Deep Learning for Amharic Text-Image Recognition: Algorithm, Dataset and Application

Thesis

approved by
the Department of Computer Science
Technische Universität Kaiserslautern
for the award of the Doctoral Degree
Doctor of Engineering (Dr.-Ing.)

to

Birhanu Hailu Belay

Date of Defense: 10.02.2021
Dean: Prof. Dr. Jens Schmitt
Chairperson of the PhD Committee:
Prof. Dr. Ralf Hinze
Reviewer: Prof. Dr. Didier Stricker
Reviewer: Prof. Dr. Marcus Liwicki

D 386

Abstract

Optical Character Recognition (OCR) is one of the central problems in pattern recognition. Its applications have played a great role in the digitization of document images collected from heterogeneous sources. Many of the well-known scripts have OCR systems with sufficiently high performance that enables OCR applications in industrial/commercial settings. However, OCR systems yield very-good results only on a narrow domain and very-specific use cases. Thus, it is still a challenging task, and there are other exotic languages with indigenous scripts, such as Amharic, for which no well-developed OCR systems exist.

As many as 100 million people speak Amharic, and it is an official working language of Ethiopia. Amharic script contains about 317 different alphabets derived from 34 consonants with small changes. The change involves shortening or elongating one of its main legs or adding small diacritics to the right, left, top, or bottom of the consonant character. Such modifications lead the characters to have similar shapes and make the recognition task complex, but this is particularly interesting for character recognition research. So far, Amharic script recognition models are developed based on classical machine learning techniques, and they are very limited in addressing the issues for Amharic OCR. The motivation of this thesis is, therefore, to explore and tailor contemporary deep learning techniques for the OCR of Amharic.

This thesis addresses the challenges in Amharic OCR through two main contributions. The first contribution is an algorithmic contribution in which we investigate deep learning approaches that suit the demand for Amharic OCR. The second is a technical contribution that comprises several works towards the OCR model development; thus, it introduces a new Amharic database consisting of collections of images annotated at a character and text-line level. It also presents a novel CNN-based framework designed by leveraging the grapheme of characters in Fidel-Gebeta (where Fidel-Gebeta consists of the full set of Amharic characters in matrix structure) and achieves 94.97% overall character recognition accuracy.

In addition to character level methods, text-line level methods are also investigated and developed based on sequence-to-sequence learning. These models avoid several of the pre-processing stages used in prior works by eliminating the need to segment individual characters. In this design, we use a stack of CNNs, before the Bi-LSTM layers and train from end-to-end. This model outperforms the LSTM-CTC based network, on average, by a CER of 3.75% with the ADOCR test set. Motivated by the success of attention, in addressing the problems' of long sequences in Neural Machine Translation (NMT), we proposed a novel attention-based methodology by blending the attention mechanism into CTC objective function. This model performs far better than the existing techniques with a CER of 1.04% and 0.93% on printed and synthetic text-line images respectively.

Finally, this thesis provides details on various tasks that have been performed for the development of Amharic OCR. As per our empirical analysis, the majority of the errors are due to poor annotation of the dataset. As future work, the methods proposed in this thesis should be further investigated and extended to deal with handwritten and historical Amharic documents.

Acknowledgments

First, I would like to thank Prof. Dr. Didier Stricker for allowing me to conduct my Ph.D. research at the Augmented Vision lab, DFKI-Kaiserslautern. Without his ongoing support, supervision, feedback, and the freedom to investigate different approaches, this thesis would not have been possible. Further, I would like to thank Prof. Dr. Marcus Liwicki for becoming my cosupervisor, his deep insights, invaluable feedback, fruitful discussions, and many great ideas are the cornerstones of this research work. I would like to thank Dr. Million Meshesha and Dr. Gebeyehu Belay, my mentors in the Ph.D. program for their guidance and feedback on my research.

I would like to thank Tewodros, who led to the first conceptual ideas for this thesis. Not only the long technical discussions but also your continuous motivation and support have always been my asset. Thank you for taking the time for the many fruitful discussions that not only challenged me but also helped me to develop many of the ideas behind this thesis and teaching me the importance of hard work. Then, Fatemeh, you and Teddy were always there to facilitate and made things easy during my conference registration wherever and whenever it is. Special thanks to Reza, who initially helped me a lot in understanding various ideas regarding my topic and letting me know and use the OCRopus tool. Many thanks also to Kina for her valuable feedback and proofreading towards the end of this thesis. I would like to acknowledge David, who is always there to maintain and provide a solution immediately when my computing infrastructure faces any problem. I would like to acknowledge Dr. Jason for his support during the thesis write-up. Whenever I drop an email to him, he always replied to the point and immediately. Last but not least, special thanks to Leivy, who helped me a lot from the very beginning of my arrival at DFKI until I was officially accepted as a Ph.D. student at TU-Kaiserslautern. Further, I would like to thank Keonna, Nicole, and Eric for helping me with various administrative issues.

I would like to thank Abrham Gebru, for spending countless hours together. Especially, every random weekend trip, we had to Frankfurt to buy Injera. Even though it was my first winter semester in Germany, he made me feel like it's summer. My best wishes to him. I would like to thank Dr. Tsegaye, Dr. Dereje, and Dr. Mulugeta, for all the times we spent in KL. I would especially like to thank Dere, for being there for me during the pandemic, he is a very sociable person and we had a good friendship and spent lots of time together and he was a very great chef of our kitchen.

It is a pleasure to express my gratitude wholeheartedly to my wife, Masti, for all her love, patience, and having my back in stressful times. She has also waited and stayed with me till midnight, and she is always there to encourage me when I work the whole night. My daughter, Nissan, and my son, Zephaniah, are not only my kids, but you are also my friends, who understand me when I work and bring out the best in me when I got bored. I am also very thankful to my brother, Workye Hailu, for caring in all my ways. My mother, Enatalem, for encouraging and supporting me in all my studies starting from early schools. I am also very grateful to my father, Hailu Belay, and the rest of all my families and friends who have a significant contribution and help me in one way or another during my research work.

Birhanu Hailu

Contents

List of Figures

List of Tables

Abbreviations

OCR	Optical Character Recognition
DIA	Document Image Analysis
NLP	Natural Language Processing
MTL	Multi-Task Learning
NMT	Neural Machine Translation
EOTC	Ethiopian Orthodox Tewahedo Church
ENALA	Ethiopian National Archives and Library Agency
PR	Pattern Recognition
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
CNN	Convolutional Neural Network
Bi-LSTM	Bidirectional LSTM
CTC	Connectionist Temporal Classification
HMM	Hidden Markov Models
ADOCR	Amharic Database for OCR
ANN	Artificial Neural Network
SVM	Support Vector Machine
FCNN	Factored CNN
CRNN	Convolutional RNN
CER	Character Error Rate
FCN	Fully Connected Network
BACN	Blended Attention CTC
AED	Attention based Encoder-Decoder

Chapter 1

Introduction

The history of Optical Character Recognition (OCR) is traced back to the 1809s [Her82, DT+14] when the first reading machine for the visually impaired was patented. Since then, it is being applied throughout the spectrum of industries, resulting in the revolutionization of the document management process. The application of character recognition is not limited to converting image data to text but it has also proliferated with various businesses including banking, legal industries, health care systems, institutional repositories and digital libraries, all over the world. In previous years, there were a number of problems that affected the implementation of OCR due to its' computationally intensiveness and training data shortage. Further, early versions of OCR had to be trained with images of each character and was limited to recognizing a single font at a time. However, nowadays, the introduction of faster Graphical Processor Units (GPU) and implementation of deep learning algorithms have enabled lots of exciting new features such as automatic multiple feature extraction, scalability for complex images and documents, which pushed the achievements of sequence-to-sequence tasks and OCR applications further.

Optical Character Recognition (OCR) refers to a set of computer vision problems that are employed to convert images of machine-printed texts or hand-written text images to machine-readable text in a format that a computer can process, store, and edit as a text file; or as a part of data entry for further manipulation [BT14]. Digitization of textual resources and hard-copy documents including books, newspapers, magazines, cultural and religious archives have been underway for decades to promote world-wide accessibility of these resources. Nowadays, documents are processed with computers, however, it is unclear whether the computer has decreased or increased

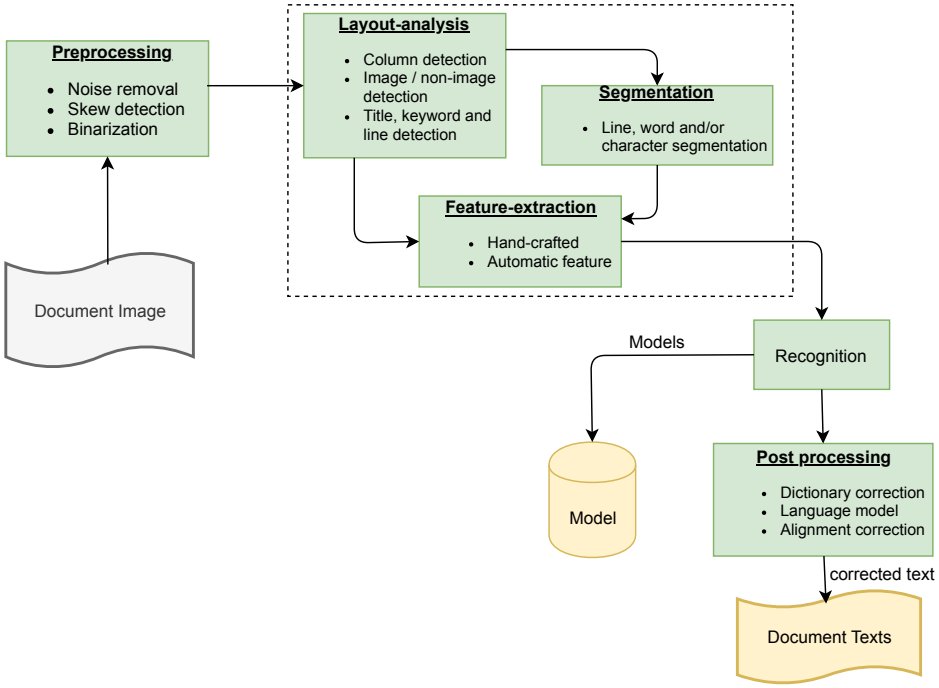


Figure 1.1: **A generic Document Image Analysis (DIA) flow diagram .** *This generic DIA flow diagram can be broken down into four major steps: The first step is image preprocessing which helps to enhance the quality of the image and it involves broad range of imaging functions such as image rotation, binarization and deskewing. The second document analysis step defines the areas for text recognition and delivers information about layout and formatting elements of each individual page as well as structure of the document as a whole. The actual texts are predicted at recognition step. Then the OCR errors are corrected and the model is updated at the post-processing stage.*

the amount of paper-based documents. A large number of documents, having different contents and written with different scripts, are still processed, archived, and disseminated in paper format [KJEY20]. During recognition, once a printed and/or handwritten document has been captured optically using digital camera, scanner or any other optical means, then the digital image goes to the following major steps of Document Image Analysis (DIA) for further processing and recognition. First, it goes to the preprocessing stage and then it goes to the feature extraction stage. Finally, the features obtained at the feature extraction step are utilized for the development of the recognition model. Figure 1.1 depicts the generic process of DIA and basic activities in each step.

- **Preprocessing stage**: In this stage the quality of the images are enhanced employing different image preprocessing techniques and the data of interest are located.
- **Feature extraction**: The distinctive characteristics of the image, useful for recognition, are captured in the feature extraction stage. Depending on the type of algorithm employed for feature extraction, document layout correction and image segmentation could be applied at different levels.
- **Classification stage**: Here, the extracted feature vectors from the previous stage are processed to recognize character, word, or text-lines.
- **Post-processing**: At this step, the recognition errors are corrected based on contextual information from the language-model or dictionary. As a result, post-processing helps to update and/or improve the recognition performance of the OCR model.

The applications of OCR have been widely used and implemented for the digitization of various documents, written in both Latin and Non-Latin scripts, ranging from historical to modern documents [Mes08, BHL$^+$19a, CPT$^+$19, IFD$^+$19]. Researchers achieved a better recognition accuracy and even most of these scripts, now, have workable and commercial off-the-shelf OCR application including the functionality of the ground truth generator from the exiting printed text so as to train the second model [Bre08]. However, OCR gives a better recognition accuracy only on a narrow domain and very specific use cases. Besides, other multiple indigenous scripts are underrepresented in the area of Natural Language Processing (NLP) and Document Image Analysis (DIA). Hence, researchers are motivated towards the development of either a generic multi-script OCR model or an OCR model for specific scripts.

Dated back to the 12^{th} century, most of the historical and literary documents in Ethiopia are written and documented using Amharic script [Mey06]. In parallel, there are some documents written in Ge'ez script which share the same writing system and characters as Amharic script. Since then, multiple documents containing various contents are stored in different places such as Ethiopian Orthodox Tewahdo Churches, museums, monasteries, public and academic libraries in the form of correspondence letters, magazines, newspapers, pamphlets, and books [Hag15]. These documents contain information

[1] S. the on-line database Mazgaba seelat (Ruba Kusa); on historical carpets of Rubakusa, s. Gervers 2004:292-93: Grant ES. Ethio-SPaRe Cultural Heritage of Christian Ethiopia: Salvation, Preservation and Research.

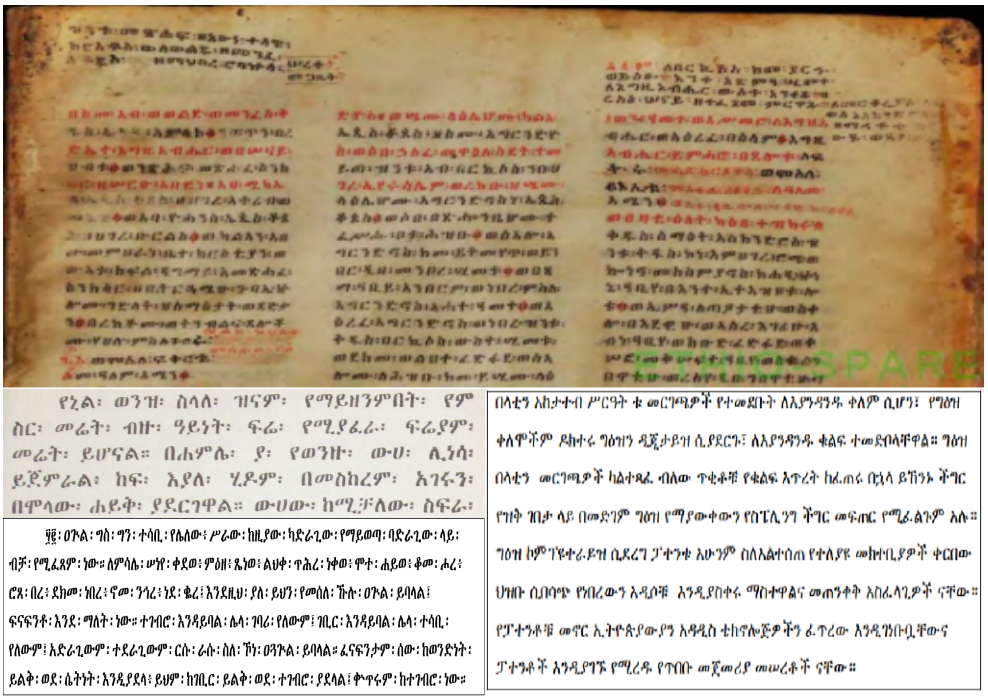


Figure 1.2: **Sample Amharic and Ge'ez document collections.** *The top image of this figure is Ge'ez document [1] while the others are Amharic documents. Each document image depicts variations of fonts, styles, and layouts of Amharic script that are limiting the implementation of a robust OCR system. The characters of Ge'ez are a subset of Amharic characters; thus, from the OCR perspective, a recognition model of Amharic script can be directly applied and/or extended to Ge'ez script recognition.*

related to religion, history, literature, politics, economics, philosophy, academic, traditions, nature, and cultural pieces of evidence of the people of Ethiopia. With a digitization campaign, many of these manuscripts are collected from various sources. However, they are still preserved in a manual catalog and/or scanned copies of them in Microfilm format [Wio06]. Sample Amharic and, its ancestor, Ge'ez document collections are shown in Figure 1.2. These documents consist of representative images of historical, modern, handwritten and machine-printed documents with varieties of writing style and document layouts.

Nowadays, document processing and preservation has been given much attention by many researchers from the field of linguistics and social science [MM18, AMT+18]. In this regard, the Ethiopian Orthodox Tewahedo Church (EOTC) and Ethiopian National Archives and Library Agency (ENALA) has

played a crucial role in shaping national strategy and supporting digital innovation by the public organization with their implementation of digital initiatives [CBC19, NAL]. To support librarian, historian, and other experts in the humanity studies, academic and industrial researchers have been attempting to develop OCR model to digitize textual-resources and documents for various scripts. However, the nature of the script, such as type of fonts, representation of encoding, quality of printing and support from operating systems affect document standardization, causing the languages to have diverse features in nature [Mes08]. Consequently, these issues add to the complexity of the design and implementation of document image recognition systems to a great extent for many scripts including Amharic script.

1.1 Motivation

The main motivation for this thesis is the research gap in the OCR development for Amharic script. Amharic follows a unique writing system that has its specific linguistic composition and indigenous characters. Amharic language has a rich collection of documents that are collected from various sources over centuries that should be digitized for further research and knowledge dissemination. Consequently, there is a need to develop an OCR model for Amharic script. However, despite the need for developing the OCR model, there is a very limited research attempt for developing a robust and automatic OCR system for Amharic script.

So far, the research attempts made for Amharic OCR development are not well organized. These attempts are also implemented based on classical machine learning algorithms which require lots of effort for prepossessing and feature extraction. The type and number of characters considered in the training set are not enough to represent the occurrence of characters even in common Amharic documents. In addition, the performance of the OCR model in general and specifically the OCR models developed for Amharic script recognition are affected by the nature and varieties of document images. The unavailability of organized datasets for training is also one of the limiting factors in Amharic OCR development. Hence, a large annotated real-world dataset is required to train OCR systems and improve their performance. However, it is a very challenging and time-consuming task to manually annotate real-world documents. Therefore, a method is needed, which can automatically generate ground truth for character or text-line images that can represent real-world artifacts of Amharic documents. This motivated the author to introduce a par-

ticular and simple method for the preparation of printed texts and automatic ground-truth generation of synthetic Amharic text-line images.

1.2 Statement of the Problem

The present growth of document digitization and changes in the world of the linguistic landscape demands an immediate solution for enabling information access for everyone. This requires research in the area of natural language processing and document image analysis. In parallel, the demands of language technology are increasing for resource-limited languages, to break language barriers. Amharic is one of these resource-limited languages with about 100 million speakers around the world. It has an indigenous script with its own alphabets. Amharic is rich in historical documents collected over centuries. However, from an OCR perspective, there is only limited research effort for Amharic scripts. The scripting nature of the language and lack of data availability poses many challenges in document image analysis and recognition tasks.

For decades, OCR was the only way to turn printed documents into data that could be processed by computers, and it remains the tool of choice for converting paper documents into editable data that can be processed further. Ultimately, OCR will continue to be a valuable tool for filling in gaps of humanity and the digital world. Since the truly paperless business doesn't yet exist, data extraction is still a useful tool and that can augment the development of OCR technology to minimize the scope for errors.

Deep learning-based approaches have improved over the last few years, reviving an interest in the OCR problem, where neural networks can be used to combine the tasks of localizing texts in an image along with understanding what the texts are. Using CNN architectures, Attention mechanisms and RNNs have gone a long way in this regard. However, to the best of our knowledge, researches are done on Amharic OCR so far is in an uncoordinated manner and none of the researchers have taken the advantage of deep learning techniques such as sequence-to-sequence learning instead segmentation-based traditional machine learning techniques have been employed nearly in all of these attempts. In addition, in the literature, attempts to Amharic OCR neither shown results on a large dataset nor considering all possible characters used in Amharic script. Therefore, this work aims to explore the problem of OCR in Amharic script and solve some of these problems by applying

state-of-the-art deep learning algorithms. In parallel, we develop a baseline Amharic database so as to address the issue of the dataset.

1.3 Research Objectives

Given the increasing need to digitize documents and its impact on accessibility of indigenous knowledge, this thesis aims to construct a robust OCR model for Amharic script recognition using contemporary deep learning algorithms. This thesis also aims to reduce the research gap in Amharic language and to develop a baseline database for Amharic OCR which can help researchers for further improvement of the OCR system. To achieve the objective of this thesis, we pose the following theoretical hypotheses and we are going to test them through an empirical investigation.

1. The performance of OCR model for basic Amharic character recognition can be improved if we develop a model of multi-task networks by leveraging the grapheme of characters on Fidel-Gebeta.

2. The ability of RNNs to process contextual information enables them to learn sequence-to-sequence mapping; thus, stacking CNNs before RNNs as a feature extractor and employing CTC alignment making them an appropriate choice for the task of text-image recognition.

3. In NMT, Attention enhances the ability of the networks in extracting the most relevant feature for each part of the output sequence; thus, it suits and performs better if we extend Attention mechanism for Amharic text-line image recognition.

4. Given that the second and third hypotheses are correct, then using the best of both worlds and blending the capability of Attention mechanism into CTC objective function boosts the recognition performance of the OCR model.

1.4 Contributions

In this thesis, we focus on designing an OCR model for Amharic script. Some of the main conceptual and technical contributions of this thesis are summarized as follows:

- A well structured and standard database of basic Amharic character images is developed in this research work. The constructed database consists of 80,000 character images and it has the potential to be used as a benchmarking resource for Amharic character recognition.

- Created the first publicly available Amharic text-line image dataset. The dataset is a collection of 337,337 synthetically generated and printed text-images. It is also evaluated by state-of-the-art OCR techniques such as LSTM-CTC, and Attention-based encoder-decoder networks.

- Propose a novel Approach for basic Amharic character recognition. To the best of our knowledge, it is the first work designed as a multi-task architecture by leveraging the structure of basic Amharic characters grapheme's on Fidel Gebeta.

- Designed an OCR for recognizing Amharic scripts at text-line level. These OCR models are extensively tested on both synthetically generated and printed images and promising results are obtained.

- Adoption of synthetic database generation methodology and propose techniques for printed text-image dataset preparation for Amharic script.

- The basic idea of Attention mechanism, on neural machine translation, is extended to the OCR of Amharic script where an Attention based encoder-decoder networks recognize text-line images. With this design we achieve a recognition efficiency comparable with CTC-based recurrent networks.

- Propose a novel sequence-to-sequence learning method by blending the concept of Attention mechanism directly into CTC. This technique is evaluated with the test sets of ADOCR database and outperforms other state-of-the-art OCR techniques by a large margin.

- A detail analysis of the results of experiments carried out with different deep learning techniques is presented. Compared to the classical machine learning based OCR approaches, the deep learning based OCR approaches yield the best reported results for Amharic scripts recognition and has fewer preprocessing steps.

1.5 Organization of the Thesis

This thesis is organized into eight chapters. The first chapter provides an overview of the general background of OCR in general and particularly, for Amharic script recognition. The motivation behind the present work, statement of the problem, research objectives, list of publications from this thesis, and the major contributions are also briefly described. In chapter two, we present the trends in OCR research and various methodologies that have been emerged over decades of OCR research. Both traditional and state-of-the-art OCR approaches that were and have been applied in DIA and OCR research are also presented. Specifically, it is noted on sequence-to-sequence learning techniques such as Attention mechanism and CTC-based recurrent networks have been utilized, in this thesis, for Amharic scripts.

Chapter three introduces Amharic languages and presents background information about Amharic, the demographic distribution of its' speakers, the genetic structure of Amharic script, character sets, and writing system. Further, related issues that are hindering the development of a reliable Amharic OCR system and unique features of Amharic script have been described. This chapter also gives a summary of related works and highlights that the unavailability of training datasets is the major obstacle to Amharic OCR research.

Chapter four is about the dataset used in this thesis; then it introduces and describes the details of an Amharic database called ADOCR which contains both printed and synthetically generated images. The dataset is organized into two levels; at the character level and text-line level image. Data collection and synthetic data generation techniques are also described.

Chapter five discusses CNN-based Amharic character recognition models and experimental results. Unlike the usual Amharic character recognition model, characters are recognized based on their grapheme on Fidel-Gebeta employing a multi-task convolutional neural network model and achieved better character recognition results. In this chapter, multi-task learning paradigms, and related research works done on Amharic character recognition are also presented in detail.

Chapters six and seven provide the systems architecture of various sequence-to-sequence learning models. Chapter **6** focuses on using LSTM as a sequence-to-sequence mapping together with CTC. CNN's were also covered and employed as a feature extractor module by stacking before recurrent networks. Further, various recurrent network-based approaches are incorporated in the

learning processes and experimental results are presented to show their applicability and performance in recognition of Amharic text-images.

Attention mechanism has been given more attention in NLP tasks for neural machine translation. Based on the idea of neural machine translation, an Attention-based encoder-decoder network is proposed in Chapter **7**. In the first part of this Chapter, an end-to-end Amharic OCR model is proposed using LSTM networks, without CTC alignment, as encoder and decoder unit. Attention is embedded between the encoder and decoder LSTMs to allow the decoder to attend to different parts of the source sequence and to let the model learn how to generate a context vector for each output time step. This model gives state-of-the-art results on ADOCR test datasets. An extension of this work, a blended Attention-CTC network formulated by directly taking the advantage of Attention mechanism and CTC network, is reported in the second part of this chapter. The performance of this blended attention-CTC network is evaluated against the ADOCR test sets and it performs significantly better than the other sequence-to-sequence models.

Finally, a summary of this thesis, concluding remarks, and guidelines for future work are provided in chapter eight.

1.6 List of Publications

Part of the work presented in this thesis has been accepted and/or presented in peer-reviewed conferences or journals. The list of the publications derived from this thesis work are provide as follows.

Journals

1. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Million Meshesha, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Amharic OCR: An End-to-End Learning". *MDPI-Applied Sciences. 2020; 10(3):1117.*

Conferences and Preprints

1. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Factored Convolutional Neural Network for

Amharic Character Image Recognition", 26^{th} IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 2906–2910.

2. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Amharic Text Image Recognition: Dataset, Algorithm and Analysis", 15^{th} International Conference Document Analysis and Recognition (ICDAR), Sydney, Australia, 20–25 September 2019; pp. 1268–1273.

3. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Gebeyehu Belay, Didier Stricker, "Multi-font Printed Amharic Character Image Recognition: Deep Learning Techniques", 6th EAI International Conference on Advances of Science and Technology: ICAST 2018, Bahir Dar, Ethiopia, October 5-7, 2018, Proceedings, vol. 274, p. 322. Springer, 2019

4. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Didier Stricker, "Amharic character image recognition", 18^{th} IEEE International Conference on Communication Technology (ICCT), Chongqing, China, 8–11 October 2018; pp. 1179–1182

5. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Blended Attention-CTC Network Architecture for Amharic Text-image Recognition, 10^{th} International Conference on Pattern Recognition Applications and Methods (ICPRAM 2021)

6. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Gebeyehu Belay, Million Meshesha, Marcus Liwicki, Didier Stricker, "Learning by Injection: Attention Embedded Recurrent Neural Network for Amharic Text-image Recognition". *MDPI-Preprints 2020*

Chapter 2

Trends in OCR Research and Development

This chapter provides a highlight of the existing methods and trends in OCR technology. Technically, OCR comprises numerous procedures and techniques which have been emerged over a century of research. The rapidly growing and availability of computing power has facilitated the use of sophisticated and diversified approaches for OCR development. The following sections of this chapter have acknowledged multiple approaches of OCR research which can be broadly categorized into two groups. The first category, the classical OCR approach, is commonly called segmentation-based OCR which involves the segmentation of a page into paragraphs, then into text-lines, followed by words, and finally into individual characters. A character recognizer would recognize the segmented characters one by one. Section 2.1 describes various methods developed for this type of OCR approach. The second OCR approach, which is called segmentation-free OCR in literature, and the modern OCR approach in this thesis. Section 2.2 describes this OCR approach in detail. Section 2.3 describes the overview of deep neural networks and section 2.4 presents state-of-the-art and emerging trends of segmentation-free OCR approach and Seq2Seq tasks in general. Since recognition in segmentation free approach is done at the word or at text-line level, and it avoids the task of extracting characters from text-lines and minimizes the efforts required for preprocessing and feature extraction.

