and_correct_image(thresh):
        # Take the x, y coordinates of all pixel values that
        # have white intensity, then use these coordinates to
        # compute a rotated bounding box that contains all
        # coordinates
        coordinates = np.column_stack(np.where(thresh > 0))
        angle = cv2.minAreaRect(coordinates)[-1]

        # the `cv2.minAreaRect` function returns values in the
        # range [-90, 0); as the rectangle rotates clockwise the
        # returned angle trends to 0 -- in this special case we
        # need to add 90 degrees to the angle
        if angle < -45:
                angle = -(90 + angle)

        # otherwise, just take the inverse of the angle to make
        # it positive
        else:
                angle = -angle

        # rotate the image to de_skew it
        (h, w) = thresh.shape[:2]
        center = (w // 2, h // 2)
        rotation_matrix = cv2.getRotationMatrix2D(center, angle, 1.0)
        thresh = cv2.warpAffine(thresh, rotation_matrix, (w, h),\
        flags=cv2.INTER_CUBIC, borderMode=cv2.BORDER_REPLICATE)
        return thresh
```

3. Line segmentation method

```
def extract_all_lines(image_thresh, image_original):
        image_original_before = image_original.copy()
        # shade every line this will make easier to locate and find out lines
        kernel = np.ones((3, 100), np.uint8)

        img_dilation = cv2.dilate(image_thresh, kernel, iterations=1)

        # Find all contour
        im2, ctr_s, var_three = cv2.findContours(img_dilation.copy(), /
        cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
        all_contours_with_data = []
        valid_contours_with_data = []

        for npaContour in ctr_s:  # for each contour
                # instantiate a contour with data object
                contour_with_data = ContourWithData()
                # assign contour to contour with data
                contour_with_data.contour = npaContour
                # get the bounding rect
                contour_with_data.boundingRect = cv2.boundingRect(contour_with_data.contour)
                # get bounding rect info
                contour_with_data.calculate_bounding_attributes()
                # calculate the contour area
                contour_with_data.fltArea = cv2.contourArea(contour_with_data.contour)
                # add contour with data object to list of all contours with data
                all_contours_with_data.append(contour_with_data)
        # end for

        for contour_with_data in all_contours_with_data:  # for all contours
                if contour_with_data.is_contour_valid():  # check if valid
                        if contour_with_data.intRectHeight >= RE_SIZED_IMAGE_HEIGHT - 10 /
                                and contour_with_data.intRectWidth >= RE_SIZED_IMAGE_WIDTH - 10:
                                # if so, append to valid contour list
                                valid_contours_with_data.append(contour_with_data)
                # end if
        # end for

        valid_contours_with_data = sorted(valid_contours_with_data, /
        key=lambda contour_data_s: contour_data_s.intRectY)
```

```
sorted_contours = []
for sor_c in valid_contours_with_data:
        sorted_contours.append(sor_c.contour)

all_contours_line = []
all_contours_line_original = []

for i, ctr in enumerate(sorted_contours):
        # Get bounding box
        x, y, w, h = cv2.boundingRect(ctr)
        # Crop the line
        roi = image_thresh[y:y + h, x:x + w]
        roi_original = image_original_before[y:y + h, x:x + w]
        roi_copy = roi.copy()
        all_contours_line_original.append(roi_original)
        all_contours_line.append(roi_copy)
return all_contours_line, all_contours_line_original
```

4. CNN design method

```
def build_cnn_model():
        # the first convolutional layer
        model.add(Conv2D(32, kernel_size=5, padding='same', \
        activation='relu', input_shape=input_shape))
        # the first max pooling
        model.add(MaxPooling2D((2, 2)))
        model.add(Dropout(0.25))

        # Second Convolutional layer
        model.add(Conv2D(64, kernel_size=3, padding='same', activation='relu'))
        model.add(Dropout(0.2))

        # third convolutional layer
        model.add(Conv2D(128, kernel_size=2, activation='relu'))

        # flatten the final output of the convolutional feature
        model.add(Flatten())

        model.add(Dense(256, activation='relu'))
        model.add(Dropout(0.2))

        model.add(Dense(256, activation='relu'))
        model.add(Dropout(0.2))

        model.add(Dense(28, activation='softmax'))

        model.compile(loss=keras.losses.categorical_crossentropy, \
        optimizer=keras.optimizers.Adam(), metrics=['accuracy'])
        model.summary()
```