2.1 Classical OCR Approach

The classical OCR approach is commonly called the segmentation-based approach. Segmentation is the process of partitioning a digital image into multiple manageable segments in general and specifically for OCR, it is the process of segmenting document images into characters. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. In the segmentation-based OCR approach, the feature extraction and recognition process depend on the quality of segmented parts [Nag00]. Therefore, in this approach, segmentation is the fundamental step for designing an efficient OCR system. However, the following challenges have been considered as the major bottleneck of segmentation task in document image analysis [CL96, Eik93].

- **Over segmentation problem**: Due the nature/complexity of the writing system, touching and fragmented characters being interpreted as one single character, or a part of a character is believed to be an entire symbol.
- **Document scanning problem**: During scanning, joints will occur if the document is a dark photocopy or if it is scanned at a low resolution. The characters may be split if the document is scanned at a high resolution or digitized using a light photocopy.
- **Ambiguity of noise and/or texts**: Understanding dots and diacritics as noise wrongly for and vice versa. As a result, a wrong data or non-texts being sent for recognition. This is mainly happens if characters are geometrically connected to some graphics.

Segmentation-based OCR techniques have been used for years, during this period significant works have been done for recognition of different scripts [PC04, CL96, LV12] and it has been also widely applied for Amharic script recognition [Mes08, AB08, MJ07, CH03] as well. The smallest individual components feed for the final recognition stage, in the segmentation-based OCR technique, are characters. In general, character segmentation is achieved using either explicit or implicit segmentation techniques, where the former segmentation techniques are employed in segmentation-based OCR while the later segmentation technique is the feature of segmentation-free OCR approach [Cho14, HS95, SG97]. The detailed procedures of this segmentation technique are described as follows:

- **Explicit segmentation**: It is a pure segmentation technique in which

the sequence of characters in the input document image is segmented into sub-images of individual characters and is then fed to the character recognizer for classification. The vertical segmentation approach lies in the category of explicit segmentation. In this approach, after the preprocessing stage of the input handwritten/printed word image, the word image is scanned from top to bottom [CRA13]. Algorithms in this segmentation technique follow a cut rule which normally takes a set of heuristics and information of the foreground, background, or a combination of them into consideration to generate potential segmentation cuts. In the vertical segmentation technique, once the potential segmented columns of the image are identified by assigning $0's$ and $1's$ then the cut point of a character is obtained using histogram profile projection. In explicit segmentation, the over-segmentation problem occurred either when the two consecutive characters in the word image are connected by a ligature or when characters are open characters [CRA13, RMS09].

- **Implicit segmentation**: It is commonly called recognition-based segmentation and it uses each pixel column as a potential cut location. In the implicit segmentation approach both segmentation and recognition of characters are achieved at the same time [CdSBJB$^+$06, RMS09]. The implicit segmentation approach split words into segments which should be characters, and then pass each segment to a classifier. If the classification results are not satisfactory, call segmentation once more with the feedback information about rejecting the previous result. The implicit segmentation approach provides all the tentative segments and lets the recognizer decide the best segmentation hypothesis. However, there is a trade-off in selecting the number of segments for a word. Computation becomes efficient if the number of segments is less but wide-character may be covered in the hypothesis. Whereas, for a large number of segments, it is computationally expensive.

  Like explicit segmentation, implicit segmentation is also exposed for over-segmentation. To overcome segmentation problems, the implicit segmentation approach follows a two-stage process. In the first stage, a heuristic rule-based segmentation scheme is applied to segment the words. In the second stage, verification is performed. Then segmentation will be finalized based on recognition [RMS09]. Such segmentation techniques are usually applied in the segmentation-free OCR approach, where CTC network layer is used in the output layer to calculate

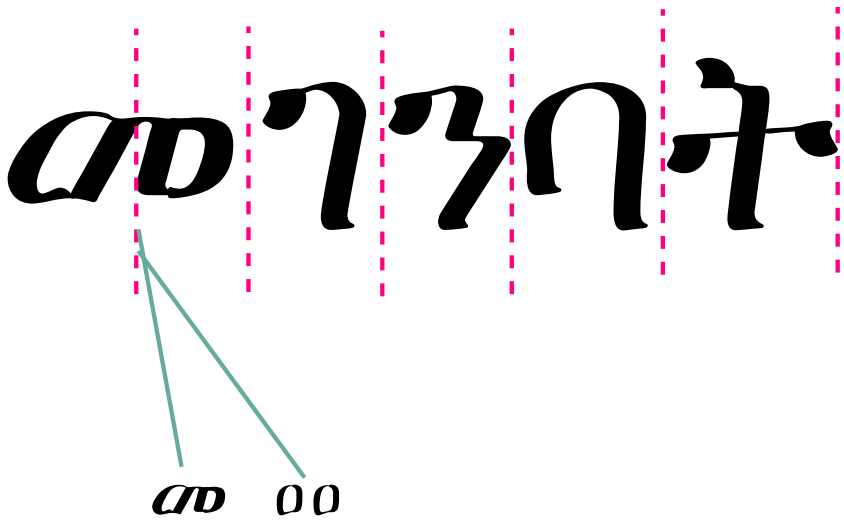


Figure 2.1: **Character segmentation process and segmentation problems on sample Amharic text.** *Character ሙ is over segmented into ሀ ሀ (i.e a single Amharic character pronounced as* me *is considered as two other characters pronounced as* $'a$*).*

the loss and decode the feature sequence.

The majority of errors in segmentation-based OCR-systems are often due to problems in the scanning process and then poor segmentation process resulting in joined or broken characters. Confusion between text and graphics or between text and noise are also the other cause of errors in segmentation-based OCR systems [Eik93]. The sample segmentation problem in Amharic script is illustrated in Figure 2.1. The most common type of segmentation-based OCR model is template matching. In this method, characters are extracted and are subsequently matched against all the possible templates of the characters. The recognition is achieved by performing a normalized cross-correlation between a template character and a new character to be matched [CAC+04]. Comparison between characters is made to detect the highest matching area. This comparison is done by sliding a template character against the new character by one pixel at a time, from left to right and top to down. The value at each point, which tells how similar the template is to that particular area, can be computed using the cross-correlation function f which is the basic similarity measure used in practice and the formulation is given in Equation (2.1)

[UH16].

$$f(x,y) = \frac{\sum_{x,y}[I(x,y) - \bar{I}_{u,v}][T(x-u,y-v) - \bar{T}]}{\sqrt{\sum_{x,y}[I(x,y) - \bar{I}_{u,v}]^2 \sum_{x,y}[T(x-u,y-v) - \bar{T}]^2}} \quad (2.1)$$

where I is the image, T is the template, $\bar{T}$ and $\bar{I}_{u,v}$ is the mean of the template and mean of the region under the template $I(x,y)$ respectively.

In practice, nowadays, the use of segmentation-based methods is quite limited as this method greatly suffers from image noise and font variations. In addition, it is script-dependent and may not work well if we applied it for the segmentation of texts written in any other script. For example, the technique developed for segmenting touched characters in Arabic script may not work well to segment touched characters of texts written in Amharic scripts.

However, the segmentation-based OCR approach has been remained state-of-the-art for a very long time, in the field of OCR, until it is finally surpassed by segmentation-free OCR techniques. The next section presents the details of segmentation-free OCR approaches, which is called the modern OCR approach in this thesis, and highlights the methodologies employed for script recognition under this approach.

2.2 Modern OCR Approach

This section describes modern OCR approach, which is commonly called segmentation-free OCR, in detail. Although segmentation-based OCR approach widely used in literature, segmentation inaccuracy has been the major drawback. Lately, the research in the domain of optical character recognition has moved towards segmentation-free approach that can incorporate contextual information in the recognition process [All95, Nag92]. The modern OCR approaches perform the character localization and recognition at a text-line level; thereby avoiding the need of explicit word or character segmentation. Recognition in segmentation-free OCR approaches does not only depend on the features of a single character rather it also depends on the basis of its surrounding context.

Segmentation-free approaches for OCR, differ from the segmentation-based methods in the level of segmentation required from layout analysis steps. Traditionally such approaches need the extraction of words or part of words for a given text-line. The whole word or part of word are then recognized by

the recognition engine. In segmentation-free OCR approaches, discriminating features are extracted from the whole word or text-line. These features are then used to train a classifier to recognize the whole word or complete text-line. For a long period of time, Hidden Markov Model (HMM) were the state-of-the-art and preferred method [MF06, MSLB98] until the introduction of deep neural networks and it was surpassed by Recurrent Neural Networks (RNNs)-based OCR methods.

2.3 Deep Neural Networks

This section describes deep neural networks, an artificial neural network with multiple layers between the inputs and outputs, in detail. The concept of the deep neural network is originated from Artificial Neural Networks (ANN) [HS06], which are inspired by the structure and function of the biological brain. Until the birth of deep learning [HOT06], ANN and other statistical classifiers such as Support Vector Machines (SVM), K-Nearest Neighbor (KNN) [HM16], and Random Forests (RN)[Pal05] have been adopted and widely implemented for various applications. Those classical machine learning techniques, in general, consists of two modules; the feature extractor and the classifier module. The feature extractor is responsible for the extraction of discriminant features, usually called handcrafted features, from the given data. Once those features are extracted, they are subsequently fed into the classifiers. However, after the introduction of deep neural networks such as Convolutional Neural Networks (CNNs) such stepwise procedures are put to one unified framework, where both the feature extractor and classifier modules are trained in an end-to-end fashion using back-propagation [LFZ+15, SJS13].

2.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a specialized artificial neural network which is designed with the ability to process and extract spatial information from image data [LBBH98]. There are various architectures of CNNs such as, LeNet [LBBH98], AlexNet [KSH12], GoogLeNet[SLJ+15], ResNet [HZRS16], and VGGNet [SZ14], which basically consist of 2 layers in common namely convolutional layers, and pooling layers. Convolutional layers convolve around the input image with multiple convolutional filters, usually performs a dot product between filters and local regions, and pass the output

to the next layer, while pooling layers are responsible to reduce the spatial size of the input [PZLL20].

In addition to these two basic layers, CNNs may consist of fully connected layers which take the flattened matrix from previous layers and compute the class score and they may also consist of various network hyper-parameters such as stride, padding, and depth (number of filters) which could control the learning algorithms [BSS17]. Convolutional neural networks has been successfully applied for various computer vision applications such as, image recognition [TT18], facial recognition [HML+18], image segmentation [YNDT18], and character recognition [BHL+19b]. CNNs are the dominant methods in various image-based tasks and they still continue to attract the interest of the machine learning community across many fields including researchers from document image analysis and recognition.

2.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are designed to address the temporal behavior of input data, which is not easily handled by convolutional neural networks. The intuition of using RNNs for sequential data is the ability of recurrent networks to learn and remember over long sequences of inputs. In sequence-based tasks such as handwritten recognition [LGF+07], speech processing [GMH13], and machine translation [LPM15], we usually need to know what was in the past and what is coming next to better understand and detect the present context.

There are various architectures of recurrent neural networks ranging from a typical vanilla recurrent network, which usually suffers from a problem called exploding and vanishing gradient [RTAS20], to advanced RNN architectures such as Long Short-term Memory (LSTM) and Gated Recurrent Unit (GRU) network, that use gates to control information flow and remedy the gradient problem of traditional RNN. Recurrent networks are trained by back-propagation though time [DJH01, Wer90], where the loss *(*l) is computed based on the loss at every time step as,

$$l = \sum_{i=1}^{T} l(\hat{y}_i, y_i) \tag{2.2}$$

where T, $\hat{y}$, y denotes the maximum time-step, prediction and ground-truth.

For the input variable x at time step t, x_t, the output of the RNN layer at

time-step t can be computed as,

$$O_t = h_t W_o + b_o \tag{2.3}$$

where O_t is the output, W_o is the weight at the output layer and b_o bias and h_t is the hidden state variable at time-step t which can be expressed as follows,

$$h_t = f(x_t W_{xh} + h_{t-1} W_h + b_h) \tag{2.4}$$

where f is an activation function, W_{xh} and W_h weights at current time-step(t) and previous time-step $(t-1)$, h_{t-1} is the hidden state at $(t-1)$ and b_h is the bias term.

To process data both spatially and temporally, researchers recommended using a hybrid of CNNs and RNNs. This combination has been successfully applied in many domains including image captioning and visual question answering [WSW+17], handwritten recognition [ZDD18]. The next section presents the two state-of-the-art techniques for sequence-to-sequence tasks from the perspective of optical character recognition.

2.4 Sequence-to-sequence Learning for OCR

A more recent approach in the segmentation-free OCR domain is attention-based sequence-to-sequence learning. However, the use RNNs in combination with CTC approach are also widely applied to date. These two sequence-to-sequence learning approaches are the main focus, of this thesis, and the details are discussed in the next sections. Section 2.4.1, presents the details of CTC-based sequence learning and attention-based sequence learning is described in section 2.4.2.

2.4.1 CTC-based Sequence Learning

Connectionist Temporal Classification (CTC), introduced by Graves [GFGS06], is a type of neural network output and associated scoring function, for training RNNs such as Long Short-Term Memory (LSTM) networks to tackle sequence alignment problems where the timing is variable. CTC-based supervised training of RNNs, on sequential data, has shown great success in many machine learning areas including end-to-end speech recognition [ZYDS17] and handwritten character recognition [LGF+07].

In CTC-based sequence learning, the input is a sequence of observations, and the output is a sequence of labels, which can include a blank token as an additional outputs [GFGS06]. For a given training data D, the CTC objective function minimizes the negative logarithm of likelihood loss formulated as Equation (2.5).

$$l_{CTC} = -log \sum_{\pi} p(\pi/x) \tag{2.5}$$

where $x = x_1, x_2, ..., x_{\mathrm{T}}$ is the input sequence with length T, $z = z_1, z_2, ..., z_{\mathrm{C}}$ is the corresponding target for $C < T$ and $(x, z) \in D$. $p(\pi_t/x)$ is computed by multiplying the probability of labels along the path π that contains output label over all time steps t as shown in Equation (2.6).

$$p(\pi/x) = \prod_{t}^{T} p(\pi_{\mathrm{t}}, t/x), \tag{2.6}$$

where t is the time step and π_{t} is the label of path π at t. CTC is independent of the underlying neural network structure rather it refers to the output and scoring. The difficulty of training comes from there being many more observations than the actual labels. For example in a text-line, there can be multiple time slices of a single character. Since we don't know the alignment of the observed sequence with the target labels, we predict a probability distribution at each time step [Han17] and then the CTC is used to align the output activation of a neural network with target labels.

A target label in path π is obtained by mapping reduction function B that converts a sequence of Soft-max output for each frame to a label sequence, which will be less than or equal to the length of the frame, by first removing repeated characters and then blank (ϕ) tokens of the given sequences. As an example, for a give Eighteen sequence length observation (o), $o = \phi aa\phi mm\phi h\phi aa\phi rr\phi ic\phi$. Then, the paths are mapped to a label sequence as $l_{\mathrm{s}} = B(o) = B(\phi aa\phi mm\phi h\phi aa\phi rr\phi ic\phi) = B(\phi a\phi m\phi h\phi a\phi r\phi ic\phi)$ = 'amharic', where B is a reduction mapping function which works by first collapse the repeated tokens and then remove blanks.

The target sequence probability y from input sequence x is the sum of the probability of all paths by reducing each path to this label sequence using B and this probability is again obtained using Equation (2.7).

$$p(y/x) = \sum_{\pi \in B(y)} p(\pi/x), \tag{2.7}$$

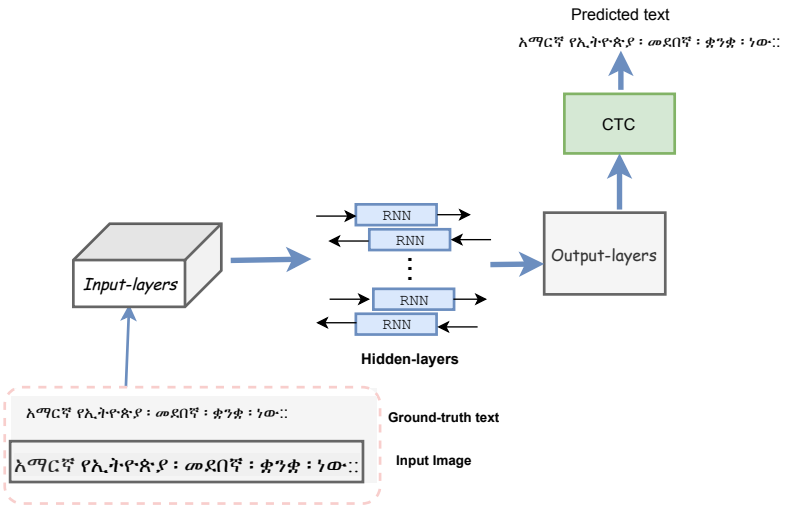


Figure 2.2: **Illustration of CTC-based sequence learning.** *In CTC-based sequence learning paradigm, the CNNs and RNNs are trained together in a supervised manner using the whole text-line image and the corresponding ground-truth sequence. The output activations are aligned with the ground-truth using CTC algorithm. Then the CTC network is responsible for error calculation and transcription.*

Figure 2.2 shows general flow diagram of CTC-based sequence learning paradigm, that uses Bi-LSTM as a sequence learner and CTC cost of target transcription, for an input sequence of sample Amharic text-line image.

2.4.2 Attention-based Sequence Learning

Attention has been used in different contexts across multiple disciplines, for example, psychologists define Attention as the behavioral and cognitive process of selectively concentrating on a discrete aspect of information while ignoring other perceivable information [Jam07]. Human visual attention, Biologically, allows us to focus on a certain region with high resolution while perceiving the surrounding in low resolution and then adjust the focal point or do the inference accordingly [All89]. In Computing, Attention is one of the components of the network architecture and is in charge of managing and quantifying the interdependence of sequences [BCB14].

Attention is, to some extent, motivated by how we pay visual attention to different regions of an image or correlate words in a sentence. It is first intro-

duced, by Bahdanau [BCB14], as a solution to the limitation of the encoder-decoder model in sequence transcription. The architecture of encoder-decoder networks has two modules, the encoder, and the decoder. Basically, both the encoder and the decoder modules can be any variant of the neural network, but they usually use recurrent neural networks, where the role of the encoder module is to transform the given variable-length input sequence into a fixed-length sequence vector and then the decoder module uses these vector so as to generate the target output sequences.

To generate the output sequence from a given input sequence, the encoder and decoder modules are unified and process the information from end-to-end. All the information of the input sequence goes to the decoder module through the last hidden state of the encoder module which usually causes a bottleneck on this hidden state. It was a challenge for encoder-decoder models until the introduction of the attention mechanism, which advances the working principle of the encoder-decoder model and makes it the most competitive model in sequence-to-sequence problems to date.

Attention-based network models have emerged as an extension of the existing encoder-decoder model so as to solve the problems that occur during decoding long sequences. Nowadays, various attention models are proposed for solving various sequence-to-sequence tasks. This section introduces the two most commonly used attention models, based on work of Bahdanau [BCB14], Xu [XBK+15], and Luong [LPM15], namely Soft/global and Hard/Local attention respectively. The differences, in these attention models, lie mainly in their system architectures, scoring functions, and computations. The details of these commonly used attention mechanisms are discussed as follows.

- **Soft Attention**: This attention considers the entire input state spaces of the encoder to drive the context vector and it is usually referred to as *Global* attention [XBK+15, LPM15] and/or *Additive* attention [BCB14]. As explained in the work of Bahdanau [BCB14], the intuition of implementing attention was with the aim of improving the sequence-to-sequence model in machine translation by aligning the decoder with the relevant input sentences. The soft-attention mechanism uses *Concat*, as an alignment scoring function, whereby the decoder hidden state is added to the encoder hidden states. Since soft/global attention attends all the encoder hidden-states, it is computationally expensive, especially for longer input sequences.

- **Hard Attention**: The second type of attention which is proposed to address the limitations of global [LPM15] or soft [XBK+15] attention

and it is often referred to as *Multiplicative* attention or *Local* attention [LPM15]. Unlike the soft attention, hard attention attends part of an input sequence. The local attention proposed by [LPM15] consists a blended features of both soft and hard attentions of [XBK+15] and, therefore, it avoids the expensive computations experienced in attention model of [BCB14, XBK+15]. In addition to the alignment score function, *Concat* used in *Additive* attention, this attention mechanism [LPM15] introduces two other alignment score functions the *Dot* and *General*. The *Dot* alignment score function produces the alignment score by multiplying the hidden states of the encoder and the hidden state of the decoder. Except the weight matrix added in the formulation, the *General* alignment score function is similar to the *Dot* alignment function.

As stated in the literature, the way that the alignment score is calculated and the position at which the attention mechanism is being introduced in the decoder are the major differences in various attention mechanisms. For example, in Luong's local attention, *Concat* alignment score function, the decoder hidden state and encoder hidden state will not have their own individual weight matrix, but a shared one instead, unlike in the attention model proposed by Bahdanau. Therefore, the decoder hidden state and encoder hidden states are added together first before being passed through a linear layer, and then, after being passed through the Linear layer, a tanh activation function will be applied to the output before being multiplied by a weight matrix to produce the alignment score. A generic flow of Attention-based sequence learning, which is designed based on the work of Bahdanau [BCB14], is shown in Figure 2.3.

The sequence-to-sequence learning was born in the field of language modeling [SVL14] that takes a sequence of inputs and then outputs another sequence of items where both sequences can be of arbitrary lengths. The encoder-decoder architecture for recurrent neural networks is proving to be powerful on various sequence-to-sequence prediction problems such as machine translation and caption generation. However, a fixed-length context vector design of this model is a critical disadvantage. It has forgotten the first part once it completes processing the whole input specifically on long input sequences.

Therefore, the attention mechanism was originally introduced to help memorize long source sentences in Neural Machine Translation (NMT) [BCB14]. Rather than building a single context vector out of the encoder's last hidden state, the advantage of Attention is to create shortcuts between the context vector and the entire source inputs. The weights of these shortcut connections

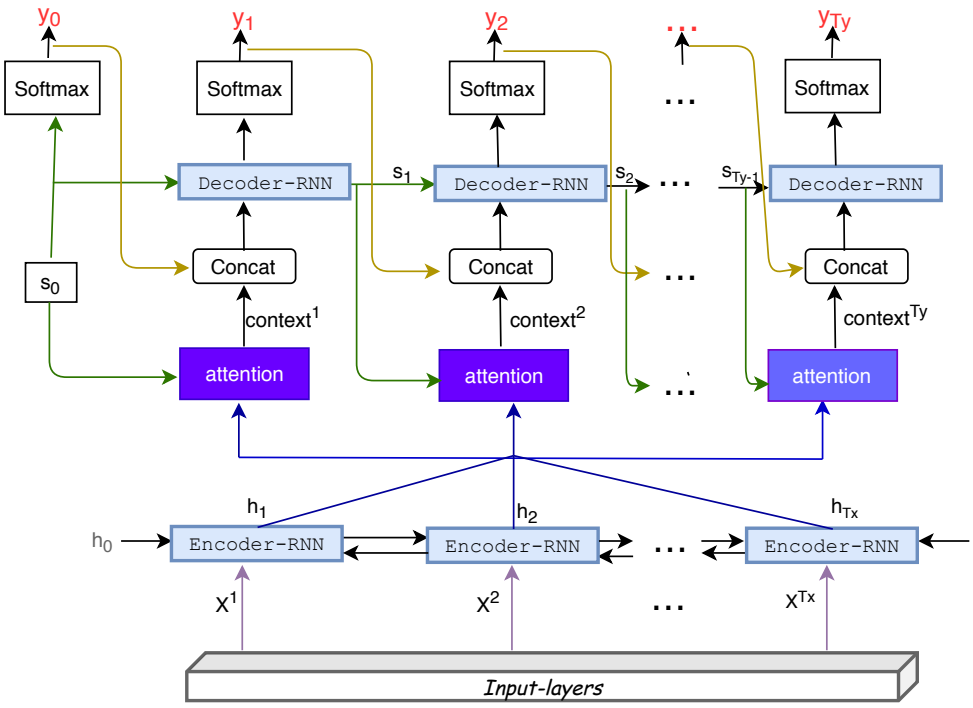


Figure 2.3: **Illustration of Attention-based sequence learning.** *In attention-based sequence learning paradigm, the two RNNs (Encoder and Decoders) are employed. An encoder processes the input sequence $(x^1, ..., x^{T_x})$ with length T_x and compresses the information into a context vector $(context^i)$ of a fixed length. This representation is expected to be a good summary of the meaning of the whole input sequence while the decoder is initialized with the context vector to emit the transformed output sequence ($y_1, ..., y_{T_y}$) of length T_y.*

are customizable for each output element. Since the context vector has access to the entire input sequences, we don't need to worry about forgetting. The alignment between the source and the target is learned and controlled by the context vector.

Attention-based sequence-to-sequence learning employees two recurrent networks; the Encoder-RNN encodes the input sequence and outputs a sequence of the same length vector while the Decoder-RNN interprets the encoded fixed length vector and generates a target sequences. Let's define a generic Attention-based encoder-decoder network model, which is introduced for solving a sequence-to-sequence tasks specifically focus on NMT. For an input sequence, X =$[x_1, x_2, ..., x_{Tx}]$, of length T and an output target sequence, Y =$[y_1, y_2, .., y_{Ty}]$, with different length of t. Suppose the encoder is

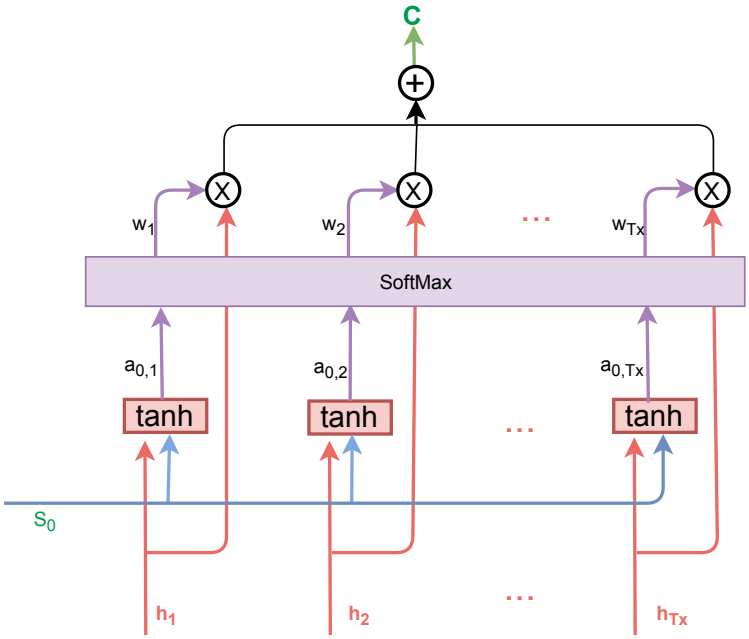


Figure 2.4: **A typical attention model.** *This design computes the output c from the initial decoder hidden state s_0 and the part of the given sequence h_i, where the decoder hidden states at $t-1$ are first concatenated with each encoder hidden states and passes through $tanh$ activation. The $tanh$ layer can be replaced by any other network that is capable to produce an output form s_0 and h_i. Here, w_i and a_i denotes attention weights and alignment scores respectively.*

a bidirectional RNN with a forward hidden states $\overrightarrow{h}_{\mathrm{i}}$ and a backward hidden states $\overleftarrow{h}_{\mathrm{i}}$. With the motivation to include both the preceding and following words in the annotation of one word, the encoder state h_{i} is the concatenation of the two hidden states and represented as, $h_{\mathrm{i}} = [\overrightarrow{h}_{\mathrm{i}}, \overleftarrow{h}_{\mathrm{i}}]$ for $i = 1, .., T$.

The decoder network has hidden state $s_{\mathrm{j}} = f(s_{\mathrm{j\text{-}1}}, y_{\mathrm{j\text{-}1}}, c_{\mathrm{j}})$ for the output sequence at position $j, j = 1, ..., t$, where the context vector $c_{\mathrm{j}} = \sum_{i=1}^{Tx} s_{\mathrm{ji}} h_{\mathrm{i}}$, is a sum of hidden states of the input sequence, weighted by alignment scores, $a_{\mathrm{ji}} = Align(y_{\mathrm{j}}, x_{\mathrm{i}})$, which shows how well two sequences y_{j} and x_{i} are aligned. The alignment score a_{ji}, as described in the work of Bahdanau [BCB14], is computed by a feed-forward network with $tanh$ activation function which is jointly trained with other parts of the model. Then the alignment score is given as, $score(s_{\mathrm{j}}, h_{\mathrm{i}}) = v_a^T tanh(W_a[s_{\mathrm{j}}; h_{\mathrm{i}}])$, where both v^{a} and W^{a} are weight matrices to be learned in the alignment model. A typical attention layer architecture that computes a context vector for the first decoding time step is

depicted in Figure 2.4. Even though the Attention mechanism is designed and has been used to solve problems in NMT, nowadays, it is also widely applied in various fields such as image captioning [LTD+17], visual question-answer [AHB+18], and OCR [ZDD18, PV17, CV18] and promising experimental results are reported.

2.5 Chapter Summary

Trends in OCR research and widely used character recognition methodologies are highlighted in this chapter. Research in the OCR model development can be either segmentation-based or segmentation-free. Since segmentation-based OCR depends on the segmentation stage which is usually affected by many factors, (like the nature of the scripts and quality of documents), the segmentation-free approach is the choice and state-of-the-art techniques used to develop the OCR model. The segmentation-free OCR approach can be either a CTC-based or attention-based sequence learning. The CTC-based networks have been used to tackle sequence-to-sequence problems where the timing is variable. In this approach, the current output of the CTC model relies on the current input and does not account for context weighted inputs and it assumes that each output is conditionally independent of the other outputs. Therefore, attention-based sequence-to-sequence learning has revolutionized and considers a standard fixture in most state-of-the-art NLP models. Attention-based sequence learning focuses on specific sequences and considers context-weighted inputs when formulating a response, unlike in CTC-based sequence learning. Nowadays, attention-based networks are widely used in solving problems in the field of OCR. The research attempts made to develop Amharic OCR model so far are segmentation-based and there is no workable OCR of this language. Thus, in this thesis, such a research gap is further investigated by employing state-of-the-art, segmentation-free, sequence-to-sequence learning techniques. In addition to solving segmentation-related challenges, the existing sequence-to-sequence approaches are extensively investigated and then the best features from the CTC and attention-based techniques are integrated for an improved feature representation and output sequence generation.

Chapter 3

Amharic Script and Writing System

This chapter provides an overview of the script, Amharic, which we are going to deal with in this thesis. Amharic is an exotic script that has its own indigenous alphabet and a rich collection of documents. Understanding the language and its writing system are indispensable steps in Natural Language Processing (NLP) tasks in general. OCR application heavily depends on the writing system of the language. This chapter first presents the demographic distribution of Amharic speakers and the genetic structure of Amharic script. Furthermore, the writing system and orthographic identity of Amharic characters with their corresponding unique features, in the script, have been presented. This chapter also provides a brief description of related works, on Amharic script, which has been done prior and/or in parallel with this thesis. Finally, the summary of the chapter is presented.

3.1 Amharic Script and its Genetic Structure

In Ethiopia, there are more than 86 languages, that are divided into Semitic, Cushitic, Omotic, and Nilo-Saharan groups [EF20], with up to 200 different dialects spoken [MJ05, Mes08]. The languages are written with either Ge'ez/Amharic and/or Latin scripts. Of the 86 languages in Ethiopia, 41 are living languages at the institutional level, 14 are developing, 18 are vigorous, 8 are in danger of extinction, and 5 are near extinction [EF20]. Of these languages, Amharic is an official working language of the Federal Democratic Republic of Ethiopia and is spoken by more than 50 million people as their mother language and over 100 million as a second language in the

Figure 3.1: **Demographic distribution of Amharic speakers.** *Amharic is widely spoken in different countries while a country marked with yellow color is a nation which uses Amharic as an official working language*[3].

country [pre, MM18]. The demographic distribution of Amharic speakers is illustrated in Figure 3.1. In addition to Ethiopia, it is also widely spoken in other countries like Eritrea, USA, Israel, Sweden, Somalia, and Djibouti [MJ07, Mey06].

The script of Amharic language is originated from the Ge'ez script, an ancient Ethiopian script, around 300 A.D. Amharic script consists of the full set of Ge'ez characters and includes its own additional characters. Nowadays, Amharic characters are used to write various languages in Ethiopia including Amharic, Guragegna, Awi, Argoba, and Tigrigna. Figure 3.2 depicts the

[3] `https://www.adams.africa/works/ethiopia/`

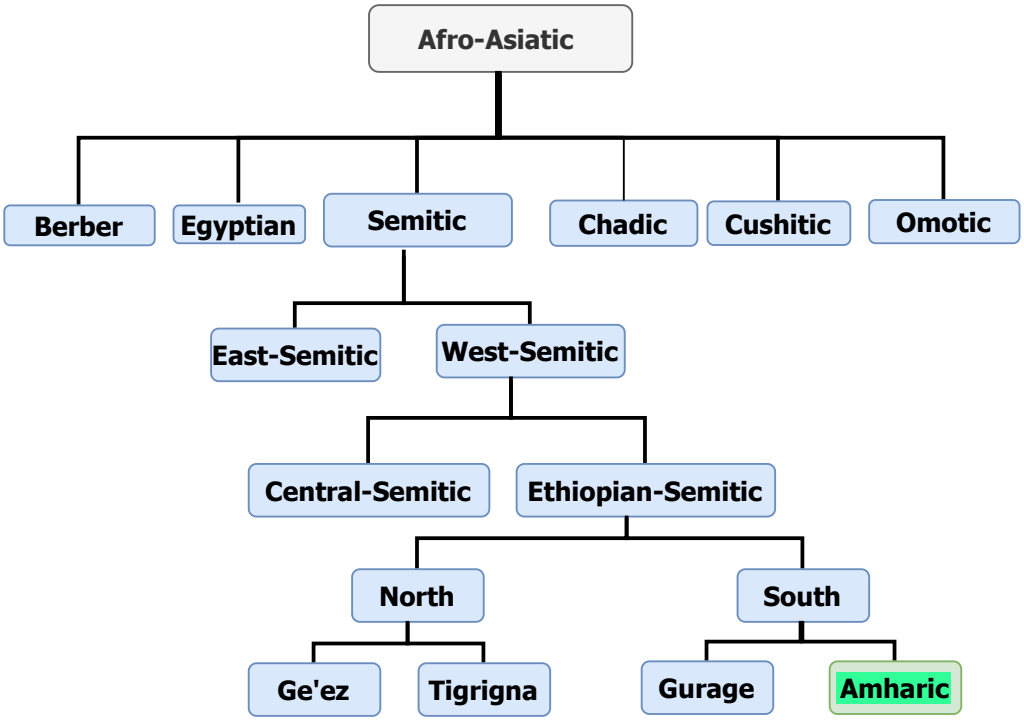


Figure 3.2: **Family of Amharic language.** [4]

genetic structure of Amharic language. Amharic script is designed as a meaningful and graphic representation of African knowledge systems which is considered as one of the signal contributions made by Africans to world history and cultures. It is created to holistically symbolize and locate the cultural and historical parameters of the Ethiopian people [Asn12].

3.2 Orthographic Identity of Amharic Characters

Writing systems can be placed into different categories such as alphabetic, syllabic, or logographic [PC00, Mes08]. In addition, any particular writing system may have also attributes of more than one category [Cou03]. In the alphabetic category, a standard set of letters formed from consonant and vowel combinations represent speech sounds (e.g. Latin alphabet). In a syllabic/Abugida, each symbol correlates to a syllable (consonant-vowel pairing) in which consonants are written as the main letters, and then special symbols/diacritics are used to indicate which vowels follow the consonant (e.g.

[4] Obeid, Nassim. "Etymological study of semitic languages." (2009).

Amharic and Devanagari characters). In a logography, each character represents a semantic unit such as a word or morpheme (e.g Chinese character).

Generally, alphabetic scripting typically uses a set of fewer than 100 symbols to fully express a language, whereas syllabic can have several hundred, and logographies can have thousands of symbols [Cou03]. Many writing systems also include a special set of symbols known as punctuation which is used to aid interpretation and help capture variations in the message's meaning that are communicated verbally by cues in timing, tone, accent, or inflection [Aro85].

Amharic uses the writing system called Fidel, which grew out of the Geez abugida, and it consists of about 317 different characters which are written and read, as in English, from left to right in horizontal lines [BHM+20, MJ07]. In Amharic writing system, each symbol represents a syllable consisting of combinations of consonants and vowels as a single alphabet. The basic signs are modified in a number of different ways to indicate the various vowels from their corresponding consonants [BHL+19b].

All vowels and labialized characters in Amharic script are derived, with a small change, from the 34 consonant characters. The changes involve a considerable regularity for modifying the structure of consonant characters by adding a straight line, or shortening and/or elongating one of its main legs. It also includes the addition of small diacritics, such as strokes or loops/circles to the right, left, top, or bottom of the character. The 34 consonant Amharic characters and their corresponding six vowel variants of each character are illustrated in Figure 3.3.

In addition to the 238 basic characters, Amharic script uses nearly 79 other characters which are depicted in Figure 3.4 includes 50 labialized characters, 9 punctuation marks, and 20 numerals [BHM+20, HA13]. This brings the full sets of Amharic Fidel used in the Script to 317 characters. A matrix that consists of all these Amharic characters is called Fidel-Gebeta Unlike vowels which are derived with a regular pattern modification of consonant Amharic characters, Labialized Amharic characters are derived with less regular patterns modification. The modern Amharic writing system particularly uses few of Amharic punctuation marks like full stop (።), Comma (፣) and preface colon (፦) [Ale97]. Others have largely fallen into disuse and a number of punctuation marks, such as exclamation mark ! and the question mark ?, have been adopted from the Western practices [Tes10]. Amharic numbering system represents numbers from one to ten, multiples of ten (twenty to ninety), hundred, and thousand. However, Amharic numbering system has long since

	0	1	2	3	4	5	6
0	ሀ hā	ሁ hu	ሂ hī	ሃ ha	ሄ hé	ህ hi	ሆ ho
1	ለ lā	ሉ lu	ሊ lī	ላ la	ሌ lé	ል li	ሎ lo
2	ሐ ḥā	ሑ ḥu	ሒ ḥī	ሓ ḥa	ሔ ḥé	ሕ ḥi	ሖ ḥo
3	መ mā	ሙ mu	ሚ mī	ማ ma	ሜ mé	ም mi	ሞ mo
4	ሠ śā	ሡ śu	ሢ śī	ሣ śa	ሤ śé	ሥ śi	ሦ śo
5	ረ rā	ሩ ru	ሪ rī	ራ ra	ሬ ré	ር ri	ሮ ro
6	ሰ sā	ሱ su	ሲ sī	ሳ sā	ሴ sé	ስ si	ሶ so
7	ሸ šā	ሹ šu	ሺ šī	ሻ šā	ሼ šé	ሽ šei	ሾ šo
8	ቀ qā	ቁ qu	ቂ qi	ቃ qā	ቄ qé	ቅ qi	ቆ qo
9	በ bā	ቡ bu	ቢ bī	ባ bā	ቤ bé	ብ bi	ቦ bo
10	ተ tā	ቱ tu	ቲ tī	ታ tā	ቴ té	ት ti	ቶ to
11	ቸ čā	ቹču	ቺ čī	ቻ čā	ቼ čé	ች či	ቾ čo
12	ኀ hā	ኁ hu	ኂ hī	ኃ hā	ኄ hé	ኅ hi	ኆ ho
13	ነ nā	ኑ nu	ኒ nī	ና nā	ኔ né	ን nei	ኖ no
14	ኘ ñā	ኙ ñu	ኚ ñī	ኛ ñā	ኜ ñé	ኝ ñi	ኞ ño
15	አ ʾa	ኡ ʾu	ኢ ʾī	ኣ ʾā	ኤ ʾé	እ ʾe	ኦ ʾo
16	ከ kā	ኩ ku	ኪ kī	ካ kā	ኬ ké	ክ ki	ኮ ko

	0	1	2	3	4	5	6
16	ከ kā	ኩ ku	ኪ kī	ካ kā	ኬ ké	ክ ki	ኮ ko
17	ኸ xā	ኹ xu	ኺ xī	ኻ xā	ኼ xé	ኽ xi	ኾ xo
18	ወ wā	ዉ wu	ዊ wī	ዋ wā	ዌ wé	ው wi	ዎ wo
19	ዐ ʿa	ዑ ʿu	ዒ ʿī	ዓ ʿā	ዔ ʿé	ዕ ʿe	ዖ ʿo
20	ዘ zā	ዙ zu	ዚ zī	ዛ zā	ዜ zé	ዝ zi	ዞ zo
21	ዠ žā	ዡ žu	ዢ žī	ዣ žā	ዤ žé	ዥ ži	ዦ žo
22	የ yā	ዩ yu	ዪ yī	ያ yā	ዬ yé	ይ yi	ዮ yo
23	ደ da	ዱ du	ዲ dī	ዳ dā	ዴ dé	ድ di	ዶ do
24	ጀ ǧā	ጁ ǧu	ጂ ǧī	ጃ ǧā	ጄ ǧé	ጅ ǧi	ጆ ǧo
25	ገ gā	ጉ gu	ጊ gī	ጋ gā	ጌ gé	ግ gi	ጎ go
26	ጠ ṭā	ጡ ṭu	ጢ ṭī	ጣ ṭā	ጤ ṭé	ጥ ṭi	ጦ ṭo
27	ጨ čā	ጩ ču	ጪ čī	ጫ čā	ጬ čé	ጭ či	ጮ čo
28	ጰ pā	ጱ pu	ጲ pī	ጳ pā	ጴ pé	ጵ pi	ጶ po
29	ጸ ṣā	ጹ ṣu	ጺ ṣī	ጻ ṣā	ጼ ṣé	ጽ ṣi	ጾ ṣo
30	ፀ šā	ፁ šu	ፂ šī	ፃ šā	ፄ šé	ፅ ši	ፆ šo
31	ፈ fā	ፉ fu	ፊ fī	ፋ fā	ፌ fé	ፍ fi	ፎ fo
32	ፐ pā	ፑ pu	ፒ pī	ፓ pā	ፔ pé	ፕ pi	ፖ po
33	ቨ vā	ቩ vu	ቪ vī	ቫ va	ቬ vé	ቭ vi	ቮ vo

Figure 3.3: **List of basic Amharic characters.** *Row-column-wise* (34×7) *order, where half of the characters are give on the left and the other halves are given on the right side of this figure. writing and reading are from left to right: The first column gives the consonant characters while the next six columns are derived by combining the consonant with vowel sounds. Across all the seven orders, the shapes are constrained by the vowels they represent.*

been retired to a reserved use primarily for calendar dates and sometimes for paging. As stated, in literature, Amharic writing system does not have a symbol for zero, negative, decimal point, and mathematical operators. Therefore, the Arabic numerals are used everywhere else while Latin mathematical operators are used for arithmetic computation.

3.2.1 Features of Amharic Characters

Since a character in Amharic script is formed from its corresponding consonant by adding diacritics, most of the characters have some common notable characteristics such as shape similarity, vowel formation, structural relationship, height, and width difference. Each symbol is written according to the sound that goes with it. The vowels are integrated into the characters by modifying the base characters in some form, which together represent syllable combinations consisting of a consonant and vowel. Thus Amharic writing system is often called syllabic rather than alphabetic. Some of the features of Amharic characters could be described as

- The direction of writing and reading the characters is from left to right and top to bottom, where characters are written in a disconnected manner and proportional spacing.

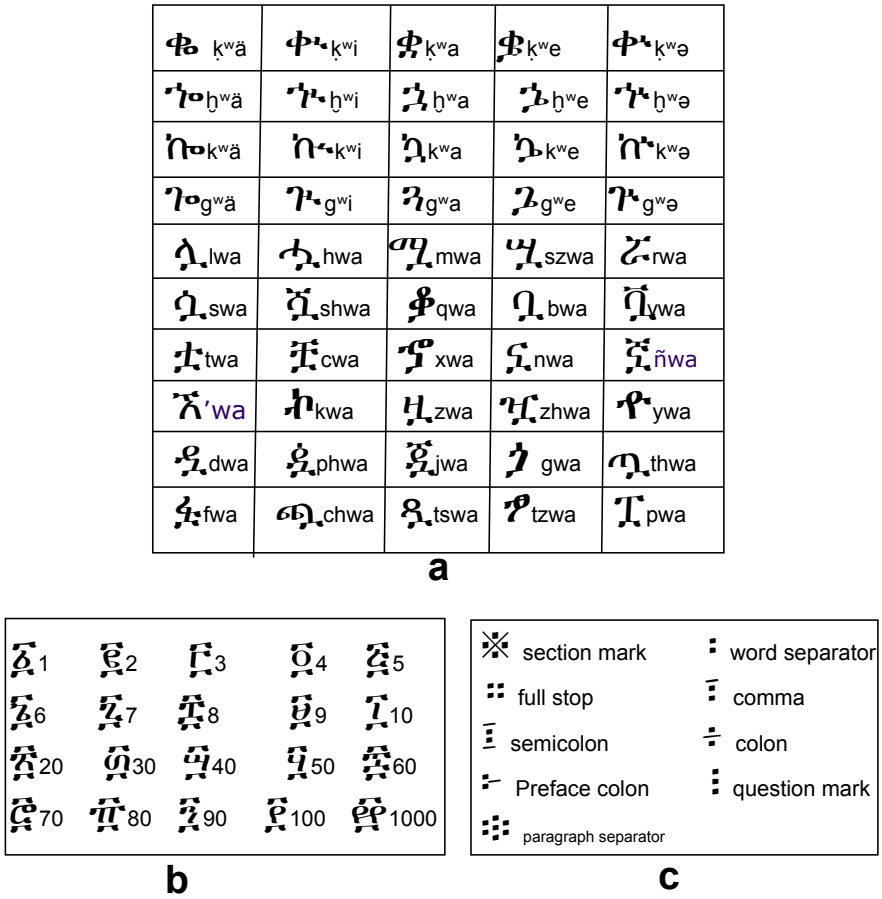


Figure 3.4: **Other symbols of Amharic Fidel.** *(***a***) Labialized characters. (***b***) Numerals and (***c***) Punctuation marks. In most moder Amharic documents, Amharic numerals are replaced by Latin numbers while some of the punctuation marks such as section mark and paragraph separators are not used any more.*

- There is a space between words which usually either a blank space or an Ethiopic two dots/word separator(፡). Sometimes, in early printed books, a word separator followed by blank space can be used together (see Figure 3.5).

- Vowel characters are derived from the consonants, with a small modification of their shapes. For example, as shown in Figure 3.3, the 1^{st} column consists of the basic symbol of Amharic characters with no explicit vowel indicator which is usually called consonants. In the 2^{nd} and 3^{rd} columns the corresponding consonants are modified by adding a hyphen (-) like diacritics in half-way down the right leg and the base of the right leg of the character respectively. The 4^{th} column has a short

left leg while the 5^{th} column has a loop/circle on the right leg. The 6^{th} and 7^{th} columns are less structured and lack consistency, but some regular modification has been made on either the left or right leg of the base characters. Otherwise, some other modifications such as loops or small diacritics could also be added to the top of the characters.

- Sentences are separated with Ethiopic four dot/full-stop (።) while the paragraphs are separated with a full line of horizontal space or can be started with indentation.
- Unlike the Latin characters, there is no capital and lower case distinction among Amharic characters.
- There is a noticeable variance in width and height among characters, some of them such as character ሰ, ደ and ጀ are longer than character ሙ, ሠ and ወ vertically, while the later characters are longer than the former characters horizontally.

In addition, the line of printed Amharic script lies at the same level, having no ascent and descent features which always have a white line between two consecutive lines of characters. Sample Amharic scripts, including some of the features stated above, are illustrated in Figure 3.5.

3.3 Challenges for the OCR of Amharic Script

Even though there is a great demand for Amharic OCR, research in this field is still behind that of the well-known scripts, such as Latin, and even it is the untouched one. Amharic script is much more complex as compared to Latin scripts. The character shapes in Amharic script are far too many and vary widely with different non-standard fonts. Since Amharic is an indigenous language with its own unique characters and writing system, understanding the characteristics and distinctive features of this script is challenging and also interesting because it has an enormous use for the development of optical character recognition system for Amharic script. Some of the generic and script/language-specific challenges are outlined below:

- **Visual similarity among characters**: This is due to shape similarity between consonant characters and also across vowels which are derived from the base characters with small modifications. Since characters in Amharic script are derived from 34 basic characters, there is a high

[5] https://www.bbc.com/amharic/news-51780956

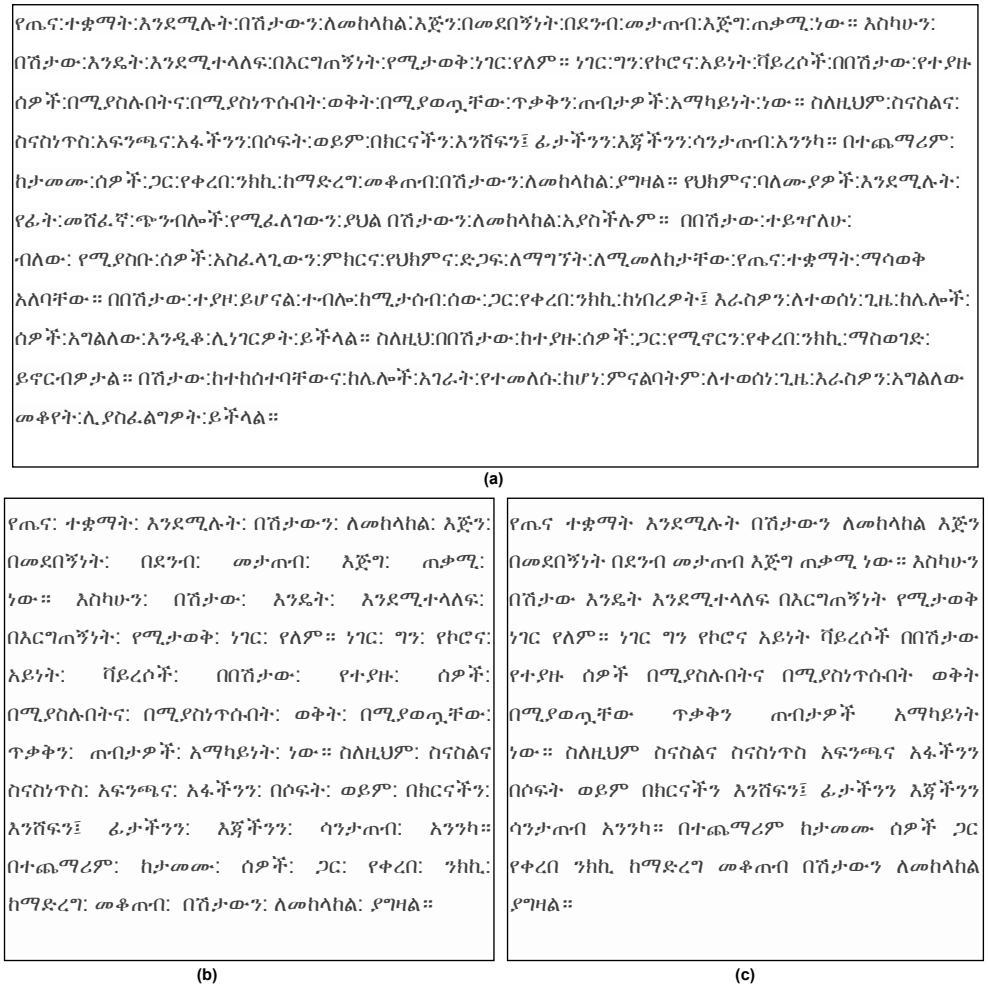


Figure 3.5: **Sample Amharic scripts.** *(***a***) Word are separated with two dots. (***b***) Words are separated by two dots followed by blank space. (***c***) Words separated with blank space. The texts are obtained from BBC Amharic news*[5].

chance of being visually similar; thus the task of recognition becomes hard for machines as well as humans.

- **Character formation inconsistency across columns**: Even though, there is considerable shape modification regularity of characters across some columns, some of them are more consistent than others while others are completely inconsistent. Some of them, such as the 3^{rd} and 5^{th} column are more consistent than others, such as the 2^{nd} and 4^{th} columns, while others, such as the 6^{th} and 7^{th} columns are completely inconsistent. The 6^{th} column is the least consistent, with the greatest number of patterns, so that the character formation is largely unpre-

dictable although in the entire Amharic Fidel some clusters of Amharic characters follow similar patterns modification to the 6^{th} column.

- **Printing quality and font variations**: Printed documents are also written using various fonts, styles, and sizes with different quality of printing materials such as ink, printing paper, and even the printing device it self. As a result of these features, the shape and appearance of characters vary substantially. For printing in a specific Amharic script, there are different fonts (for example, *Power Geez*, *Visual Geez*, *Nyala*, and *Agafari*) with several stylistic variants such as *Normal*, *Bold*, and *Italic*. In addition, different font sizes including 10, 12, 14, 16 pixel [Mes08]. Since these fonts, styles, and sizes produce texts that greatly vary in their appearances, it is a challenging task for developing character recognition systems. However, these challenges are independent of the scripted nature rather they are incurred due to either font variation or the printing quality.

- **Language-specific challenges**: Since indigenous languages of Africans are underrepresented in the area of Natural Language Processing (NLP), most of them are resource-limited. As a result, they face many additional challenges including a lack of standard representation for the fonts and encoding, even operating systems, browsers, and keyboards may not directly support some of these languages. This leads to language processing routines than the actual effort required for the OCR model development.

Even though numerous attempts have been made, to overcome the above-stated general and language-specific challenges of Amharic script, only few attempts are made to develop an OCR model for Amharic character recognition. The OCR models developed by researchers so far are based on classical machine learning techniques and limited Amharic characters were considered for training as a result, the OCR model yields poor character recognition performance and even no organized and publicly available OCR dataset. Therefore, new paradigms should be researched and even the accuracy of the existing algorithm should be verified and optimized. The next section presents efforts that have been made for developing Amharic OCR.

3.4 Related Works

Following the 1997 first attempt made by Worku [Ale97], different statistical machine learning techniques have been explored by many researchers for Amharic script recognition. In 1999, Dereje [Tef99] has conducted research so as to recognize a typewritten Amharic text with the aim of improving Amharic OCR and recommended as the algorithms adopted for Amharic OCR should not be very sensitive to the features of the writing styles of characters. A year later, Million [Mil00] adopted a recognition algorithm that can handle the different typefaces of Amharic characters with the aim to investigate and extract the attributes of Amharic characters so as to generalize the previous research works on Amharic OCR.

As a continuation of the research activities done so far, Yaregal [Ass02] proposed an integrated approach and come up with a versatile algorithm that is independent of the fonts and other quantitative parameters of Amharic characters. In addition, other recent attempts have also been made employing deep learning techniques [GSTBJ19, RRB18]. However, nearly all of these attempts performed segmentation of text images into words and then into character level which directly affected the performance of the OCR model.

The only exception was Assabie [AB09], proposed an HMM-based model for off-line handwritten Amharic word recognition without character segmentation using structural features of characters as building blocks of the recognition system. Recently published work [ALT18], proposed a segmentation-free technique based on Long-Short-Term Memory (LSTM) networks together with CTC (Connectionist Temporal Classification) for Amharic text-image recognition. However, the datasets are not publicly available and all these attempts to Amharic OCR neither shown results on a large dataset nor considering all possible characters.

Although OCR development attempted regarding Amharic script so far is limited, in this thesis, previous research attempts are taken as a baseline and further investigations are made so as to facilitate the OCR of Amharic script. In addition, new Amharic OCR databases are prepared and made freely available for the public with benchmark results recorded empirically with state-of-the-art OCR techniques.

3.5 Chapter Summary

In this chapter, we have discussed the overview of Amharic script. The origin of Amharic script, character sets with its formation, the generic structure, and its relationship with other Semitic languages are presented. In addition, we highlight the difficulties and challenges related to Amharic script that have bearings on the OCR development, especially, the existence of visually similar symbols and the use of a number of distinct fonts in printed documents that are designed in an unstructured manner.

We also present a short survey of related works made for the indigenous languages with a particular emphasis on Amharic script which is currently used as the official working language of Ethiopia with its own indigenous alphabets and writing system called Fidel. Amharic script has 34 base characters each of which occurs in seven orders, which represent syllable combinations consisting of a consonant and following vowel. The total number of unique characters, used in Amharic script, is about 317, including 238 basic characters and other symbols representing labialization, numerals, and punctuation marks.

Developing the OCR model for an indigenous, but the resource-limited language is challenging and new algorithms are required to address these challenges. Unlike most modern Latin scripts enjoying a collection of resources including a database and benchmark results of OCR, the scarcity of such in-need resources makes the OCR development very challenging for Amharic script. In the next chapter, a baseline database prepared for Amharic OCR development is described in detail.

Chapter 4

Amharic OCR and Baseline Datasets

This chapter gives a brief explanation about a baseline database developed for Amharic OCR and techniques employed for Amharic database creation. First, it gives highlights on the effect of dataset unavailability on Amharic character recognition tasks. Next, it introduces related works regarding datasets in general and in particular for Amharic script. Finally, it gives a summary of all works and contributions regarding Amharic OCR database.

4.1 Introduction

The shortage of training dataset is one of the limiting factors in developing a reliable system in the area of image processing and pattern recognition, in general, and for Amharic OCR in particular. Since the performance of the OCR model is depending on the size and quality of training datasets, we need to prepare data that is enough for learning and generalization. Manual annotation of the dataset requires lots of time and effort. Even, it is sometimes difficult to transcribe the text image into its corresponding ground-truth texts unless we are an expert of the language.

The following solutions, as the contribution of this thesis, are proposed to overcome the shortage of annotated datasets with a particular focus on Amharic script.

- Synthetically generated Amharic database has been introduced. This database consists of character-level images and text-line-level images with their corresponding ground-truth texts in two common Amharic

fonts (Power Ge'ez and Visual Ge'ez fonts).

- Scanned printed Amharic database. This database contains scanned printed text-line images written in Power Ge'ez font for the OCR of Amharic script.

All these databases are named Database for Amharic OCR (ADOCR) and it is, now, freely available at `http://www.dfki.uni-kl.de/~belay/`. The details of these databases are described in section 4.3.

4.2 Dataset for OCR Tasks

In literature, various researchers use different approaches to prepare a dataset for developing the OCR model. Some of them employed synthetic text-image generator engine so as to render a synthetic image with their corresponding ground-truth texts while others used manual annotation techniques. The other approach in developing a database is first creating documents electronically and then printed out. Finally aligned the scanned version of this document image with the previous corresponding electronically created texts.

Nowadays, the use of synthetically generated data is getting popular in the area of image processing and document image analysis tasks. There are multiple OCR databases created, and reported in the literature, using this technique. Sabbour and Shafait [SS13] presented a synthetic text-line image database called Urdu Printed Text Images (UPTI) database. This database consists of 10,063 Urdu text lines images, synthetically generated with the Nastaliq font, and their corresponding ground-truth text.

Arabic Printed Text Image (APTI) database is also a synthetically generated Arabic database developed by Slimane et al [SIK+09]. This database is generated using more than 100 thousand words, 10 Arabic fonts with a variety of font sizes and styles which are widely used on a computer screen, Arabic newspapers, books, and many other Arabic documents. The database contains over 45 million words of images and more than 250 million total characters.

The other type of synthetically generated dataset is the SynthText in the Wiled Dataset published by Gupta et al [GVZ16]. The database consists of 800 thousand images with approximately 8 million synthetic word instances that are placed in natural scene images. Each text instance is annotated with its text word and character level bounding-boxes. There are also other numerous scanned text images of OCR databases such as IAM-DB Handwritten [MB02]

database, RIMES [GCBG08] database which contains 1539 page of English texts and 12,723 scanned page of French mails respectively.

A real-world image dataset called the Street View House Numbers (SVHN) [NWC+11] is proposed for machine learning and object recognition tasks. This dataset consists of 600,000 digits of house-number images with a variable resolution, where 73,257 digits for the training set, 26,032 digits for the test set, and 531,131 digits of the extra training set with less difficult samples. SVHN dataset is similar to MNIST [LBBH98] dataset, but SVHN comes from a significantly harder and unsolved problem which are obtained from house numbers in Google Street View image. COCO-text [VMN+16] is the other real-world and benchmarking database specifically designed for text detection and recognition in natural images. The COCO-text dataset consists of 63,000 images and more than 145,000 instances of machine-printed and handwritten texts.

Chen and Qingcai [ZCW10] proposed the Harbin Institute of Technology Opening Recognition Corpus for Chinese Characters (HIT_OR3C) database which contains more than 900 thousand images of Chinese characters. The characters have been collected using a handwriting pad and are recorded and labeled automatically via the handwriting document collection software called OR3C Toolkit.

The IMPACT dataset [PPCA13], proposed by Papadopoulos et al, is a printed text database that contains over 600,000 historic document images that originate from major European libraries and covering texts from newspapers, books, pamphlets, and typewritten notes. However, to the best of our knowledge, there is no publicly available dataset for Amharic script recognition. In this regard, we have prepared a new public Amharic database and a baseline recognition system. The dataset is a composition of several datasets, a character image and a text-line image dataset that includes scanned printed and synthetic texts.

4.3 Database for Amharic OCR

The primary issue in the field of OCR in general and in particular for Amharic OCR is a lack of annotated dataset. Therefore, this section describes various approaches followed to develop the database and the detail of Amharic OCR database called ADOCR [BHL+19a]. The ADOCR database contains character and text-line-level images. Both datasets are available freely and can

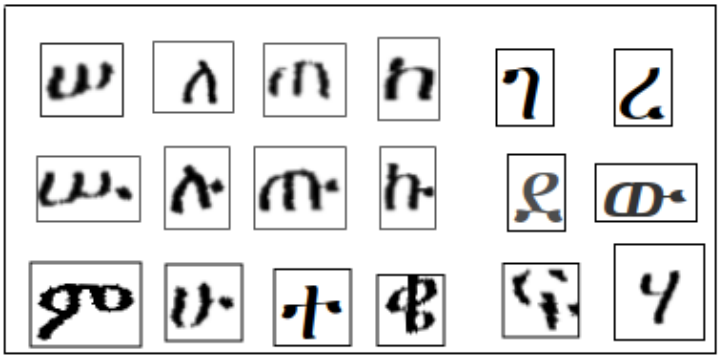


Figure 4.1: **Sample synthetically generated Amharic character images.** *All these character images are generated using different degradation levels*

be also obtained from the author, for research purposes. Sample character images from this dataset are illustrated in Figure 4.1. The following section presents the details of the ADOCR database.

4.3.1 Amharic Character Image Dataset

Despite their numerous attempts to do so, researchers have not been able to prepare Amharic datasets that are large enough to train deep learning models. In addition, none of them are made the datasets publicly available. Therefore, to advance the OCR research for Amharic script, a new dataset of Amharic characters is proposed.

This dataset is organized into two groups. The first group, *Type-I*, consists 80,000 Amharic character images with the corresponding Ground-Truth (GT) texts. The character images and their corresponding GT, in this group, are in the format of *png* and *txt* file respectively. Since it has no explicitly separated training and test dataset, researchers are free and should split both the training and test sets accordingly.

However, the second group of the character level dataset contains 72,194 sample Amharic character images as a training and 7800 sample images as a test set. In this dataset, *Type-II*, both the character images and the corresponding GT are stored in *numpy* file format. Moreover, the images are Grey-level with a size of 32 by 32 pixels and generated synthetically using, OCRopus [Bre08], an open-source OCR framework. Except about 2006 distorted Amharic character images, which are removed from *Type-I*, both *Type-I* and

	0	1	2	3	4	5	6
0	ሀ hā	ሁ hu	ሂ hī	ሃ ha	ሄ hé	ህ hi	ሆ ho
1	ለ lā	ሉ lu	ሊ lī	ላ la	ሌ lé	ል li	ሎ lo
2	ሐ ḥā	ሑ ḥu	ሒ ḥī	ሓ ḥa	ሔ ḥé	ሕ ḥi	ሖ ḥo
3	መ mā	ሙ mu	ሚ mī	ማ ma	ሜ mé	ም mi	ሞ mo
4	ሠ śā	ሡ śu	ሢ śī	ሣ śa	ሤ śé	ሥ śi	ሦ śo
5	ረ rā	ሩ ru	ሪ rī	ራ ra	ሬ ré	ር ri	ሮ ro
6	ሰ sā	ሱ su	ሲ sī	ሳ sā	ሴ sé	ስ si	ሶ so
7	ሸ šā	ሹ šu	ሺ šī	ሻ šā	ሼ šé	ሽ šei	ሾ šo
8	ቀ qā	ቁ qu	ቂ qī	ቃ qā	ቄ qé	ቅ qi	ቆ qo
9	በ bā	ቡ bu	ቢ bī	ባ bā	ቤ bé	ብ bi	ቦ bo
10	ተ tā	ቱ tu	ቲ tī	ታ tā	ቴ té	ት ti	ቶ to
11	ቸ čā	ቹču	ቺ čī	ቻ čā	ቼ čé	ች či	ቾ čo
12	ኀ hā	ኁ hu	ኂ hī	ኃ hā	ኄ hé	ኅ hi	ኆ ho
13	ነ nā	ኑ nu	ኒ nī	ና nā	ኔ né	ን nei	ኖ no
14	ኘ ñā	ኙ ñu	ኚ ñī	ኛ ñā	ኜ ñé	ኝ ñi	ኞ ño
15	አ ʾa	ኡ ʾu	ኢ ʾī	ኣ ʾā	ኤ ʾé	እ ʾe	ኦ ʾo
16	ከ kā	ኩ ku	ኪ kī	ካ kā	ኬ ké	ክ ki	ኮ ko
17	ኸ xā	ኹ xu	ኺ xī	ኻ xā	ኼ xé	ኽ xi	ኾ xo
18	ወ wā	ዉ wu	ዊ wī	ዋ wā	ዌ wé	ው wi	ዎ wo
19	ዐ ʿa	ዑ ʿu	ዒ ʿī	ዓ ʿā	ዔ ʿé	ዕ ʿe	ዖ ʿo
20	ዘ zā	ዙ zu	ዚ zī	ዛ zā	ዜ zé	ዝ zi	ዞ zo
21	ዠ žā	ዡ žu	ዢ žī	ዣ žā	ዤ žé	ዥ ži	ዦ žo
22	የ yā	ዩ yu	ዪ yī	ያ yā	ዬ yé	ይ yi	ዮ yo
23	ደ da	ዱ du	ዲ dī	ዳ dā	ዴ dé	ድ di	ዶ do
24	ጀ ǧā	ጁ ǧu	ጂ ǧī	ጃ ǧā	ጄ ǧé	ጅ ǧi	ጆ ǧo
25	ገ gā	ጉ gu	ጊ gī	ጋ gā	ጌ gé	ግ gi	ጎ go
26	ጠ ṭā	ጡ ṭu	ጢ ṭī	ጣ ṭā	ጤ ṭé	ጥ ṭi	ጦ ṭo
27	ጨ ċā	ጩ ċu	ጪ ċī	ጫ ċā	ጬ ċé	ጭ ċi	ጮ ċo
28	ጰ p̄a	ጱ p̄u	ጲ p̄ī	ጳ p̄ā	ጴ p̄é	ጵ p̄i	ጶ p̄o
29	ጸ ṣā	ጹ ṣu	ጺ ṣī	ጻ ṣā	ጼ ṣé	ጽ ṣi	ጾ ṣo
30	ፀ ṡā	ፁ ṡu	ፂ ṡī	ፃ ṡā	ፄ ṡé	ፅ ṡi	ፆ ṡo
31	ፈ fā	ፉ fu	ፊ fī	ፋ fa	ፌ fé	ፍ fi	ፎ fo
32	ፐ pā	ፑ pu	ፒ pī	ፓ pā	ፔ pé	ፕ pi	ፖ po

Figure 4.2: **Fidel Gebeta.** *Basic Amharic characters are arranged in row-column-wise (33×7) order; read from left to right: The first column gives the consonant characters while the next six columns are derived by combining the consonant with vowel sounds.*

Type-II datasets contain the same character images. In this dataset, we only considered 231 basic Amharic characters which are arranged in a matrix-like structure, called Fidel Gebeta, and illustrated in Figure 4.2.

4.3.2 Amharic Text-line Image Dataset

Many research works on Amharic script recognition so far were focused on character level recognition as a result no attempts have been made by researchers to prepare a text-line database for Amharic OCR; thus the first attempt is made in this work. Amharic text-line image dataset, part of ADOCR database, contains 337,337 Amharic text-line images composed of the two commonly used fonts (*Visual Ge'ez* and *Power Ge'ez* fonts) of Amharic script.

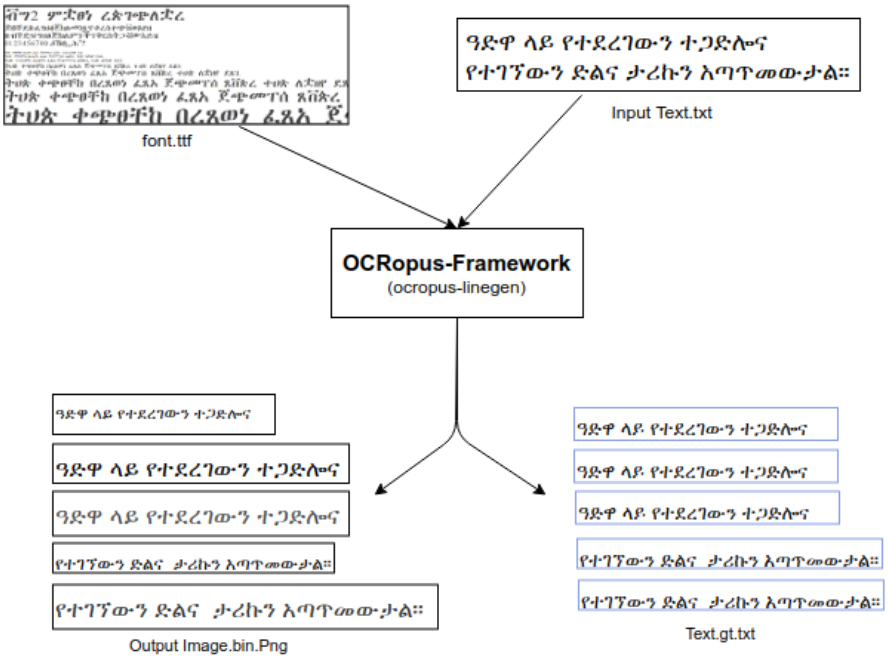
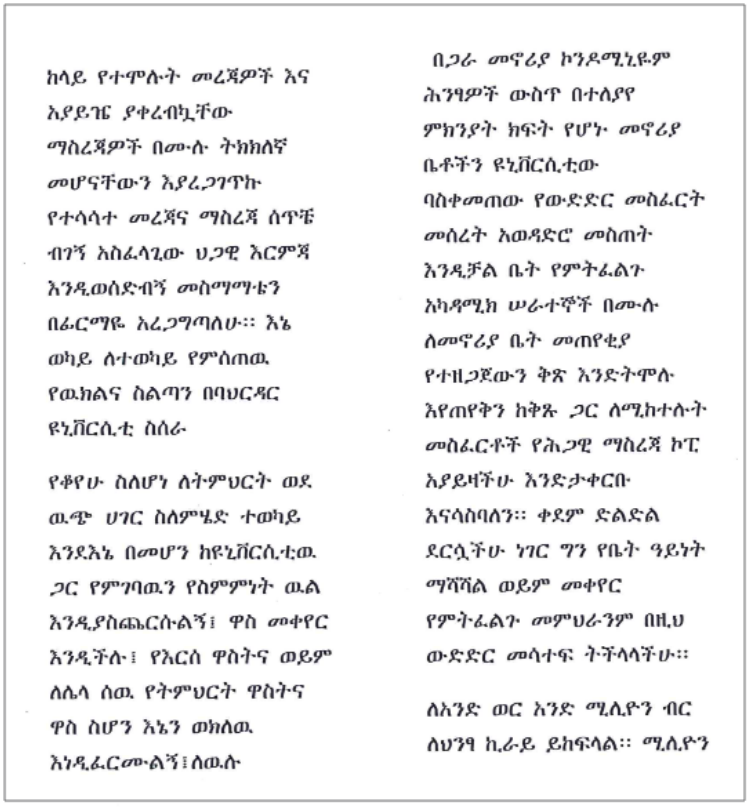


Figure 4.3: **Synthetic Amharic text-line image generation pipelines.** *The textual input along with desired font type are goes through the OCRopus-linegen and then the text-line images and their corresponding Ground-Text GT texts are generated. Variations of each text-line image are depends of degradation parameters and fonts considered during data generation.*

These text-line images have two parts. The first part is synthetically generated Amharic text-line images while the second part is a printed Amharic text-line images.

Following the popularity of using artificial data in computer vision and pattern recognition, the shortage of training datasets is, now, partially solved. A similar approach is employed, in this thesis, to address the problem of limited labeled Amharic script dataset. During synthetic data generation, to make the synthetic data closely similar to the scanned documents that represent real-world data (such as degradation, font, and skew), we use different parameters from the OCRopus framework such as threshold with values between 0.3 and 0.5, jitters with the value of 0.5 and so on.

As shown in Figure 4.3, the *utf-8*-encoded text line and *ttf*-type font-files are used as the inputs of *OCRopus-linegen* engine and generate the text-line image with corresponding GT texts. Since the *OCRopus-lingen* engine doesn't generate the character-level image directly, we prepare a text file that



ከላይ የተሞሉት መረጃዎች እና
አያይዤ ያቀረብኳቸው
ማስረጃዎች በሙሉ ትክክለኛ
መሆናቸውን እያረጋገጥኩ
የተሳሳተ መረጃና ማስረጃ ሰጥቼ
ብገኝ አስፈላጊው ህጋዊ እርምጃ
እንዲወሰድብኝ መስማማቴን
በፊርማዬ አረጋግጣለሁ። እኔ
ወክይ ሰተወክይ የምሰጠዉ
የዉክልና ስልጣን በባህርዳር
ዩኒቨርሲቲ ስሰራ

የቆየሁ ስለሆነ ለትምህርት ወደ
ዉጭ ሀገር ስለምሄድ ተወካይ
እንደእኔ በመሆን ከዩኒቨርሲቲዉ
ጋር የምገባዉን የስምምነት ዉል
እንዲያስጨርሱልኝ፤ ዋስ መቀየር
እንዲችሉ፤ የእርሰ ዋስትና ወይም
ለሌላ ሰዉ የትምህርት ዋስትና
ዋስ ስሆን እኔን ወክለዉ
እነዲፈርሙልኝ፤ስዉሉ

በጋራ መኖሪያ ኮንዶሚኒዬም
ሕንፃዎች ውስጥ በተለያየ
ምክንያት ክፍት የሆኑ መኖሪያ
ቤቶችን ዩኒቨርሲቲው
ባስቀመጠው የውድድር መስፈርት
መሰረት አወዳድሮ መስጠት
እንዲቻል ቤት የምትፈልጉ
አካዳሚክ ሠራተኞች በሙሉ
ለመኖሪያ ቤት መጠየቂያ
የተዘጋጀውን ቅጽ እንድትሞሉ
እየጠየቅን ከቅጹ ጋር ለሚከተሉት
መስፈርቶች የሕጋዊ ማስረጃ ኮፒ
አያይዛችሁ እንድታቀርቡ
እናሳስባለን። ቀደም ድልድል
ደርሷችሁ ነገር ግን የቤት ዓይነት
ማሻሻል ወይም መቀየር
የምትፈልጉ መምህራንም በዚህ
ውድድር መሳተፍ ትችላላችሁ።

ለአንድ ወር አንድ ሚሊዮን ብር
ለህንፃ ኪራይ ይከፍላል። ሚሊዮን

Figure 4.4: **A sample scanned page of Amharic document.** *This is a two column sample image among a collection of printed Amharic documents.*

consists of a character per line first and then feed this text file to the engine for character-level image generation. Furthermore, there are some necessary changes regarding the *OCRopus* framework code so as to adapt for documents written in Amharic script.

Even though using synthetic data is common in most pattern recognition and OCR research, it is known that the synthetic data doesn't represent all the real-world artifacts. Moreover, the OCR model trained with the synthetic data lacks generalization of the real data. The usage of real-world data, if available, is always a choice in the field of OCR. For Amharic OCR, there is no printed text-line image dataset that could be used to train the OCR model and explore real-world challenges so as to develop an effective digitization system for Amharic script. In this regard, we are motivated to develop a compressive Amharic text-image dataset for benchmarking state-of-the-art methods.

During printed text-line image dataset preparation the contents of the document are an important factor; thus the collection of contents should have to

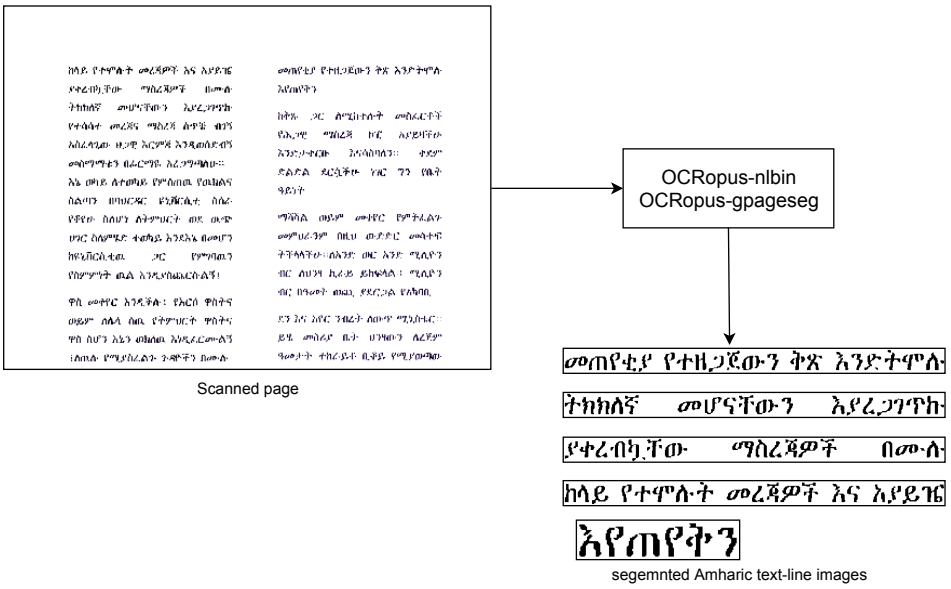


Figure 4.5: **A typical diagram of OCRopus page segmentation-engine.** *The OCRopus page segmentation engine produces a segmented text-line image from a scanned page of documents.*

represent the real facts in Amharic script. In this regard, an electronic copy of Amharic documents like letters, newspapers, books, and magazines which are written with 12 point pixel including cultural, economical, political, and religious contents was collected. Once the documents are collected and organized, the printing process is done using a laser jet printer with the usual setting used in Amharic document printing process.

The next step, after printing, is image acquisition which can be done using different technologies like cameras, scanners, and other mobile-based technologies. In most document digitization, the scanner-based acquisition is considered as a standard approach and abundantly used for scanning books, magazines, and other textual resources. The scanned contents of printed Amharic texts are acquired by scanning 721 pages of printed Amharic documents using a flatbed scanner with 300dpi. Sample scanned page of Amharic document is shown in Figure 4.4.

For page image binarization and text-line segmentation, a binarization and line segmentation techniques from OCRopus framework are employed. The overall page binarization and line segmentation procedures are illustrated in figure 4.5. The errors observed during text-line segmentation are annotated and aligned with its corresponding GT text manually. This technique can be used as an alternative solution for printed dataset preparation and it re-

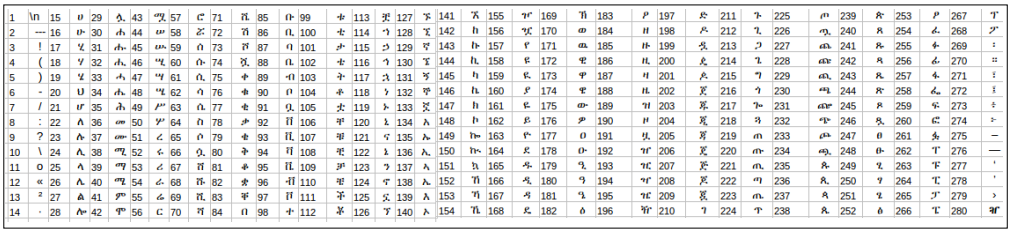


Figure 4.6: **Amharic characters and punctuation marks.** *A representative of each character and punctuation mark are presented in the ground-truth of the training and test samples of the ADOCR database.*

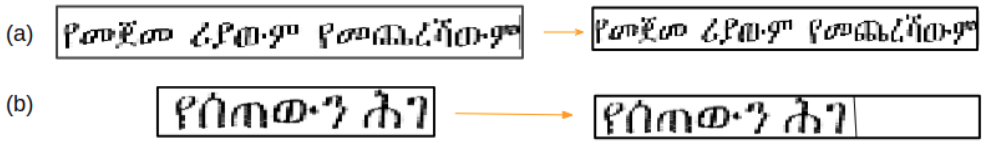


Figure 4.7: **Text-line image length normalization.** *An image in (a) has a size larger than* 48×128 *while image in (b) is less than that. Therefore, both the images are scaled proportionally. Images at the left are random text-line images from the dataset while the right side images are resized to* 48×128*. An image with empty space at the right side of the image (b) is part of the target image filled with white color. By doing image size normalization enables us to train the network with a batch of data shuffled in every epoch.*

duces the time and effort required during traditional text-line image annotation which is usually done by transcribing each text-line manually by human operators or language experts.

From the total set of text-line images, 40,929 text-lines are printed text-lines written with Power Ge'ez font, 197,484 and 98,924 text-line images are synthetically generated text-lines with Power Ge'ez and Visual Ge'ez fonts respectively. In the GT texts there are 280 unique Amharic characters and punctuation marks which are depicted in Figure 4.6 and all characters that exist in the test dataset also exist in the training set. The text-lines in the ground-truth of the ADOCR dataset have a minimum sequence length of 3 and a maximum length of 32 characters. This dataset totally consists of 851,711 words that are composed of 4,916,663 characters, where 269,419 characters are from the test datasets and 4,647,244 characters are from the training dataset.

Since the text-line images have different lengths and the implementation deep learning framework, Keras, used in this thesis requires a limited se-

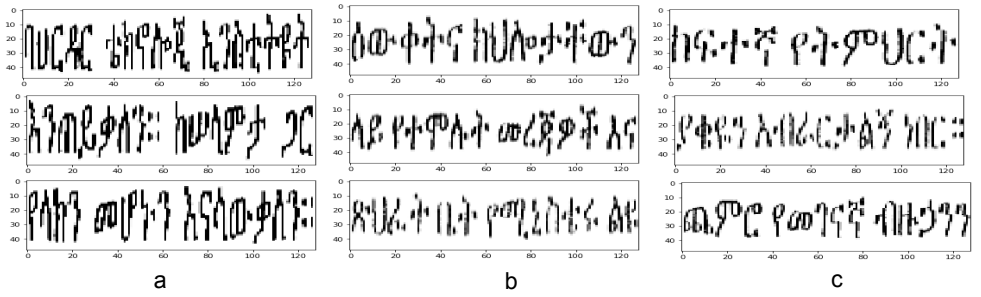


Figure 4.8: **Sample Amharic text-line images from ADOCR database.** *All images in this figure are normalized to a size of 48 by 128 pixels. (**a**) Printed text-line images written with the Power Ge'ez font type. (**b**) Synthetically generated text-line images with the Visual Ge'ez font type. (**c**) Synthetically generated text-line images with the Power Ge'ez font type.*

quence length to optimize GPU parallelization, all text-line images are resized to a width of 128. During text-line image length normalization, first, we create a white target image of size 48×128. Then we copy each text-line image into the white target image, as shown in Figure 4.7. Now all the text-line images are resized to an equal and fixed length which is easy for batch training where the batch size is greater than one.

The synthetic text-line images, in this database, are generated and stored at a different level of degradation. The text-line images and their *GT* text files in the training set are in the order of printed, synthetically generated images with Power Ge'ez and Visual Ge'ez fonts respectively. All images and their corresponding *GT* text are in the format of *numpy* file. Images in a test-set are organized as a separate *numpy* file with their corresponding *GT* texts each. Sample printed and synthetically generated text-line images taken from the ADOCR database are shown in Figure 4.8. The test dataset contains a total of 18,631 randomly selected text-line images where 2907 of them are scanned printed, 9245 and 6479 of them are synthetically generated text-line images with *Power Geez* and *Visual Geez* fonts types respectively.

4.4 Chapter Summary

In this chapter, a baseline Amharic OCR database, called ADOCR, is presented. This database is composed of printed and synthetically generated images of Amharic documents. The ADOCR database consists of two types

of datasets; the first dataset is composed of images at character-level while the second database consists of text-line level images. Besides, the ADOCR database consists of two common Amharic fonts (Power Ge'ez and Visual Ge'ez) To develop these datasets two approaches are employed, one is to generate images synthetically using different degradation levels and font types while the second approach is to prepare the scanned printed text-line images. The main contribution of this chapter is the creation of the ADOCR database, an Amharic OCR database that consists of synthetically generated images of Amharic characters and text-lines. It also contains scanned printed Amharic text-line images with their corresponding ground-truth texts. The contribution of this new Amharic database is not only to the field of Amharic script recognition but also to many other document image analysis tasks, in general. In the next chapters, of this thesis, we provide the details of OCR models proposed for Amharic script recognition and experimental results recorded using this OCR database.

Chapter 5

Recognition of Amharic Characters

Nowadays, it is becoming increasingly important to have digital documents for easy access to information, efficient data storage, and retrieval. Therefore, the use and applications of OCR systems have been developed and widely applied for the digitization of most languages. In this part of the thesis, an overview of Amharic script and CNN-based Amharic character recognition systems are provided. This chapter has two sections: the first section of this chapter states the background of Amharic script and attempts done to develop a character recognition model for basic Amharic characters based on a standard CNN architecture. The second section presents a novel CNN-based Amharic character recognition network architecture which is designed by leveraging the grapheme of Amharic characters in Fidel-Gebeta and by taking the concept of multi-task learning.

5.1 Amharic Character Recognition Using CNNs

In this section, we introduce the first attempt for Amharic character image recognition based on deep learning benchmarks. A standard Convolutional Neural Network (CNN) has been employed for the recognition of basic Amharic characters. The proposed method is trained with synthetically generated Amharic character images so as to replace the shortage of training data and the performance of the model evaluated using both synthetically generated and printed Amharic characters. The detailed implementation procedure and related works are provided in the following section.

5.1.1 Introduction

Amharic is an old Semitic language that has been and is widely spoken in Ethiopia. It is widely used in the country and serves as the official working language of the state. Most historic and literary documents are written and documented in this language. Despite the importance of the language, scanned image documentation and retrieval in Amharic script is still a tedious process. Technologies that can automate the process of changing this scanned image into a full and editable document will be of enormous use in many sectors of Amharic speaker nation's economy. However, unlike the well-known Latin scripts such as English and Fraktur and Non-Latin script such as Arabic scripts, there is no workable and effective off-the-shelf OCR software for Amharic script recognition. Over the last few decades, a limited amount of research efforts have been done for the recognition of Amharic scripts, where almost all these attempts are based on classical machine learning techniques.

Most published research works on Amharic OCR [Tef99, Ass02, MJ07, Mes08], relied on traditional and statistical machine learning-based techniques such as Support Vector Machines and Multi-layer Perceptrons. However, significant progress has been made in OCR techniques for many other scripts [KJEY20, BUHAAS13, YBJL12, BCFX14]. Adoption of recent tools and algorithms, to Amharic scripts, will be greatly advance the state-of-the-art in Amharic OCR.

OCR has been and is widely used for many scripts as a method of digitizing printed texts which can be electronically edited, searched, stored more compactly, displayed on-line, also used to facilitate the human-to-machine and machine-to-machine communication such as machine translation, text-to-speech, key data, and text mining [YBJL12, Mes08, GDS10]. Previous research works on Amharic character recognition follows a long procedure which starts by segmenting input image into lines, characters and then followed by feature extraction and selection. The selected features are used as an input for a classifier which causes recognition lattice [YBJL12, Tef99] and also requires more time and techniques during preprocessing and feature extraction.

To optimize the efficiency of the OCR model and minimize information loss during feature extraction, we adopt deep learning-based techniques from the existing work [AT17] for Amharic character recognition. Convolutional neural network (CNN), is selected for its less number of steps and computational resources which have been used for statistical feature extraction

[Bre08, YBJL12, ZHZ17]. CNNs take the advantages of GPU-days of computation to train on large-scale datasets and it is also widely applied for character recognition [ZHZ17], image recognition [SZ14] and numerals recognition [AT17]. CNNs enable to automatically extract the salient features, which are invariant for a certain degree to shift and shape distortions of the input characters [BCFX14, ZHZ17]. In addition to this, the use of shared weight in convolutional layers, the same filter is used for each input in the layers. Sharing weight across CNN layers, reduces the number of parameters and improves recognition performance [BCFX14].

As reported in the literature, a small private database of handwritten and machine-printed images, for most Latin-based scripts such as English and non-Latin scripts such as Arabic and Amharic, were widely used to train classical machine learning algorithms and reported many challenges [MJ07, Tef99, BCFX14]. These researchers are agreed on the nature of the dataset and most of them recommended that; the use of large and multi-font character images that have various levels of degradation, for OCR model training, enhances the recognition performance. Therefore, we develop an Amharic OCR system that is capable to recognize Amharic characters written with multiple fonts and generated at a different level of degradation.

In this research work, a convolutional neural network-based model is proposed with the goal of training the model using a synthetic Amharic character image and then applied it for recognition of both synthetically generated and printed Amharic characters. This model is also a benchmark model for character-level Amharic OCR. In this regard, different CNN architectures are trained rigorously so as to achieve very low error rates compared to other standard systems and methods.

5.1.2 Related Work

For character recognition, various methods have been proposed and high character recognition performances are reported for the OCR of various scripts such as English [BCFX14], Fraktur [BUHAAS13], Arabic [GDS10], Devanagari [BCFX14] , Malayalam [AMKS15] and Chinese[HZMJ15]. In the last two decades, many researchers [Tef99, Ass02, MJ07, Hag15] are also attempted to address the problems in Amharic script recognition.

Dereje [Tef99] proposed a Binary Morphological filtering algorithm for OCR of Amharic typewritten texts. He recommended that adopting recogni-

tion algorithms that are not very sensitive to the features of the writing styles of characters helps to enhance the recognition rate of Amharic OCR. Million [MJ07], conducted empirical research to investigate and extract the attributes of Amharic characters written in different fonts and then generalize previously adopted recognition algorithms. Using different test cases, 49.38%, 26.04%, and 15.75% recognition accuracy were registered for WashRa, Agafari, and Visual Geez fonts respectively. In addition, Yaregal [Ass02] noted that font variation is also a problem in designing an OCR system for printed Ethiopic documents.

Shatnawi and Abdallah [SA15] tried to model a real distortion in Arabic script using real handwritten character examples to recognize characters. The distortion models were employed to synthesize handwritten examples that are more realistic and achieved 73.4% recognition accuracy. Haifeng [ZHZ17], proposed a deep learning method based on the convolutional neural networks to recognize scanned characters from real life with the characteristics of illumination variance, cluttered backgrounds, geometry distortion, and reported promising results.

In summary, the above studies noted that the writing style of characters that includes font variation, level of degradation, and size are important factors for OCR. Preprocessing, feature extraction and classification algorithm selection is also the major step for constructing an effective recognition model. Even though, many researchers have been attempted to recognize a small set of Amharic character images using hand-crafted features which is usually not robust, to the best of our knowledge, no attempts have been made for Amharic character recognition using deep learning-based techniques. Therefore, there is still room for improvement of Amharic character image recognition using state-of-the-art techniques that can reduce preprocessing steps and computational resources.

5.1.3 Research Methods

This study is concerned with the development of Amharic character image recognition model using deep learning techniques. For experimentation, about 80,000 synthetically generated Amharic characters and 6715 printed Amharic character images are employed. The detail description of the character images is given in chapter 4 section 4.3.1. For generating the synthetic character images, we employed an open-source generic OCR framework called OCRopus, which is developed by Breuil [Bre08]. The printed characters are prepared us-

ing the usual printer setting, in Amharic document, and then scanned using Kyocera TASKalfa 5501i flatbed scanner at a resolution of 300 DPI. Preprocessing steps including binarization, and character image segmentation has been done using the same tool, the *OCRopus* framework. The *OCRopus* OCR framework and the detail procedure of dataset generation is described in chapter 4 section 4.3.1 and 4.3.2.

5.1.4 The Proposed CNN Architecture

After doing several experiments considering different Convolutional Neural Network architecture (CNN), the better results reported, in this section, with CNN architecture depicted in Figure 5.1 and a pseudo-code for training the proposed model is shown in *Algorithm 1*. In this architecture, five convolutional layers, two in each block except the last layer, are used consecutively with a rectified linear unit (ReLU) as an activation function followed by single max-pooling. we also use the *Same* padding scheme so as to determine the output size and two fully connected layers followed by the soft-max function for the final layers. A 3×3 convolutional filter and a 2×2 max-pooling are applied for performing filtering and sub-sampling operations respectively.

For training, we consider Amharic character images that have an input size of 32×32 pixel. To optimize the proposed network model, training is carried out by using the stochastic gradient descent method with the mini-batches size of 150 and momentum value of 0.9. A dropout rate of 0.25 was used to regularize the network parameters. The output of each convolutional block is fed to the next network block until the probability of the last layer is calculated where the activation function is $Relu$. The stochastic gradient descent method optimizes and finds the parameters of connected network layers that minimize the prediction loss. The final output of the model is determined by a Soft-max function, which tells that the probability of the classes being true is computed using Equation (5.1).

$$f(z_j) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}, \text{for } j = 1, ..., k \tag{5.1}$$

Where z is a vector of the inputs to the output layer and k is the number of outputs.

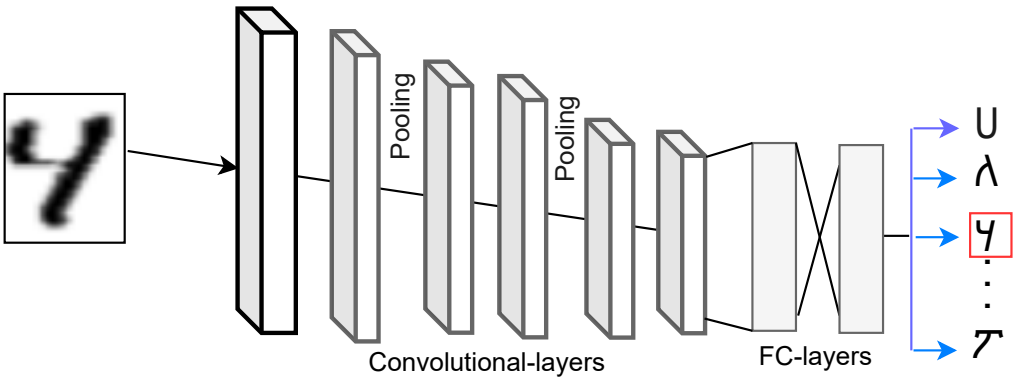


Figure 5.1: **A CNN-based OCR model for Amharic character image recognition.**

Algorithm 1: **Pseudocode for training the proposed model**

Input: Model, T, t Model= Keras sequential model, T=training data-set, t=test data-set
Output:trained model
Start: adding convolutional and pooling layers
Model.add(x Conv y, T, relu) x= feature-maps, y= filter size, conv=convolution layer
Model.add(x Conv y, T, relu)x= feature-maps, y= filter size, conv=convolution layer
Model.add(xPy) x= is pooling size, y= stride, p=pooling layer
... here adding the 2 convolutional followed by pooling layers and then Conv
Model.add(xFC) x=number of neurons, FC= fully connected layer
Model.add(NC) NC=number of class
Model.add(softmax)
Model.compile(Optimizer, accuracy measure)
Model.fit(T, epochs, validation-split)
Stop

5.1.5 Experimental Results

The implementation is based on Keras Application Program Interface (API) with TensorFlow backend. Experiments are run following the network architecture and classification settings introduced in section 5.1.4, then the training and validation loss observed during experimentation is depicted in Figure 5.2. During experimentation, we train our model using scholastic gradient descent optimizer for different epochs, compared the results, and rerun our model on more epochs with selected parameters, and the most promising validation accuracy was recorded at 100^{th} epoch using 70%, 10% and 20% of training, validation and test sets of synthetically generated Amharic character image dataset respectively.

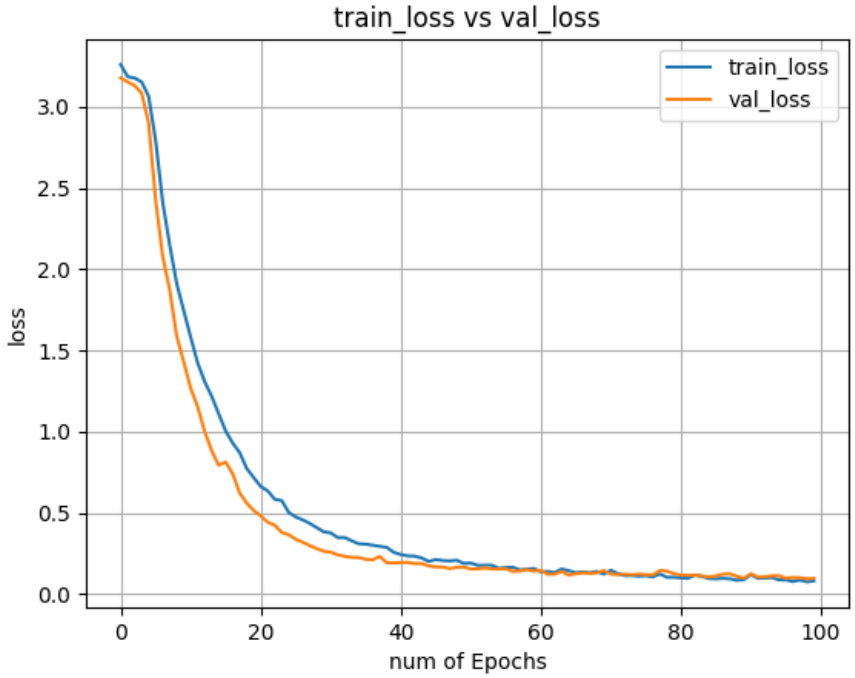


Figure 5.2: **Training versus validation loss of CNN-based OCR model.**

Once the recognition model is developed, the performance of the proposed model is evaluated against the test datasets. The first test dataset contains synthetic Amharic character images generated using OCRopus OCR-framework with the most common Amharic font called Visual Geez. The second test sets are printed Amharic character images which are prepared separately so as to evaluate the performance of the model trained with the synthetic one. This dataset consists 6,715 printed Amharic characters written with the different Amharic fonts such as *Power Geez*, *Visual Geez*, *Agafari* and *WashRa*. Based on the results recorded during experimentation, 96.02 % of printed characters are correctly predicted while 92.71% of synthetically generated character images are correctly predicted.

An investigation of experimental results indicated that most of the characters are incorrectly recognized due to shape similarity while the others are incorrectly recognized even with no similarity between them. The main reason to have lower recognition performance in the synthetically generated dataset is the nature of the character images (i.e most of the synthetic character images which are incorrectly recognized are deformed characters). On the other hand, compared to the synthetic character images the printed character images have relatively better image quality. In this section, we only consider the 231

basic Amharic character sets.

5.1.6 Recognition Performance Comparison

To the best of our knowledge, there is no publicly available dataset for Amharic script recognition and all researchers have been done experiments by preparing their own private datasets. Therefore, it may not be directly relevant to compare the current Amharic character recognition model with other works which are performed under different experimental conditions and datasets. However, the comparison between these works can be used as an indication of research progress for Amharic character image recognition. The result of the current Amharic character recognition model and the results of previous classical machine learning-based research works are shown in Table 5.1.

Table 5.1: Character level recognition accuracy (out of 100%) and performance comparison with related research works.

	Dataset size	**Document type**	**Classifier**	**Accuracy**
Dereje [Tef99]	5172	Typewritten	ANN	61%
Million [MJ07]	7680	Printed	SVM	90.37%
Yaregal [Ass02]	1010	Printed	ANN	73.18%
Fitum [Dem11]	1500	Printed	SVM	91%
Ours	80000	Synthetic	CNN	92.71%

The second column (data size) refers to the number of character images used for experimentation

5.1.7 Conclusion

We experimentally developed an Amharic character image recognition model using a convolutional neural network and the proposed Amharic character recognition model outperforms other state-of-the-art methods proposed for Amharic script recognition. The convolutional neural network tends to work better with raw input pixels rather than features or parts of an image. During dataset preparation, we consider only 231 basic Amharic character images. In our dataset, each character exists 376 times on average. Once we developed the recognition model with the synthetically generated Amharic character images, we used both synthetically generated and printed test sets of Amharic character images. The printed characters are about 10% of the total dataset and achieved average recognition accuracy of 96.02% while 92.71% of the synthetic character images are correctly predicted. As we observed from the

empirical results, confusion among visually similar character across rows and columns were the major challenges. Therefore, we need to develop an alternative method to handle such confusion among these visually similar Amharic characters. The next section, Section 5.2, presents an alternative OCR approach which is designed by leveraging the grapheme of basic Amharic characters on Fidel-Gebeta.

5.2 Amharic Character Grapheme-based OCR

This section presents a novel CNN-based approach for Amharic character image recognition, which is designed by leveraging the structure of Amharic grapheme and taking the advantage of multi-task learning. Of 317 Amharic characters [MJ07], the most widely used characters in Amharic script are basic characters which we consider, in this section, are arranged in a matrix structure with a 34 by 7 row-column order and illustrated in Figure 4.2. These Amharic characters could be decomposed into a consonant and a vowel. As a result of this consonant-vowel combination structure, Amharic characters lie within a matrix structure called Fidel-Gebeta. The rows and columns of Fidel-Gebeta correspond to a character's consonant and the vowel components, respectively. However, the character ፕ, on the last row of the Fidel-gebeta that is a lately introduced Amharic character, is not considered in our dataset. Thus, we only use the 33 by 7 matrix of Amharic character distribution of Fidel-Gebeta..

In section 5.1, we noted that the characters shape similarly across rows, and having large classes affect the performance of the OCR model developed for Amharic character recognition and results from the OCR being complex. However, there is a point to be considered so as to overcome class size complexity and character similarity across rows of the Fidel-Gebeta. Based on the structure of Fidel-Gebeta, dividing the number of large classes into two smaller subclasses (row-wise and column-wise classes).

To the best of our knowledge, no attempts have been made to take this advantage and recognize a character based on the row-column order of Amharic character in Fidel-Gebeta which has motivated us to work on it. Therefore, in this section, we propose a method called Factored Convolutional Neural Network (FCNN) for Amharic OCR. FCNN has two classifiers that share a common feature space before they fork-out at their task-specific layer, where the two task-specific layers are responsible to detect the row and column compo-

nents of a character. The following section first presents a generic character-level OCR model developed, using classical and deep learning algorithms, for various scripts including Amharic, and then the proposed script-specific system architecture designed for Amharic OCR is described in detail.

5.2.1 Character-level OCR

Extensive research efforts have been made to develop an OCR model for various scripts and most of these scripts, now, have robust OCR applications while other low-resource scripts, like Amharic, are still remained relatively unexplored [BHS18]. This section presents various character-level OCR works that have been done, using either classical machine learning or deep learning-based techniques, for different scripts including Amharic.

The first work was done on Amharic OCR by Worku in 1997 and he adopted a tree classification scheme built by using topological features of a character [Ale97]. It was only able to recognize an Amharic character written with $Washera$ font and 12 point type. Then other attempts have been made ranging from typewritten [Tef99], machine-printed [MJ07], Amharic braille document image recognition [HA17], Amharic document image recognition and retrieval [Mes08], numeric recognition [RRB18] to handwritten [AB09]. However, all these attempts are based on classical machine learning techniques and trained with a limited number of private datasets. In addition, several other works have been made based on CNN-based methods together with the classical approaches. Maitra et al [MBP15] proposed a CNN as a trainable feature extractor and SVM classifier for numeric character recognition in multiple scripts. A similar study, Elleuch et al [ETK16] employed a CNN as a feature extractor and SVM as recognizer to classify Arabic handwritten script using the HACDB database.

Even though attempts have done for Amharic OCR so far are based on traditional machine learning techniques, in section 5.1 of this thesis, a CNN-based architecture for Amharic character image recognition is presented. Besides, all of these methods mentioned above consider each character as a single class [MJ07, Tef99, BHS18] but not consider the row and column order location of the character in Fidel-Gebeta. The problems of these approaches are; treated structurally similar characters in a different class and using a large number of classes which leads the network to be complex. It is also difficult to obtain a good performance when considering each structurally similar character as a class and without explicitly using the shared information, in

Fidel-Gebeta, between characters. The following section presents the proposed FCNN model.

5.2.2 Research Methods

There are very few databases used in various works on Amharic OCR reported in the literature. As reported in [Tef99], the authors considered the most frequently used Amharic character with a small number of samples, which are about 5172 core Amharic characters with 16 sample images per class. Later work done by million et al [Mes08] uses 76800 character images with different fonts type, sizes with 231 classes. Other researchers' work on Amharic OCR [Ale97, AB09, RRB18] reported as they used their own private database, but none of them made their database publicly available for research purpose.

In this section, we use the same Amharic character image database employed in section 5.1 that contains 80000 sample Amharic character images. In this database, there are many deformed and meaningless character images. Therefore, for our experiment, we removed about 2006 distorted character images from the database. We also labeled each character image in a row-column order following the arrangements of Fidel-Gebeta.

Sample Amharic characters from Fidel-Gebeta and shape difference across their corresponding vowels in row-wise order is depicted in Figure 5.3.

5.2.3 Multi-Task Learning

This section describes the Multi-Task Learning (MTL) paradigm, one of the central training paradigms in machine learning, that can learn and solve multiple and related tasks at the same time while exploiting commonalities and differences across all the tasks [DCBC15, DHK13]. Learning multiple related tasks simultaneously is inspired by the biological nature of learning in humans, where we often apply the knowledge we have already acquired by learning-related tasks from the environment so as to learn new tasks in the same and/or different environment. MTL has been applied in many domains. Zhang et al [ZLLT14] propose a convolutional neural network to find facial landmarks and then recognize face attributes. Kisilev et al [KSBH16] present a multi-task CNN approach for detection and semantic description of lesions in diagnostic images. Yang et al [YZX+18] introduce a multi-task learning algorithm for cross-domain image captioning. The common part in all these re-

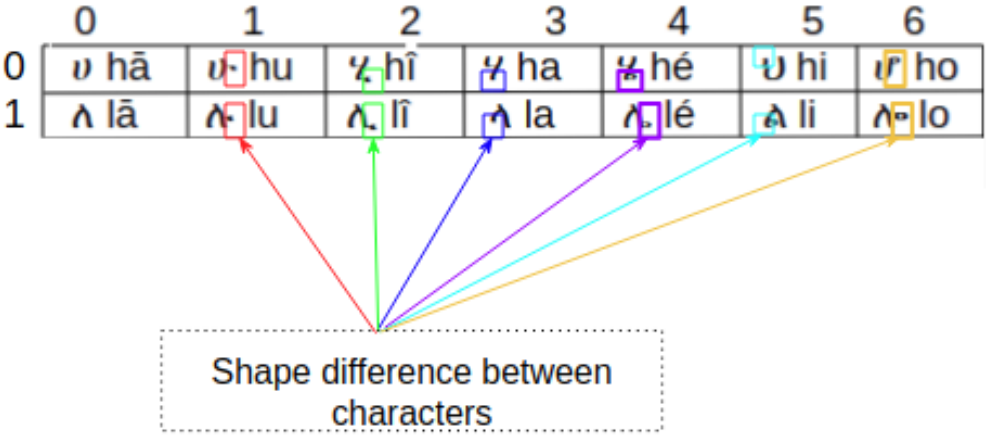


Figure 5.3: **Shape changes in row-column structure of Fidel-Gebeta.** *We call the first entries of each row the base character. Each row has its own base character, as a result it has a unique shape. Each column has a unique "pattern" of modification it applies to the base characters. For instance the third column marked by green box adds a dash at the right bottom, the fourth column marked by blue box makes the left leg of the character shorter, the fifth column marked by violet box adds circle at the right leg and so on.*

searchers' work is to use the same parameters for the bottom layers of the deep neural network while task-specific parameters at the top layers. As explained by Rgyriou et al [AEP07], learning multiple related tasks simultaneously has been often shown significantly better performance relative to learning each task independently. Compared to training tasks separately, MTL improves learning efficiency and prediction accuracy for the task-specific models. MTL provides various services such as implicit data augmentation, attention focus, and eavesdropping [AM90].

MTL could follow different training strategies with the objective of improving the performance of all the tasks [AEP07, ZY17]. Joint training is one of the most commonly used training strategies, where the model uses hard parameter sharing and tried to optimize a single joint loss L which is computed from each task using,

$$L = \sum_{i=1}^{N} l_i \tag{5.2}$$

where l_i is the loss of a task T_i and N is the total number of tasks involved in the training procedure. The character recognition model proposed in this section is designed by taking the advantages and learning strategies of MTL.

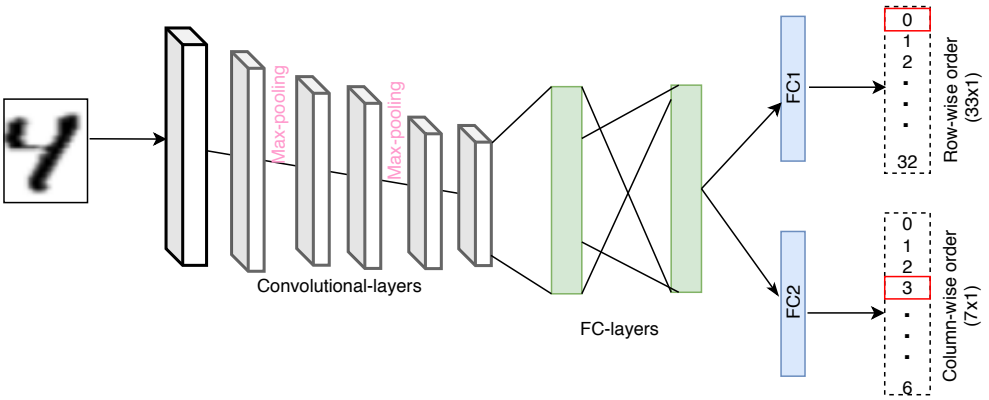


Figure 5.4: **Factored CNN architecture:.** *This method has two classifiers which share a common feature space at the lower layer and then fork-out at their last layers. FC1 and FC2, at the last layer, represent the fully connected layers of row and column detector having 33 and 7 neurons that corresponding to the number of classes for each task respectively.*

The details of the proposed system architecture designed, based on MTL, for Amharic character image recognition are described in the next section.

5.2.4 The Proposed FCNN Architecture

The overall framework of the proposed approach is shown in Figure 5.4. In this architecture, we employ six 3×3 convolutional layers with rectified linear unit (ReLU) activation function and two stacked fully connected layers in common for both classifiers and fork-out by adding one fully connected layer with soft-max activation function for each classification branch which has 33 neurons for row detector and 7 neurons for column detector branch. The final output of each task is determined by a Soft-max function using Equation (5.1).

In the proposed method, we jointly trained the two tasks (row detector and column detector) by taking the advantage of multi-task learning. In such a way, when we train the network, the parameters of the row detector layer should not change no matter how wrong we get in the column detector, and vice versa, but the parameters of the shared layer changes with both tasks since we only optimize a single joint loss which is formulated as

$$l_{joint} = l_r + l_c \tag{5.3}$$

Where l_r and l_c denotes different losses of row and column detector which

can be computed by Equation (5.4).

$$loss = -\sum_{i}^{C} t_i log(s_i) \tag{5.4}$$

Where C is the number of class, t_i is ground truth for each class and s_i is a score of each class calculated by Equation (5.1).

The overall character recognition accuracy of the model is determined by the recognition accuracy of both tasks. In this case, a single character is correctly recognized when the row and column components of a character are correctly detected by both classifiers simultaneously. The index of a character in Fidel-Gebeta can be computed from a row and a column component values of a character as:

$$I_c = a \times 7 + b \tag{5.5}$$

where I_c is the character index, a and b are the row and column order values of the character in Fidel-Gebeta.

The total number of characters correctly recognized by the proposed method depends on the number of times in which both row and column detector networks are correct, for each character, simultaneously. Then the recognition performance of the proposed model is measured using accuracy and can be formulated as Equation (5.6).

$$A = \left(\frac{n(S_r \bigcap S_c)}{t} \right) \times 100 \tag{5.6}$$

where A is overall accuracy, $\bigcap$ is set intersection, S_r and S_c are samples correctly recognized by the row and column detector respectively, t is number of test samples and $n(S_r \bigcap S_c)$ is total number of samples that are mutual detected as correct by the row and column detector.

5.2.5 Experimental Results

Experiments were run following the network architecture introduced in section 5.2.4. The input sample character is recognized in terms of its row and column components, as shown in Figure 5.5.

We carry out our experiments on the Amharic database employed in section 5.1 by removing distorted and meaningless images. Then 77,994 character images that contain on average 2364 samples from each row of Fidel-Gebeta

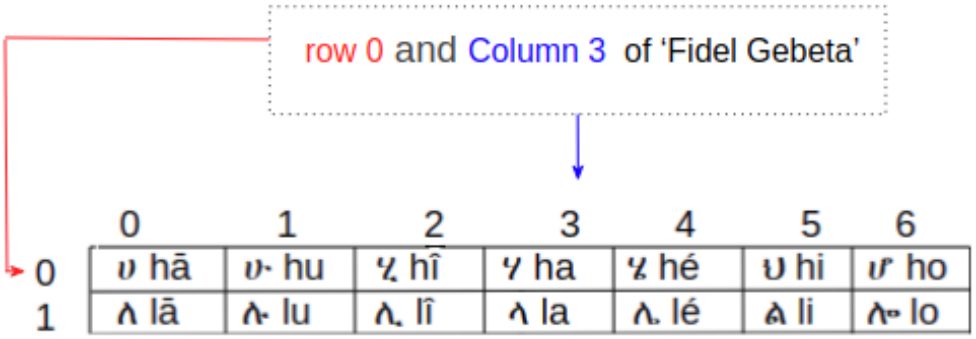


Figure 5.5: **A typical diagram that shows the prediction of FCNN model.** *For an input character image, the row component detector recognizes zero and the column component detector recognizes three then the model output becomes the character (ሃ) which is located at row zero and column three of the Fidel-Gebeta*

were used for training and testing. The proposed FCNN architecture contains two multi-class classification problems (the row and column detector) and the convolutional layers have 32 filters of size 3×3. Every two blocks of convolutional layers are followed by a 2×2 max-pooling. The two fully connected layers have 512 neurons each while the last forked layers have a size corresponding to the number of unique classes in each specific task, in our case 33 for row detector and 7 for column/vowel component detector. Considering the size of the dataset, we train our network using the different batch sizes and the best test result is recorded with the batch size of 256 and Adam optimizer running for 15 epochs. Based on the results recorded during experimentation,

Table 5.2: Performance of prior works (CNN) and proposed method(FCNN)

	Dataset size	**Recognition**	**Accuracy**
CNN [BHS18]	80,000	character	92.71%
FCNN	77,994	Row-column	94.97%

96.61 % of the row and 95.96% of the column components of the characters are correctly detected by the row detector and column detector respectively. An overall character recognition accuracy of 94.97% was recorded and the training converges faster since we reduce the number of classes from 231 to 40 classes (33 row-wise and 7 column-wise classes). As can be observed in Table 5.2, we achieved better recognition performance compared with works done on Amharic OCR using 231 classes. The value 'character' in the third column of Table 5.2 refers to the recognition made with 231 classes while

'row-column refers to recognition with two smaller subclasses, which is done by dividing 231 classes into (33 row-wise and 7 column-wise classes).

5.2.6 Conclusions

In this section, we have introduced a novel CNN-based method, for Amharic character image recognition. The architecture consists of two classifiers: the first is responsible to detect a row component and then the second is responsible to detect the column component of a character in Fidel-Gebeta. A character is correctly recognized when both detectors are correctly identified in its row and column components. We train and evaluate our Factored CNN-based recognition model on a synthetically generated Amharic character database that has 77994 sample Amharic character images. Empirical results show that our proposed method achieved state-of-the-art performance.

Furthermore, our proposed method minimizes the problems of class size complexity and row-wise character similarly by considering the structural arrangement of Amharic characters in Fidel-Gebeta and then splitting the classes into two smaller subclasses. As observed from the empirical results, the confusion between visually similar characters such as ሸ and ሽ , ለ and ሉ which are found in row 1 and 6, 7 and 33 of Fidel-Gebeta respectively is greatly reduced and therefore row-wise character similarity is not now a challenge for this method. Besides, the performance of the new model is far better than the standard CNN-based OCR model.

5.3 Chapter Summary

This chapter discusses the application of CNN-based OCR methodology for Amharic character image recognition. In this regard, this chapter introduces a novel methodology for Amharic character recognition. A Factored CNN-based character recognizer is designed by following the architecture of multi-task learning and has been trained to learn class associations of individual characters based on their grapheme on Fidel-Gebeta. Experimental results show that the new factored CNN-based model outperforms the other CNN-based model for Amharic character image recognition. It also minimizes the problems of class size complexity and row-wise character similarly that have been limiting the recognition performance of the OCR models designed for Amharic character recognition. In general, empirical results on both synthetic

and printed Amharic characters, convolutional neural network-based OCR models are significantly better than classical machine learning approaches. In addition to these character-level Amharic OCR methods, the next chapters of this thesis describe various state-of-the-art sequence-to-sequence methods proposed and implemented for Amharic script where the recognition is at a text-line level.

Chapter 6

Amharic Script Recognition using CTC Sequence Modeling

In the classical OCR approach, different handcrafted features have been extracted from segmented components of a given script, and then pattern classification algorithms were employed for recognition. However, contemporary deep learning techniques can learn the distinguishing features automatically; thereby eliminating the manual feature extraction steps. Sequence learning is based on the sequence-to-sequence matching principle where a neural network is trained such that both the input-output pairs are comprised of variable length sequences. This part of the thesis overviews state-of-the-art segmentation-free OCR techniques and contains two sections. Each section provides a detailed analysis of the corresponding OCR techniques and reports their recognition performance on Amharic text-image recognition.

Further this chapter is organized as follows: Section 6.1 describes the use of LSTM-CTC-based networks for Amharic OCR. Section 6.2 presents the details of a hybrid model based on the CNN and LSTM networks for Amharic text-line image recognition. Section 6.3 concludes the chapter with a short summary.

6.1 Amharic Text-image Recognition with LSTM-CTC Networks

This section provides necessary details about the LSTM-based OCR methodology that has been used for Amharic text-image recognition and detail anal-

ysis is also presented based on the results obtained from experimentation.

6.1.1 Related Work

Over decades of research in OCR, various techniques have been proposed to carry out the text-line image recognition task for multiple scripts and even most of the scripts, now, have workable OCR systems. Some of these works are presented in chapter 5 of this thesis. Further, deep learning-based systems have demonstrated ground-breaking performance on image recognition [HZRS16], sequence prediction and labeling [LGF+07, GLB+08]. Based on deep neural network, many segmentation free OCR techniques have been studied and impressive progress has been made for many Latin and Non-Latin scripts of both printed and handwritten such as Bidirectional Recurrent Neural Network with Long Short Memory (LSTM) architecture for online handwritten recognition [LGF+07], Convolutional Recurrent Neural Network (CRNN) for Japanese handwritten recognition [LNNN17], segmentation free Chinese handwritten text recognition [ML15], a hybrid Convolutional-LSTM for text image recognition [Bre17], Multidimensional-LSTM for Chinese handwritten recognition [WYCL17], combined Connectionist Temporal Classification (CTC) with Bidirectional LSTM (BLSTM) for unconstrained on-line handwritten recognition [GLB+08].

Research attempts, on Amharic OCR, have been made to develop Amharic OCR starting from 1997 using different statistical machine learning techniques [MJ07] [Ale97, CH03, Tef99]. Following the success of deep learning, other attempts have also been made for Amharic character recognition employing Convolutional Neural Network (CNN) [BHS18, RRB18, BHL+19b, GSTBJ19]. As presented in chapter 5 in detail, nearly all attempts have performed segmentation of text image into character level which directly affects the performance of the OCR. The only exception is Assabie [AB11], proposed an HMM-based model for offline handwritten Amharic word recognition without character segmentation using structural features of characters as a building block of the recognition system. Further, the research attempts done for Amharic OCR so far neither use enough training data nor considering all possible characters of Amharic script in their experiment [BHS18].

There are many effective off-the-shelf commercial and open source OCR applications for many languages including the functionality of a ground truth generator from the exiting printed text and even a synthetic handwritten training data generator OCR engines [GSR15, RBO17]. The effectiveness of var-

ious open-source OCR engines was evaluated on 19^{th} century Fraktur scripts and the evaluation shows as the open-source OCR engine can outperform the commercial OCR application [RSWP18]. However, the OCR system for many scripts especially those which are indigenous to the African continent (such as Amharic) remains under-researched and none of the researchers take the advantage of deep learning techniques such as LSTM, used for many languages, for developing Amharic OCR.

Therefore, in this section of the thesis, we propose a recurrent neural network based on Bidirectional-LSTM architecture with Connectionist Temporal Classification (BiLSTM-CTC) for Amharic text-line image recognition. In addition, we introduce a technique for the preparation of printed Amharic datasets.

6.1.2 Database

As explained in chapter 4, there are few datasets of Amharic script reported in the literature. for example, as reported in [Tef99], the authors considered 5172 images of the most frequently used Amharic characters. A later work by Million et al. [MJ07] uses 76,800 character images with different font types and sizes which belong to 231 classes. Other researchers' work on Amharic OCR [Ass02, RRB18, Ale97] reported that they used their own private databases, but none of them were made their dataset publicly available. Therefore, the shortage of datasets has been continued as the main challenge and one of the limiting factors in developing reliable OCR systems for Amharic script to date. Synthetic data generation has become a surrogate technique and solves the shortage of training datasets to some extent. However, to date, getting a real training dataset is one of the limiting factors in developing a reliable system in the area of image processing and pattern recognition.

Therefore, in this section, Amharic text-line image database presented in chapter 4 section 4.3.2 is employed for both training and model performance evaluation. The database consists 337,337 Amharic text-line images. The detail statistics of this database is summarized in Table 6.1.

6.1.3 The Proposed Model Architecture

The overall network architecture of the proposed model, which is adapted from the existing work for other scripts [SUHP$^+$15, UHAR$^+$13], is depicted

Table 6.1: The detail of Amharic text-line images database

	Printed	**Synthetic**	
Font type	Power Geez	Power Geez	Visual Geez
Number of samples	40,929	197,484	98,924
No. of test-samples	2,907	9,245	6,479
No. of train-samples	38,022	188,239	92,445
No. of unique chars.	280	261	210

in Figure 6.1. In a general pipeline of text-line image recognition, the input of the model is an image containing a line of texts and the output is the corresponding strings of texts. In this architecture, a grey-scale text-line image used as an input of BLSTM network, and the output of BLSTM is fed into a soft-max layer which has *n+1* nodes, where each node is corresponding to a label or each unique character in the GT including one blank character. In the ADOCR database, there are 280 unique Amharic characters. Therefore, the soft-max outputs 281 probabilities at each time step.

The CTC objective function can automatically learn the alignment between the input and output sequence [GFGS06], where the CTC loss function is used to train the network. During training, CTC requires only an input sequence and the corresponding transcription. The detail formulation of the CTC cost function explained in chapter 2, section 2.4.1.

Once the probability of label sequence y from an input sequence x is obtained with the CTC forward-backward algorithm proposed by [GFGS06]. We employ the best path decoding method to find a character (C) that has the highest score from outputs (i) at every time-step and the final recognized string text can be generated like the example given in chapter 2, section 2.4.1, without segmentation of the input sequence and formulated as Equation (6.1).

$$C_{\text{max}} = B(\arg\max_{i}(y_i^t)), \text{for } t = 1, 2, ..., T \tag{6.1}$$

In all experiments described below, we use the same network architecture and the performance of the proposed model is calculated in terms of character error rate which is computed by counting the characters inserted, substituted, and deleted in each sequence and then divided by the total number of characters in the ground-truth (see Equation (6.2)).

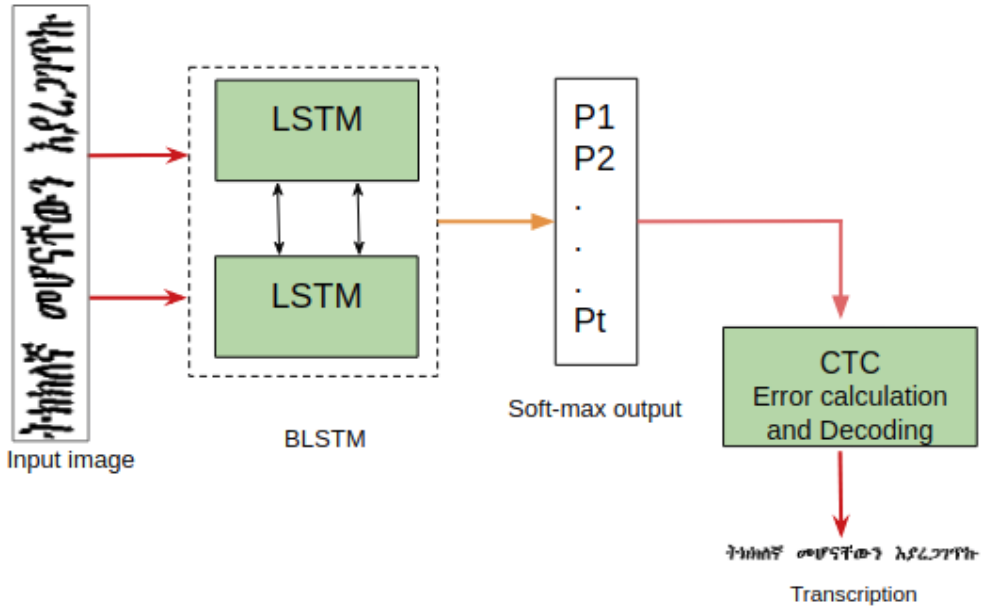


Figure 6.1: **The proposed network architecture.** *This network consists of two components. The first layer is Recurrent layers as a sequence learner and then the soft-max function that predicts a label distribution of each sequence. The second layer is CTC layer which takes the soft-max prediction as input and transcribes these predictions to final label sequences. All the input text-line images length is normalized to the fixed height of 48 pixels. A raw pixel value of the 1D sequence is extracted by traversing a* 48×1 *time window over a text-line image and fed to BLSTM network as input. A character with the highest score at each time step is computed using Equation* (6.1) *and the final string of a text is generated using the reduction process.*

6.1.4 Experiments

Experiments are conducted using, ADOCR, a database of Amharic text images which consists of both printed and synthetically generated Amharic text-line images. To select suitable network parameters, different values of these parameters were considered and tuned during experimentation and the results reported, in this section, were conducted using Adam optimizer, and the model is trained to minimize the CTC-based loss. We employed the BLSTM network with two networks hidden layers size of 128 each and a learning rate of 0.001. Once we implement the model with Keras Application Program Interface (API) on a TensorFlow backend, two experiments are undertaken so as to evaluate the performance of the model against two types of test datasets of

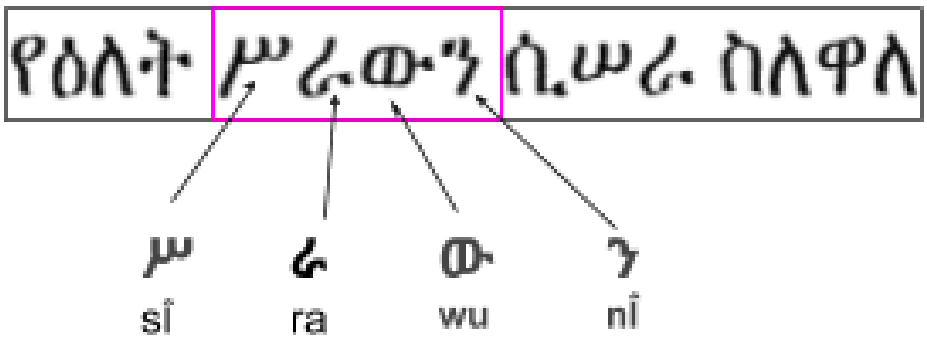


Figure 6.2: **Sample text-line image for illustration of Amharic script.** *A word marked by violet box is composed from four individual Amharic characters.*

the ADOCR. The first experiment is testing our model with synthetic Amharic text-line images generated with *Power Geez* and *Visual Geez* fonts. The second experiment is used to test how the model, that trained with a synthetic and part of printed text-line images, works on printed text-line images written with *Power Geez* font type. Sample Amharic text-line image taken from ADOCR database is illustrated in Figure 6.2.

We randomly split the training and test-set. We use a total of 318,706 training and 18,631 samples for testing. For validation, we use 7% of randomly selected images from the training dataset. The detail information of the database used for our experiment is shown in Table 6.1. The training and validation loss of the proposed model recorded for 50 epochs is depicted in Figure 6.3. The network was trained for 50 epochs with a batch size of 200 and then a character error rate of 4.24% on synthetically generated text-line images with *Visual Geez* font, 8.54% on printed text-lines with *Power Geez* font, and 2.28% on synthetic text-line images generated with *Power Geez* font are recorded. The results recorded during experimentation are summarized in Table 6.2. The performance of the proposed model is measured using Character Error Rate (CER) formulated as follows:

$$CER(P,T) = \left(\frac{1}{q}\sum_{n\in P, m\in T} D(n,m)\right) \times 100 \tag{6.2}$$

where q is the total number of target labels, P and T are the predicted and ground-truth labels, and $D(n,m)$ is the edit distance between sequence n and m. Edit distance is the difference between the predicted and ground-truth texts

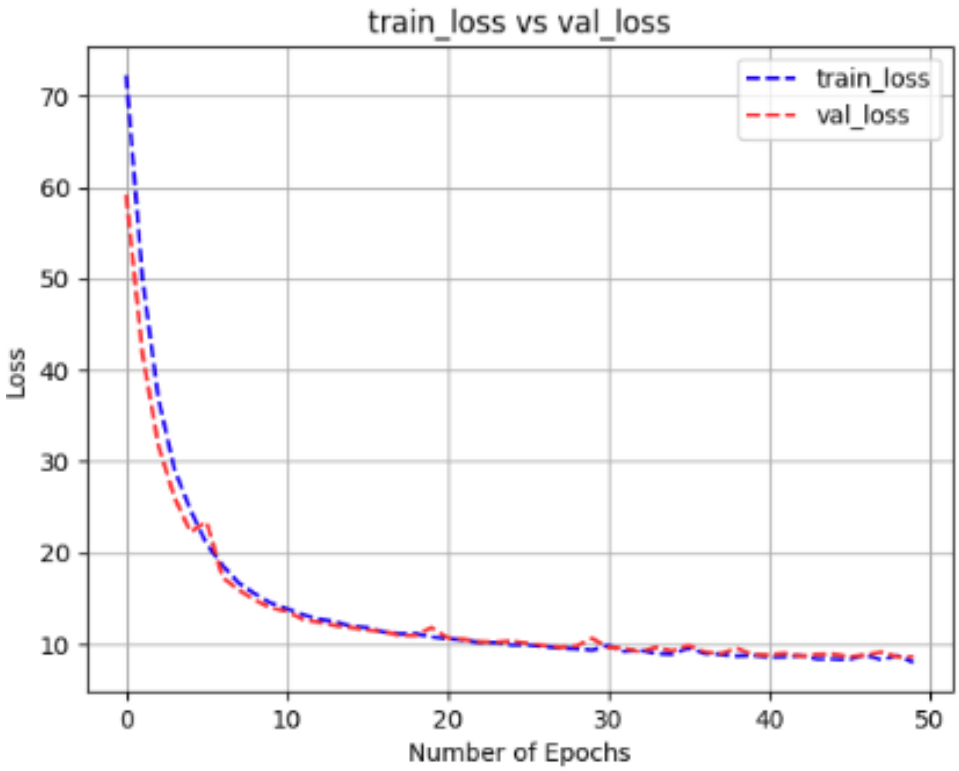


Figure 6.3: **Training versus validation loss of an LSTM-CTC model.**

which can be occurred due to the deletion, substitution and insertion of characters during recognition. The results show that using a synthetic dataset is the

Table 6.2: A summary of experimental results (CER) of LSTM-CTC model

	Font type	CER(%)
Printed	Power Geez	8.54
Synthetic	Power Geez	4.24
	Visual Geez	2.28

best alternative solution to overcome the shortage of labeled data in general and specifically for Amharic OCR application which has no publicly available dataset for research purposes. In addition, the model requires enough data generated from multiple fonts so as to perform better on varieties of test data.

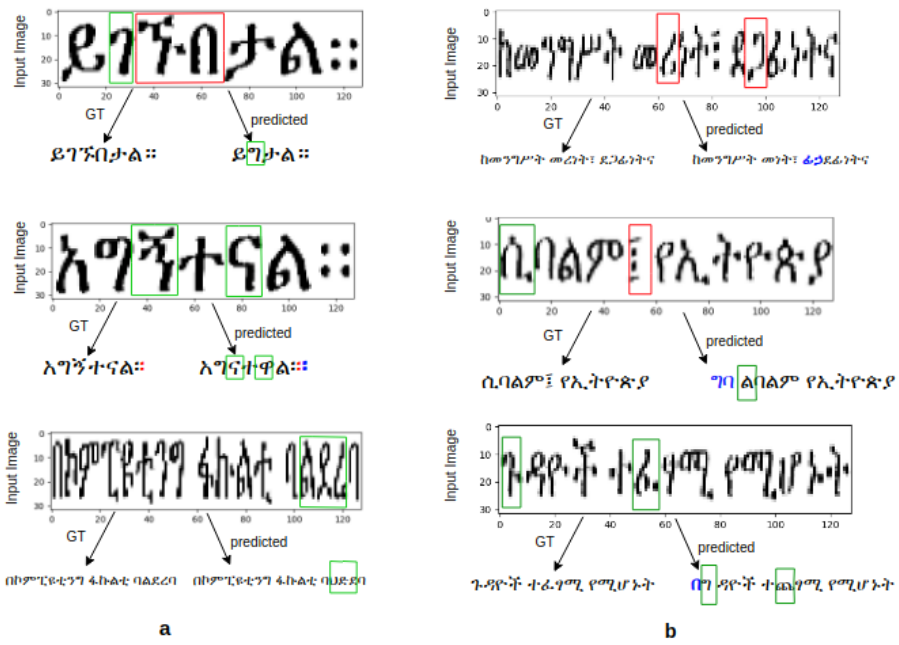


Figure 6.4: **Sample miss-recognized characters in each text-line image of the test dataset.** *(a) Printed text-line images. (b) Synthetically generated text-line images. The characters marked by redbox are sample miss-recognized (substituted with other characters), characters marked with green box are sample deleted characters and characters marked with violet box are sample inserted characters which are generally called the substitution, deletion and/or insertion errors, in this thesis, respectively. As we observed during experimentation, most of the substitution errors are occurred on the synthetic test images while most of insertion and deletion errors are observed on printed test images.*

6.1.5 Analysis of the Result

Sample Amharic text-line images and characters miss-recognized during testing are shown in Figure 6.4. In this figure, sample output texts containing different types of character errors that may occur due to misspelled characters, spurious symbols, or lost characters are presented. The proposed model works better on a synthetic test dataset compared to the character error rate observed on printed test data. The generalization of this model, especially on printed data, is unsatisfactory. This happens mainly because of a significantly smaller number of printed text-line images and their nature in the training samples.

For example, as illustrated on Figure 6.4a, on the top part of the figure, the

character (ገ) marked by the green rectangle in the input image is substituted by the character (ግ) in the predicted text. On the other hand, the two characters (ጉ and በ) marked with the red rectangle are deleted characters. Other character errors are depicted on the top part of Figure 6.4b; characters (ፈ and ጋ) from the input image, marked with the red rectangles, are deleted characters, while characters (ፊ and ኃ) in the predicted text, written with a blue color, are inserted characters.

To the best of our knowledge, there is no similar work which is directly relevant to compare the current work with other Amharic OCR model since other works are performed under different experimental conditions and datasets. Therefore, we don't compare our results with others' work. In general, the performance of the proposed method obtained promising results compared to attempts done on Amharic OCR using segmentation-based traditional methods and this work will be used as a benchmark for future works on Amharic OCR.

6.1.6 Conclusions

In this section, we present a recurrent network based on a Bidirectional LSTM network architecture with a CTC objective function for Amharic text-line image recognition motivated by its success on the task of sequence labeling. We evaluate our model both on synthetically generated and printed Amharic text-line images, from the ADOCR Amharic database, and achieved state-of-the-art results. The next section presents an extension of this model by integrating CNNs before the LSTMs as a feature extractor.

6.2 Deep CNN-RNN Hybrid Networks for Amharic OCR

In this section, we introduce an end-to-end Amharic text-line image recognition approach based on hybrid neural network architecture. Motivated by the recent success of end-to-end learning in pattern recognition, we propose a model which integrates a feature extractor, sequence learner, and transcriber in a unified module and then is trained in an end-to-end fashion. In the proposed CNN-RNN hybrid network; the CNN layer is employed at the higher level as a feature extractor, LSTM networks have been used as a sequence learner of text-line images where the target labels are the character of Amharic

scripts. Since text-line image recognition is a Spatio-temporal pattern recognition problem, the proposed model, in this section, consists of CNN and LSTM networks together with CTC. This network architecture is proposed as an extension work of the section 6.1 with the following summarized contributions:

- The proposed CNN-based feature extractor module is stacked before LSTM layer and used to extract automatic features from text-line images
- We adopt an end-to-end trainable neural network for Amharic text-line image recognition that achieves state-of-the-art results.
- Based on the experimental results obtained, a detailed analysis of the dataset is presented.

Our hypothesis in this section is the hybrid of CNN-LSTM networks is better than LSTM in handling sequential tasks that are both spatially and temporally deep, like text-line images. This hypothesis is based on the results reported in the literature; thus CNNs are good at reducing frequency variations in spatial structure, like images, while LSTM networks are designed for interpreting the features across time steps and are good at temporal modeling. To justify our hypothesis, a hybrid CNN-LSTM network is trained by stacking CNN layers on the front end and then followed by LSTM layers with a fully connected layer and CTC layer on the output. The proposed model has the flexibility to be applied to a variety of tasks that involve data with sequential inputs and outputs. Therefore, in this work, we take advantage of the complementarity of CNNs and LSTMs by combining them into one unified architecture.

6.2.1 Database

Since there is no other publicly available dataset for Amharic script recognition, to train and evaluate the performance of our OCR model, we use the same database, employed in section 6.1. In the original dataset, ADOCR, all text-line images are Greyscale and the sizes of each text-line were 48 by 128 pixels. Considering similar works done in the area, and to reduce computational costs during training, we resized the images into sizes of 32 by 128 pixels. Sample text-line images that are normalized to a size of 32 by 128 pixel and used for training and testing the model proposed in this section are illustrated in Figure 6.5.

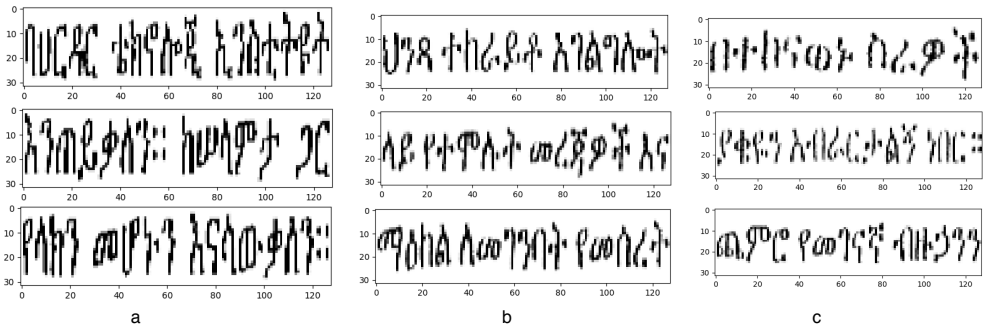


Figure 6.5: **Sample Amharic text-line images from ADOCR database.** *All images in this figure are normalized to a size of 32 by 128 pixels: (**a**) Printed text-line images written with the Power Ge'ez font type. (**b**) Synthetically generated text-line images with the Visual Ge'ez font type. (**c**) Synthetically generated text-line images with the Power Ge'ez font type.*

6.2.2 The Proposed Hybrid CNN/LSTM Network Architecture and Parameters

Since CNNs and RNNs are the most suited for image-based problems [HKR15] and sequence recognition [GVK19] respectively, we proposed a hybrid network framework which is illustrated in Figure 6.6. In this framework, we employed three modules; the feature extractor, the sequence learner, and the transcriber module. All three of these modules are integrated into a single framework and trained in an end-to-end fashion.

The feature extractor module consists of seven convolutional layers which have a kernel size of 3×3, except the one that is stacked on top with a 2×2 kernel size; each uses rectifier linear unit (ReLU) activation, four max-pooling layers with pool sizes of 2 for the first pooling layer, and 2×1 for the remaining pooling layers. Strides are fixed to one, and the 'same' padding is used in all convolutional layers. The number of feature maps is gradually increased from 64 to 512, while the image size is quickly reduced by a spatial pooling to further increase the depth of the network. For an input size N, kernel K, padding P, and stride S, the output size M at each convolutional layer can be computed as $M = (\frac{N-K+2*P}{S})+1$. In addition, to normalize the output of the previous activation layer and accelerate the training process, batch normalization is used after the fifth and sixth convolutional layers.

We also used a reshape function to make the output of the convolutional layer compatible with the LSTM layer. The sequence learner module is the

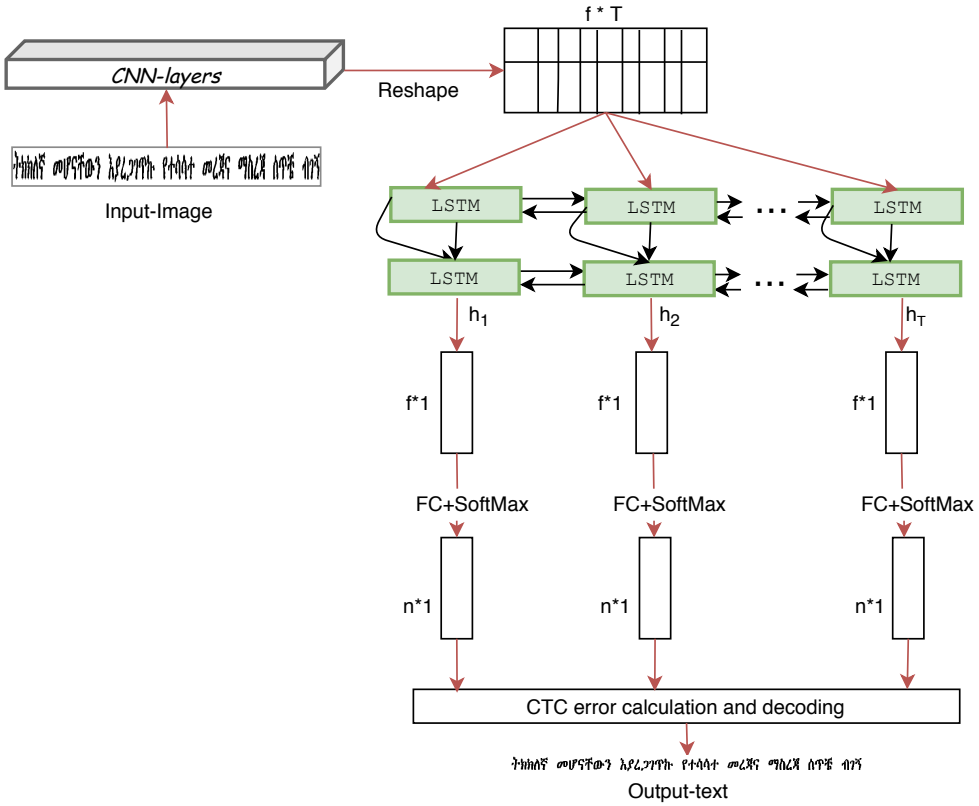


Figure 6.6: **The proposed CNN-LSTM model.** *This network consists of three components. The first component is the convolutional layer which acts as the feature extractor module, the second component is the recurrent network layer which takes the outputs of CNNs after it is reshaped to* $f \times t$ *and it outputs* $f \times 1$ *feature vector for each time step. Then the fully connected networks (FC) with soft-max function are applied to predict a label distribution, including a blank token, on* $n + 1$ *number of characters at each time-step. The third component is a Connectionist Temporal Classification (CTC) layer which takes the soft-max prediction at each time-step as input and transcribes these predictions into final label sequences. All input text-line images length is normalized to a fixed size of 32 by 128 pixels, and labels are padded with zero until they reach the maximum sequence length of 32.*

Table 6.3: Convolutional network layers of the proposed model and their corresponding parameter values for an input image size $32 \times 128 \times 1$.

Network Layers	Kernel Size	Stride	Feature Maps
Convolution	3×3	1×1	64
Max-Pooling	2×2	2×2	-
Convolution	3×3	1×1	128
Max-Pooling	2×1	1×1	-
Convolution	3×3	1×1	256
Convolution	3×3	1×1	256
Max-Pooling	2×1	1×1	-
Convolution	3×3	1×1	256
BatchNormalization	-	-	-
Convolution	3×3	1×1	256
BatchNormalization	-	-	-
Max-Pooling	2×1	1×1	-
Convolution	3×3	1×1	512

middle layer of our framework which predicts the sequential output per time-step. This module consists of two bidirectional LSTM layers, with the softmax function on top, each of which has 128 hidden layer sizes and a dropout rate of 0.25. The sequential output of each time-step from the LSTM layers is fed into a softmax layer to get a probability distribution over the $n + 1$ possible characters. Finally, transcription of the equivalent characters is done using the CTC layer. The details of the network parameters and configuration of the proposed model are depicted in Table 6.3 (configuration of CNN layers) and Table 6.4 (configuration of LSTM layers).

During training, the input text-line image passes through the convolutional layers, in which several filters extract features from the input images. After passing some convolutional layers in sequence, we apply to reshape operation on the output and obtained the sequence of 63 vectors of 512 elements. Then we feed these 63 vectors to the LSTM network and get its output which also the vectors of 512 elements. The output of the LSTM is fed into a fully connected network with the soft-max function which has $n + 1$ nodes with a vector of 281 elements. This vector contains the probability distribution of observing character symbols at each time step. Each symbol corresponds to a label or each unique character in the ground truth and one blank character which is used to take care of the continuous occurrence of the same characters. We employed a checkpoint strategy that can save the model weight to the same file each time an improvement is observed in validation loss.

Table 6.4: **The recurrent network layers of the proposed model with their corresponding parameter values.** *The input size of the Long-Short-Term Memory (LSTM) is a squeezed output of the convolutional layers, which is depicted in Table 6.3.*

Network Layers (Type)	Hidden Layer Size
BLSTM	128
BLSTM	128
Soft-Max	No. class = 281

Once the probability of label sequence y from an input sequence x is obtained with the CTC forward-backward algorithm proposed by [GFGS06], we employ the best path decoding method, fast and simple, to find a character (C) that has the highest score from outputs (i) at every time-step; the final recognized string text can be generated using B (see the formulation in chapter 2, section 2.3 Equation 2.7) without segmentation of the input sequence.

In all results reported in this section, the performance of the proposed model is described in terms of Character Error Rate (CER) (see Equation 6.2), which is computed by counting the number of characters inserted, substituted, and deleted in each sequence and then dividing by the total number of characters in the ground truth.

6.2.3 Experimental Results

During training, to select suitable network parameters, different values of these parameters were considered and tuned, and the results reported in this section were obtained using an Adam optimizer employing a convolutional neural network with a feature map that started from 64 and increased to 512, the BLSTM network with two network hidden layers with sizes of 128 each, and a learning rate of 0.001.

Based on the nature of the dataset, we conducted three experiments to evaluate the performance of the proposed model. Once we trained our network with the synthetic and some of the printed text-line images, the performance of the model was evaluated with three different test datasets from the ADOCR database. In the first and the second experiments, the model was evaluated with synthetic Amharic text-line images that are generated with the *Power Geez* and *Visual Geez* fonts, respectively. The third experiment was conducted to evaluate the recognition performance of the OCR model using a printed test dataset written with the *Power Geez* font type. For validation, we used 7% of

the training dataset, randomly selected, as proposed in section 6.1. The network was trained for 10 epochs with a batch size of 200. During the testing of

Table 6.5: Experimental results of CNN-LSTM OCR model in CER (%).

Image Type	Font Type	Test Data Size	CER (%)
Printed	Power Geez	2907	1.59
Synthetic	Visual Geez	6479	1.05
	Power Geez	9245	3.73

the proposed model, character error rates of 1.05% and 3.73% were recorded on the two tests of the ADOCR database which were generated synthetically using the *Visual Geez* and *Power Geez* fonts, respectively. The model was also tested with the third test dataset, which consists of printed text-line images that were written with the *Power Geez* fonts, and a character error rate of 1.59% was obtained. All empirical results recorded during experimentation are summarized in Table 6.5.

6.2.4 Discussion and Analysis of Results

This section provides a detailed analysis and description of the dataset first, and then the results obtained during experimentation are also presented. We performed repetitive evaluations of our method using the ADOCR benchmark dataset (see chapter 4, Section 4.3.2), and we also tried to compare with the state-of-the-art methods on both printed and synthetically generated datasets. Some of the printed text-line images are not properly aligned with the ground truth due to the occurrence of extra blank spaces between words and/or the merging together of more words during printing. In addition, with the synthetic text-line images, a character at the beginning and/or at the end of the word is missed, which results in misalignment with the ground-truth. To improve the recognition performance, it is important to annotate the data manually or to use better data annotation tools. Of several factors, samples with wrongly annotated Amharic text-line images and characters are the major factors causing recognition errors. In general, the recognition errors may occur due to misspelled characters, spurious symbols, or lost characters, and sample annotation errors per text-line image are illustrated in Figure 6.7.

The proposed model works better on the synthetic test datasets generated with the *Visual Geez* font type, compared to the character error rate observed on the *Power Geez* font type and the printed test data. The generalization of this model, especially on the synthetic dataset generated with the *Power Geez*

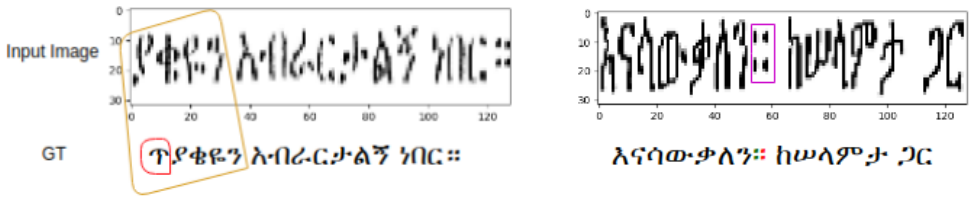


Figure 6.7: **Sample wrongly annotated images and GT from the test dataset.** *(**a**) Synthetic text-line image; the word marked with yellowrectangle is a sample mislabeled word where the first character (ጥ) in the GT, marked with a redcircle, is missed in the input image but it exists in GT. (**b**) Printed text-line image; a punctuation mark called a full stop/period, bounded with a purplerectangle in the input image, is incorrectly labeled as two other punctuation marks called word separators, indicated by the green and red text colors in the GT.*

font type, is not as good as the *Visual Geez* one. This happens mainly because of the significantly larger number of text-line images in the test set and the nature of the training samples (i.e., the text-line images and the ground truth are not properly annotated due to the existence of deformed characters and missing characters in the beginning and/or end of the text-line images during data generation but not in the ground truth). In addition, text-line images generated with the *Power Geez* font are relatively blurred, resulting in poor recognition accuracy.

6.2.5 Recognition Performance Comparison

As depicted in Figure 6.8 and Table 6.6, the performance of the proposed model is improved. Compared to an LSTM-based Amharic OCR, the proposed model achieved better recognition performance with a smaller number of epochs. However, the proposed model took a long time for training. This is due to the nature of end-to-end learning approaches [Gla17] that incorporate multiple and diverse network layers in a single unified framework. Therefore, the proposed model can be assessed and training time may be further improved by following some other concepts like decomposition [SSSS17] or the divide-and-conquer approach.

The comparisons among the proposed approach and others' attempts done on the ADOCR Database [BHL+19a] are presented in Table 6.6, and the performance of the proposed model shows better recognition results on all the

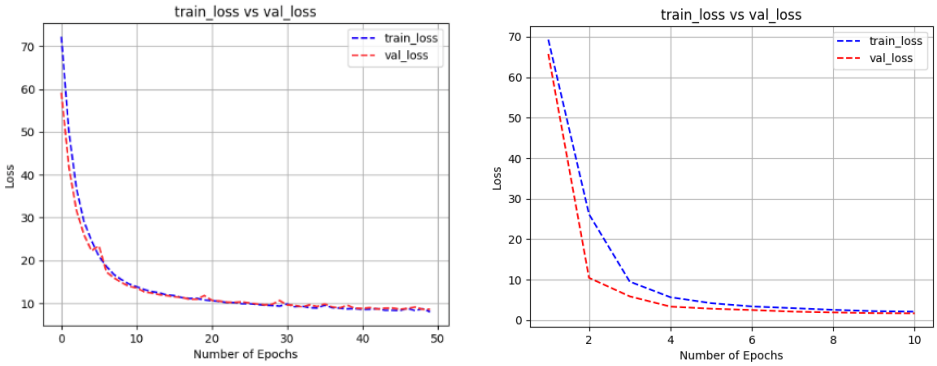


Figure 6.8: **Learning loss comparison.** (**a**) *The training and validation losses of LSTM-CTC model recorded for 50 epochs.* (**b**) *The training and validation loss of the proposed model recorded for 10 epochs.*

three test sets of the ADOCR database.

Table 6.6: Comparison of test results (CER).

	#Text-Lines	Image Type	Font Type	CER (%)
LSTM+CTC [ALT18] *	12 pages	printed	-	2.12%
LSTM+CTC[BHL+19a]	2,907	Printed	Power Geez	8.54%
LSTM+CTC [BHL+19a]	9,245	Synthetic	Power Geez	4.24%
LSTM+CTC [BHL+19a]	6,479	Synthetic	Visual Geez	2.28%
CNN+LSTM+CTC	2,907	Printed	Power Geez	1.56%
CNN+LSTM+CTC	9,245	Synthetic	Power Geez	3.73%
CNN+LSTM+CTC	6,479	Synthetic	Visual Geez	1.05%

* Denotes methods tested on different datasets.

6.2.6 Conclusions

In this section, we proposed a hybrid CNN-LSTM model as an extended version of the OCR model presented in section 6.1 of this chapter, where the new OCR model consists CNNs as the feature extractor, LSTMs as sequence learner, and CTC networks as the transcriber in a unified framework. All these three modules are trained in an end-to-end fashion. The recognition performance of the proposed model is evaluated using three different types of test datasets from the ADOCR Amharic database, and it outperforms the LSTM-CTC based OCR model by a large margin.

6.3 Chapter Summary

In this chapter, state-of-the-art techniques based on LSTM networks have investigated the newly introduced Amharic text-line images database, ADOCR. The contribution of this chapter is presented in two sections. First, a bidirectional-LSTM model is trained with CTC. Second, CNN layers are integrated before the LSTM layer and the networks are trained as a unified framework in an end-to-end fashion using the CTC objective function. The performance of both models is evaluated against the three test datasets from the ADOCR database and the results show that the performance of the recognition model is much better in the hybrid of CNN-LSTM network than the usual Bidirectional-LSTM networks. The generalization of these models proposed, in this chapter, for printed text-images is not as good as the synthetic one. This happens mainly because of the significantly smaller number of printed text-line images in the training set. We have found that the majority of the recognition errors on synthetically generated text-images are due to poor annotation of the dataset which usually could be deformed or missed characters during text-line image generation. Further, the CTC loss can be efficiently calculated by the forward-backward algorithm, but CTC still predicts target characters for every frame with the assumptions of conditional independence in target characters. Therefore, the next chapter proposes alternative methods, based on attention mechanism, for better recognition of Amharic text-line images.

Chapter 7

Attention-based Sequence Learning for Amharic OCR

RNNs trained with CTC objective function have been employed for text-image recognition and presented in the previous chapter in detail. It is also a state-of-the-art approach and widely applied for sequence learning tasks to date. Recently, another sequence learning technique has been emerged, from the field of NMT, and has often been shown to improve the performance over the existing approaches. Therefore, this chapter presents the other technical contributions of this thesis. These contributions are organized in two subsections and presented as follows. Section 7.1 describes the Attention mechanism and its implementation detail for Amharic text-image recognition; while section 7.2 shows the advantage of blending attention mechanism in to CTC objective function for sequence transcription in general and in particular for Amharic text-image recognition. Finally, it provides summarized findings and results achieved from the two text-line image recognition approaches proposed for Amharic script.

7.1 Attention-based Encoder-Decoder Model for Amharic OCR

Text-image recognition has recently been widely treated as a sequence to sequence learning task while traditional segmentation-based character recognition has been applied for a longer time. Later an LSTM-CTC based techniques have been used for the recognition of multiple scripts including Amharic

script which imparts the valuable spatial and structural information of text-line images. In this section, we aim to push the limits of such techniques using Attention-based text-line image recognition which is totally based on neural networks. Unlike the CTC, Attention-based models explicitly use the history of the target sequence without any conditional independence assumptions. The following section gives a detailed overview and training procedures of the proposed Attention-based encoder-decoder OCR model.

7.1.1 Introduction

The standard OCR tasks have been investigated based on CNNs and RNNs [BHL+19a] by utilizing the CTC [GLB+08] objective function. However, CTC-based architectures are subject to inherent limitations such as conditional independence assumption, strict monotonic input-output alignments, and an output sequence length that is bound by the subsampled input length while the Attention-based sequence-to-sequence model is more flexible, suit the temporal nature of the text and are able to focus on the most relevant features of the input by incorporating Attention mechanisms [BCB14].

In addition, Attention enables the networks to potentially model language structures, rather than simply mapping an input to an output [CCB15]. Moreover, its success in the area of NLP tasks such as neural machine translation [BCB14, LPM15] and speech recognition [DLY+19, WHK+17] motivates us to investigate such models in the context of Amharic text-image recognition.

Unlike the previous sections that use the LSTM-CTC based networks, the method proposed in this section uses the concept of Attention mechanism with the following major contributions:

- We propose an Attention-based OCR framework for Amharic text-image recognition for the first time.
- The proposed method is trained by injection learning strategy which allows the model to learn from its own error and reduce exposure of bias, unlike the existing Attention mechanisms that are usually trained by teacher forcing techniques.
- Different from the existing Attention-based encoder-decoder model that uses the last hidden state of the encoder as an initial hidden state of the decoder, the proposed model uses independent and randomly initialized hidden states for both encoder and decoder layers.

- The proposed model is designed by leveraging the architecture of the Seq2Seq framework, used in NMT, and then stacking CNN layers before the encoder network as a feature extractor. During training, the overall model components are treated as a unified framework.

7.1.2 Related Work

The exiting OCR models can utilize either traditional or holistic techniques. Methods belong to the first category were mainly applied before the introduction of deep learning, and follows step-wise routines. In contrast, the holistic approach integrated the feature extraction and sequential translation steps in a unified framework that are trained from end-to-end. The details are given in chapter 2. Therefore, in this section, we only review the existing state-of-the-art techniques that are applied for sequence-to-sequence learning tasks.

The main challenge in sequence-to-sequence learning tasks is to find an appropriate alignment between input and output sequences of variable length. For text-image recognition, one needs to identify the correct character at each time step without any prior knowledge about the alignment between the input image pixels and the target characters. The current two major methods that overcome this problem are CTC- based and Attention-based sequence-to-sequence approaches. CTC-based models compute a probability distribution over all possible output sequences, given an input sequence. They do so by dividing the input sequence into frames and emitting, for each frame, the likelihood of each character of the target alphabet. The probability distribution can be used to infer the actual output greedily, either by taking the most likely character at each time step.

In literature, most CTC-based architectures for text-image recognition were LSTM networks, in many cases using the Bi-LSTM [BHL+19a]. Others, like [LNNN17, BHM+20], integrate CNNs for an improved low-level feature extraction prior to the recurrent layers. This approach is applied to printed text-image recognition [GVK19], handwritten recognition [YBJL12], and license plate recognition [STA+18].

Recently, an alternative idea of sequence-to-sequence architectures that follow the encoder-decoder framework, is to decouple the decoding from the feature extraction. The models consist of an encoder module, at the bottom layer, that reads and builds a feature representation of the input sequence, and a decoder module, at the top layer, which generates the output sequence

one token at a time. The decoder uses the Attention mechanism to gather context information and search for relevant parts of the encoded features. Bahdanua [BCB14] and Luong [LPM15] proposed Attention-based encoder-decoder model for machine language translation. Other works, like Doetsch et al. [DZN16], apply an Attention neural network to the the output of a BLSTM network operating on frames extracted from a text line image with a sliding window and Bluche [Blu16] propose a similar technique for end-to-end handwritten paragraph recognition.

Finally, our system is based on Bahdanua [BCB14] Attention network architecture the one proposed for neural machine translation. The main difference is that we apply the Attention for text-line image recognition where the decoder network outputs character by character given the decoder history and the expected input from the Attention mechanism. Further, CNN layers are integrated with the encoder network as feature descriptors and unlike Bahdanau's networks that use GRU, both the encoder and decoder networks are LSTM networks. In addition, the hidden state of decoder LSTM is randomly initialized with a Keras default weight initializer, *Xavier uniform initializer*, instead of the final state of the encoder LSTM network. In the case of Bahdanua's Attention training was done using the teacher forcing technique while our proposed model is trained by re-injecting the previous decoder's predictions into the current decoder's input.

7.1.3 The Proposed Attention-based Network

In this section, we elaborate on the proposed Attention-based encoder-decoder network for Amharic OCR. In the text-image recognition task, the raw data is 32 by 128 text-line images which should be processed and encoded to a sequence of high-level features. To do so, as illustrated in Figure 7.1, our Amharic OCR model follows the standard encoder-decoder framework with the Attention mechanism. The model consists of three basic modules: an encoder that combines a CNN as a generic feature extractor with recurrent layers to introduce temporal contexts in the feature representation, a decoder that utilizes a recurrent layer to interpret those features, and an Attention mechanism that enables the decoder to focus on the most relevant encoded features at each decoding time step.

Since our encoder starts with CNNs, the segmented text-line images are converted into a sequence of visual feature vectors. CNN layers integrated in encoder network are pretrained model weights adopted from an Amharic

text-image recognizer proposed by Belay [BHM+20] (see section 6.2). Then, Bidirectional-LSTM layers, which read the sequence of convolutional features to encode temporal context between them, are employed. The Bidirectional-LSTM processes the sequence in opposite directions to encode both forward and backward dependencies and capture the natural relationship of texts.

Suppose that the input text-line image I consists of a sequence length Tx, then the encoder-CNN processes an input image I and transfers it into an intermediate-level feature map X, which can be thought of as a sequence of column vectors $X = (x_1, x_2, ..., x_{Tx})$. This sequence is then processed by two Bidirectional-LSTM layers and we get the combined state sequence of the final encoded feature map $H = (h_1, h_2, ..., h_{Tx})$, of the forward and backward hidden states. The third component of this model is a unidirectional LSTM layer, called the decoder that generates the target character sequence present in the image given the current image summary and state vector. At each time-step t, the decoder computes a probability distribution over the possible characters and predicts the most probable character y_t, conditioned on its own previous predictions $(y_0, y_1, ..., y_{t-l})$ and a time-dependent context vector c_t, which contains information from the encoded features. Formally, for an output sequence length Ty, it defines a probability over the output sequence $Y = (y_1, y_2, ..., y_{Ty})$ by modeling each condition as $p(y_t|(y_0, y_1, ..., y_{T_{y-1}}, c_t)) = softmax(f(y_{t-1}, s_{t-1}, c_t))$ where f and s_{t-1} represents the current and previous LSTM hidden states respectively.

At each step of decoding, Attention layer is introduced to focus on the most relevant part of the encoded feature representation. For each particular input sequence, attention mechanism has the power to modify the context vector at each time step based on some similarity of the decoder hidden state s_{j-1} with the encoded features h of the context vector c_j where c_j is computed for each target character y_j as given by Equation (7.1)

$$c_j = \sum_{i=1}^{Tx} w_{j,i} h_i \tag{7.1}$$

where the $w_{j,i}$ is an attention weight which is computed by soft-maxing its corresponding alignment score $a_{j,i}$ using Equation (7.2),

$$w_{j,i} = \frac{exp(a_{j,i})}{\sum_{k=1}^{Tx} exp(a_{jk})} \tag{7.2}$$

where $a_{j,i}$ is the alignment score of concatenated annotation between s_{j-1} and

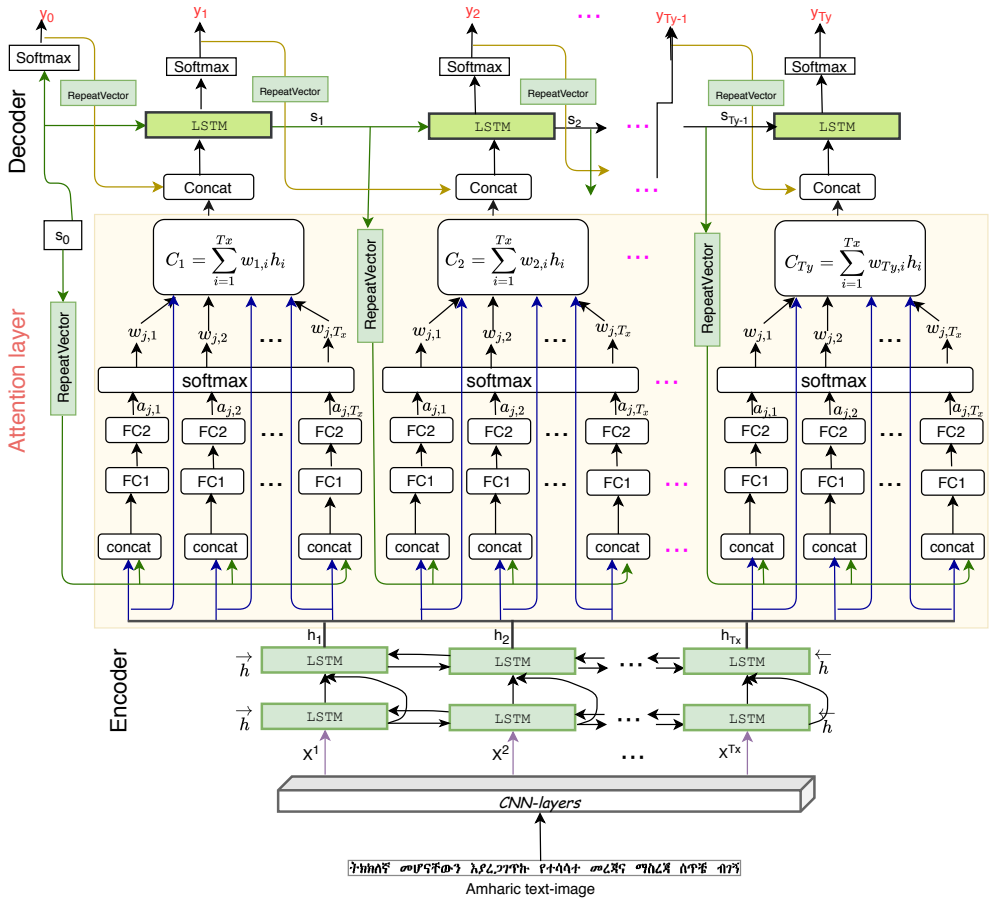


Figure 7.1: **Attention-based Encoder-Decoder model for Amharic OCR.** *The encoder converts an input text-line image I into a sequence of constant feature vectors h. The decoder generates the output sequence y =$(y_0, y_1, ..., y_{Ty})$ one character at a time, where y_0 is the dummy sequence which is generated before the actual target character sequence started to generate. At each time step t, it uses an attention mechanism to produce a context vector c_t based on the encoded feature vectors and a time-dependent decoder hidden state s_t. This context vector is used to generate the decoders output y_t for the current time step. A concatenation of the context vector at t time step and the output y at t-1 time step serves as the decoders next input. During concatenation, to have the same dimension, feature vectors are regenerated using, RepeatVector, a keras function.*

h_i at each time step t and it can be computed using Equation (7.3).

$$a_{j,i} = f(g(h_i, s_{j-1})), \text{for } i = 1, ..., T_x \tag{7.3}$$

The function g and f in Equation (7.3) are feed-forward neural networks, with tanh activation function, that are stacked consecutively. The intuition of f and g is to let the model learn the alignment weights together with the translation while training the whole model layers.

7.1.4 Experimental Results

We carried out the experiments on the ADOCR Amharic database which is described in detail in chapter 4. The encoder and decoder of the RNNs have 128 hidden units each. The encoder of the RNN consists of forward and backward RNNs each having 128 hidden units. Its decoder also has 128 hidden units. In both cases, we use a multilayer perceptron network, two in the encoder part and one in the decoder part, with a single softmax hidden layer to compute the conditional probability of each target character. The whole architecture, depicted in Figure 7.1, computes a fully differentiable function, which parameters can be trained from end-to-end in a supervised manner via the backpropagation algorithm. The optimized cost is the negative log-likelihood of the correct transcription and given as,

$$Loss_{(I,y)} = -\sum_{t}^{L} log(y_t|I) \tag{7.4}$$

where I is the image, $y = y_1, y_2, ..., y_{Ty}$ is the target character sequence and $p(y_t|I)$ are the output probability of the network.

LSTM networks are employed for both the decoder and encoder modules. The Attention mechanism, in our model, computes the attention weight and outputs the context vector once it took all hidden states of the encoder LSTM and the previous hidden state of the decoder-LSTM. In the entire model, the Attention module is iteratively called, till it reaches the maximum target sequence length, Ty times to give back the computed context vector which is going to be used by the decoder-LSTM. At this time all y_t copies have the same weights by implementing layers with shareable weights. The networks are trained, to optimize an entropy-based loss, with RMSProp optimizer and a batch size of 128 for 25 epochs.

The performance of the model is measured using the CER and 1.17% and

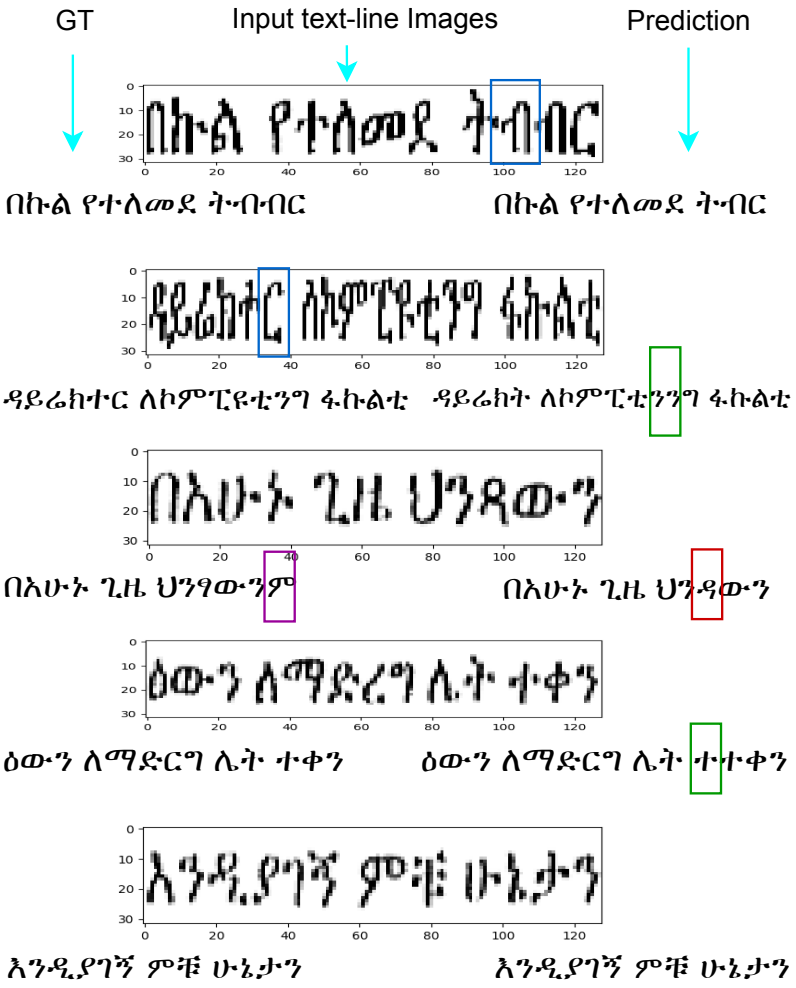


Figure 7.2: **Sample predicted text images using attention-based OCR model.** *Text-line images (middle), their corresponding GT texts (left) and model predictions (right).*

5.21% on ADOCR test datasets that are synthetically generated with *Visual Geez* and *Power Geez* fonts respectively. We also carried out other experiments on a printed Amharic text-line images dataset from the ADOCR test dataset and achieved a promising result with a CER of 2.54%. Sample text-line images that are wrongly recognized during evaluation of our Amharic OCR model are depicted in Figure 7.2. Characters marked by colored-boxes are wrong predictions (it can be deletion, substitution or insertion errors). For example, characters marked by blue-boxes, such as ብ and ር from the first and second text-line image respectively are sample deleted characters. Other characters, such as ን and ተ, from second and fourth predicted texts that are marked with green-boxes are insertion errors, while character ዳ in the

third predicted text which is marked by red-box is one of the substitution error recorded during experimentation.

The other type of error that usually affects the recognition performance of our model is the missing of characters either on the ground-truth or on the text-line image itself. For example, the character ፇ, marked by violet-box, in the third text-line image's ground-truth is one of the character missed during the text-line image generation. Even though all visible characters in the text-line image are predicted correctly, since the CER is computed between the ground-truth texts and the predicted texts, such types of errors are still considered deletion errors. Besides wrongly predicted text-line images, the fifth text-line image in Figure 7.2, is a sample correctly predicted text-line image from ADOCR test sets.

7.1.5 Conclusions

This section successfully approaches the Amharic script recognition with an Attention-based sequence-to-sequence architecture. The proposed model integrates convolutional feature extractors with recurrent neural networks to encode both the visual information, as well as the temporal context in the input image while separate recurrent neural networks are employed to decode the actual Amharic character sequence. Overall, we obtain results that are competitive with the state-of-the-art recognition performance on ADOCR datasets. As we observed the empirical results, the Attention-based encoder-decoder model becomes poor when the sequence length increases. In most cases, the first characters are always correctly predicted while the rest errors have no patterns; thus it is hard to learn in the initial training stage for longer input sequences. Such character errors are not observed in the LSTM-CTC based networks. The next section presents an OCR model which is designed by blending the concept of attention into the CTC objective function.

7.2 Blended Attention-CTC for Amharic Text-image Recognition

In chapter 6 of this thesis, CTC-based networks have been proposed for recognition of Amharic text-line image (see section 6.1 and 6.2). In addition, in section 7.1 of this chapter, Attention-based encoder-decoder networks have been

also introduced for Amharic OCR. In all attempts, the recognition performance of the proposed approaches is evaluated with the ADOCR test database and promising experimental results are reported. In this section, we present a novel Attention-based approach which is designed by blending the concept of Attention mechanism directly within the CTC objective function. The proposed model consists of an encoder module, Attention module, and transcription module in a unified framework. The new approach and implementation details are presented as follows. Section 7.2.1 talks about related works. The proposed blended Attention-CTC model is presented in section 7.2.2, and section 7.2.3 presents the detail of datasets. In the last two subsections, experimental results and conclusions are presented respectively.

7.2.1 Related work

Even though previous research on Amharic OCR was focused on segmentation-based OCR models [MJ07, BHL+19b, CH03, BHS18], recently published works [ALT18] and [BHL+19a] proposed a Bidirectional LSTM network architecture with CTC for Amharic text-line image recognition. End-to-end learning, that uses CNN, LSTM, and CTC in a unified framework [BHM+20], is also proposed for Amharic OCR and achieved a better recognition performance. These segmentation-free attempts made for Amharic script recognition push the research for the OCR of Amharic a step forward. Several segmentation-free OCR techniques, based on Attention and CTC networks, have been studied and demonstrated groundbreaking performances for multiple scrips such as Convolutional Recurrent Neural Network (CRNN) for Japanese handwritten recognition [LNNN17], segmentation free Chinese handwritten text recognition [ML15], a hybrid Convolutional-LSTM for text image recognition [Bre17], Multidimensional LSTM for Chinese handwritten recognition [WYCL17], combined Connectionist Temporal Classification (CTC) with Bidirectional LSTM for unconstrained online handwriting recognition [GLB+08].

As described in section 7.1, researchers have recently applied the Attention mechanism in different research areas including in the field of OCR. Therefore, it becomes popular and a choice of many researchers in the area of OCR. Attention mechanism has been applied for recognizing handwritten texts [PV17, CV18], characters in the wild [LO16], handwritten mathematical expression [ZDD18]. The attention mechanism assists the network in learning the correct alignment between the input image pixels and the target

characters. In addition, it improves the ability of the network in extracting the most relevant feature for each part of the output sequence. Inspired by the success of the Attention mechanism in sequence-to-sequence translation, we continue to focus on OCR tasks, and we integrate the capability of the Attention mechanism in the CTC network so as utilize the benefits from both techniques.

7.2.2 The Proposed Blended Attention-CTC Model

The proposed blended Attention-CTC model, as shown in Figure 7.3, consists of three modules. The encoder module takes the input features x and maps them to a higher-level feature representation $h^{enc} = (h_1{}^{enc}, h_2{}^{enc}, ..., h_{\text{Tx}}{}^{enc})$. The Attention module takes the output features h^{enc} of the encoder module and computes the context vector from each hidden features. The output of the Attention module, attention context information, is passed to the Soft-max layer in order to produce a probability distribution, $P(y_0, y_1, ..., y_{\text{t-1}}|x)$ over the given input sequence x. Finally, the probability distribution P goes to the CTC-decoder for transcription.

The proposed model doesn't change the usual training procedure of CTC explained in [GLB+08]. However, we embed the Attention mechanism, hereby further described as follows, in between the LSTM and CTC layers. The intuition of the attention layer, in our proposed model, is to produce a weighted context vector (C_{vec}) with an improved hidden layer representation, and this context vector is computed using Equation (7.5).

$$C^t_{\text{vec}} = \sum_{i=1}^{Tx} s_{\text{ti}} h_{\text{i}}{}^{enc} \tag{7.5}$$

where the s_t is an attention weight of each annotation $h_{\text{i}}{}^{enc}$ computed by soft-maxing its corresponding attention score using Equation (7.6),

$$s_{\text{i}} = \frac{exp(a_{\text{i}})}{\sum_{k=1}^{Tx} exp(a_{\text{k}})} \tag{7.6}$$

where a_{i} is the attention score of h_i^{enc} at each time step t and it can be computed using Equation (7.7).

$$a_{\text{i}} = f(h_{\text{i}}{}^{enc}), \text{for } i = 1, ..., T_x \tag{7.7}$$

The function f in Equation (7.7) is a feed-forward neural network scoring

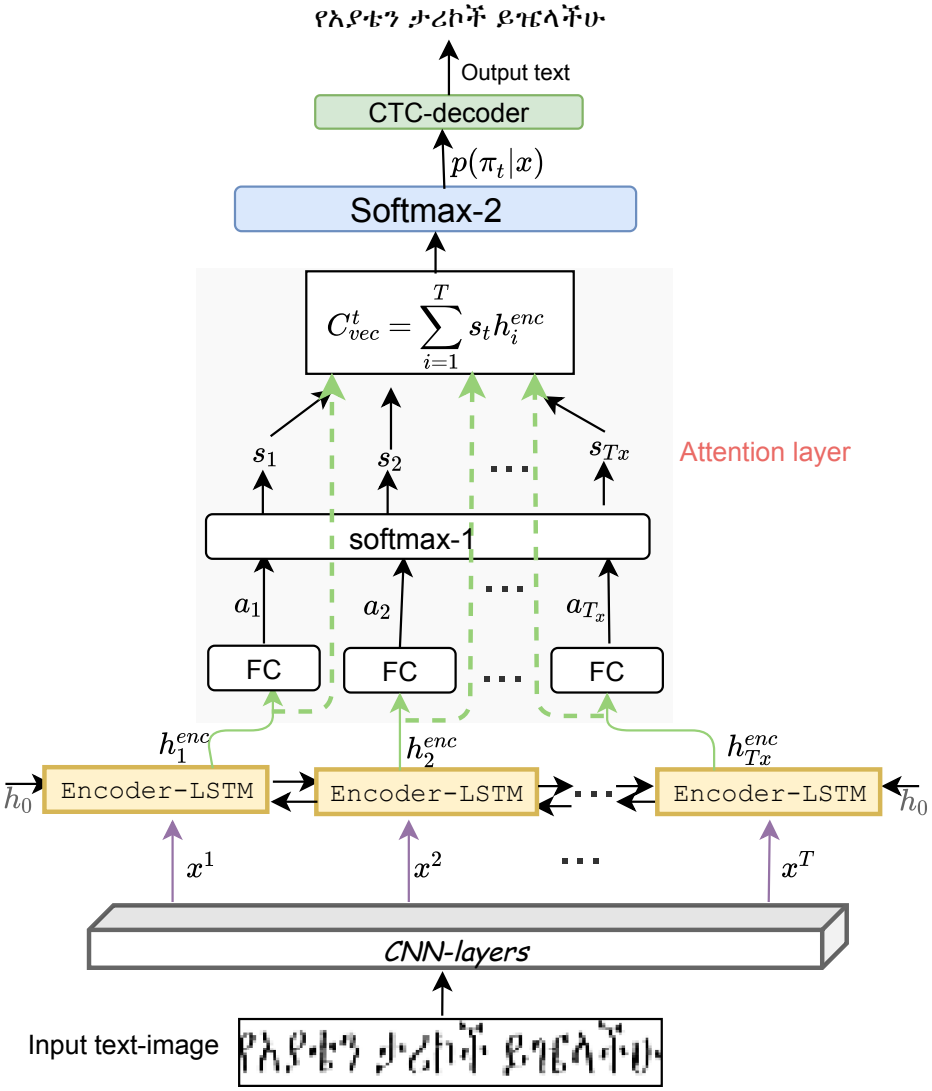


Figure 7.3: **The proposed blended Attention-CTC model.** *A score* (a_i) *of encoder hidden state* (h^{enc}_i) *at each time-step is computed using a scoring function described in Equation* (7.7)*, just by propagating the* h^{enc}_i *through fully connected network (FC). Attention distribution, called attention weights* (s_i) *are computed by running all scores over a soft-max (Soft-max-1) layer. To compute the alignment vector , we multiply each* h^{enc}_i *with its corresponding soft-maxed score* (s_i)*. Then, the sum of all alignment vectors produced the context vector* (C_{vec})*, which is an aggregated information. Once the* C_{vec} *is obtained, it passes through the second soft-max layer (Soft-max-2) for probability distribution over the n possible characters in the ground-truth (GT). The output of Soft-max-2 is a sequence of T time steps of* $(n+1)$ *characters which is then decoded using CTC-decoder.*

function that lets the model learn the alignment weights together with the translation while training the whole model layers.

During training the blended Attention-CTC model, a CTC loss function (l_{CTC}) is used to train the network from end-to-end (See Equation (2.5)) and a target label in path π is obtained by mapping reduction function B, using the example explained in [BHL+19a], that convert a sequence of Soft-max output for each frame to a label sequence by removing repeated labels and blank tokens from the sequences of character (C) with the highest score (i) generated using Equation (6.1).

7.2.3 Database

To train and evaluate the performance of the proposed model, we use the ADOCR database introduced by [BHL+19a]. The original Amharic OCR database was composed of 337,337 Amharic text-line images, each with multiple word instances collected from different sources. Sample Amharic text-line images from ADOCR database and the detail descriptions are given in chapter 4, section 4.3.2.

7.2.4 Experiments

Our model is trained with the ADOCR database. Similar to section 6.2, images are scaled to 32 by 128 pixels so as to minimize computations. Since there is no explicitly stated validation data, in the ADOCR dataset, the same selection mechanism is applied and randomly selected 7% of the training samples as validation samples. A blended Attention-CTC network is formulated by directly taking the advantage of Attention mechanism and CTC network as an integrated framework and trained in an end-to-end fashion.

During training this model, we use two bi-directional LSTM, each with 128 hidden units and a dropout rate of 0.25, on top of seven convolutional layers that are stacked serially with ReLU activation function as an encoder. The decoder LSTM, presented in section 7.1, is removed and then the CTC objective function that is blended with the Attention mechanism is in place. The convolutional layers of the encoder module are composed of seven convolutional layers which are already trained in section 6.2 and the same weights are used in this section via transfer learning.

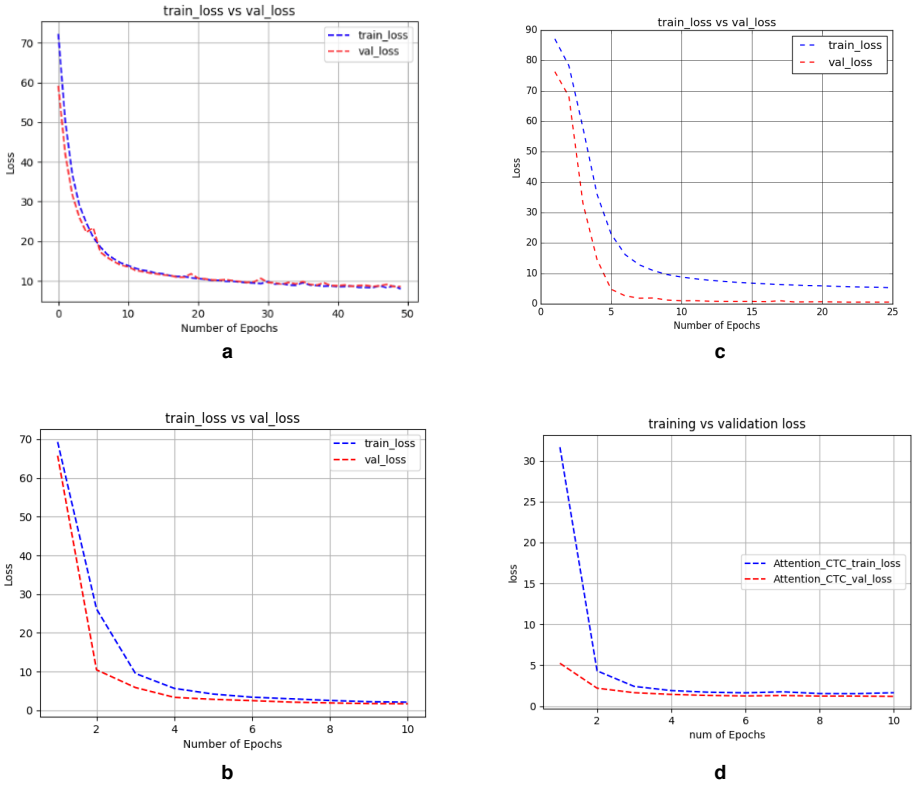


Figure 7.4: **The training & validation losses of model trained with different network settings.** *(**a**). CTC loss with an LSTM-CTC model. (**b**) CTC loss with a CNN-LSTM-CTC model. (**c**) CE loss of Attention-based encoder-decoder model. (**d**) CTC loss of the proposed blended Attention-CTC model*

We use a batch size of 128 with Adam optimizer and trained for 15 epochs. Once the probability of labels is obtained from the trained model, we use best path decoding [Gra08] to generate a character (C_{i}) that has the maximum score at each time step t (See Equation (6.1)). The learning loss of the proposed model and other models trained using different network settings are depicted in Figure 7.4.

7.2.5 Results

The performance of the proposed blended Attention-CTC model is evaluated against three test datasets, from the ADOCR database, and compared with state-of-the-art approaches and the proposed model improves the recognition performance by 0.67–7.50% from the LSTM-CTC based model and by 2% from the recently published work with CNN-LSTM-CTC network which use

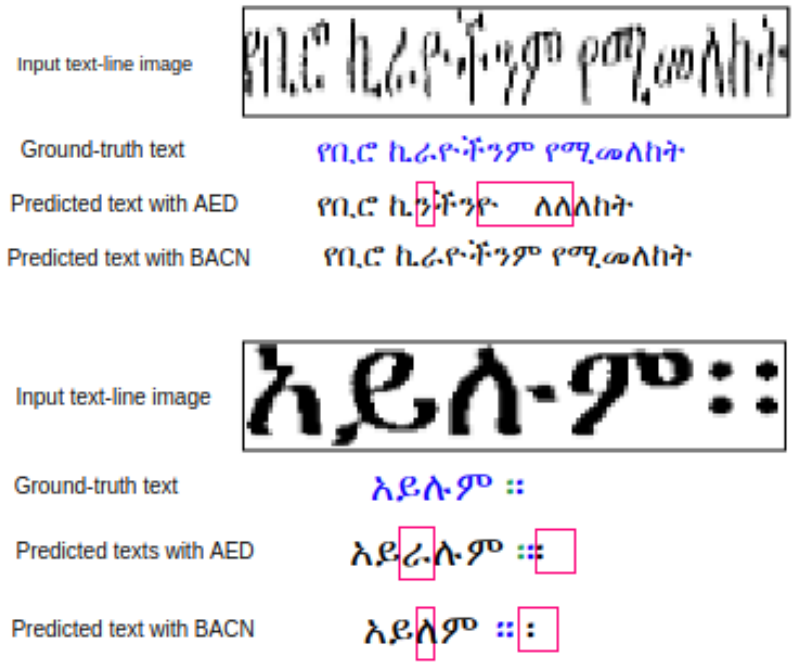


Figure 7.5: **Sample text-line image with the corresponding predicted and ground-truth texts.** *Characters, in the predicted text, marked with red rectangle are wrongly predicted by both Attention based Encoder-decoder (AED) and Blended Attention-CTC Networks (BACN).*

the same test dataset. Sample Amharic text-image from ADOCR test dataset with the corresponding GT texts and predictions are depicted in Figure 7.5.

7.2.6 Conclusions

In this section, a Blended Attention-CTC network called BACN, for Amharic text-line image recognition, is presented. BACN consists of a Bidirectional LSTM, stacked on top of CNN layers, as an encoder and a CTC layer as a decoder. To enhance the hidden layer feature representation, the Attention mechanism is embedded between the LSTM and CTC network layers without changing the CTC objective function and the training process. All the encoder, Attention, and CTC modules are trained jointly from end-to-end and a significant improvement is achieved on all the three ADOCR test datasets compared with state-of-the-art results. Thus, we can conclude that the blended Attention-CTC network is more effective for Amharic text image recognition than widely used attention-based encoder-decoder and CNN-

LSTM-CTC based networks as well.

7.3 Chapter Summary

In the previous chapter of this thesis, CTC-based networks were employed for Amharic text-line image recognition and reported baseline results on test sets of the ADOCR database. This chapter proposed an alternative and effective text-image recognizer based on the attention mechanism. The proposed model consists of CNN integrated with Bidirectional-LSTMs as an encoder and a unidirectional LSTM as a decoder that is unified as a single framework. Then a comparable recognition performance is achieved against the three test datasets from the ADOCR Amharic database. Unlike the CTC-based loss, attention-based networks minimized the categorical cross-entropy loss. In addition, to gain the advantages of both the attention and CTC-based networks, a new method is proposed by embedding the attention mechanism into the CTC network without changing the training process in CTC objective function. In summary, the blended attention-CTC model outperforms all the state-of-the-art models on the ADOCR test datasets.

Chapter 8

Conclusions and Recommendations for Future Work

This chapter presents the conclusions of the research work described in the thesis. The contributions and objectives of the thesis, outlined in chapter 1, are reviewed and their achievements are addressed. Factors that limit the efficiency of the present work are also presented. The following section presents the summary of the most relevant conclusions drawn from this thesis and based on the findings of this thesis work, recommendations are made for future research on Amharic script recognition.

8.1 Conclusions

In this thesis, we addressed the problem of Amharic script recognition and it presents substantial contributions in the field of Optical Character Recognition (OCR) for printed documents by extending the use of contemporary deep learning algorithms in this domain. The nature and complexities of Amharic script have been highlighted, and the potential of the deep learning technique is exploited because of its success in the field of computer vision and pattern recognition.

To enable information access to everyone and break language barriers in the world, there has been an increasing demand for developing workable OCR systems. This demand has been partially fulfilled over time and there are numerous works presented for multiple scripts. However, Amharic script has not been exposed to state-of-the-art OCR research methods. For indigenous

scripts in general, only a few efforts are reported to date, which is not enough to address the complexities. Amharic, whose speakers exceeds 100 million people around the globe, is one of the most underrepresented scripts in the field of NLP. Such resource-limited but unique script presents various challenges towards the development of an efficient OCR system. Some of the challenges are the use of a large number of characters in the writing system and the existence of a large set of visually similar characters. In addition, there are no standards for font type, size, and style which results in the script being more complex for recognition. There are numerous documents, in Amharic script, that is collected for centuries. For further analysis of these documents, digitization is critical. This signifies the importance of the research in this thesis.

In this thesis, an optimal OCR model for Amharic script recognition is constructed using state-of-the-art deep learning techniques. Thus, we explore different approaches for understanding and addressing problems with an emphasis on Amharic script. To this end, extensive works have been made so as to contribute to designing the OCR model for the recognition of Amharic printed document images.

This thesis consists of different conceptual and technical contributions which can be seen from two major perspectives. The first contribution is the creation of an ADOCR database with various versions and types of datasets that have been developed to train and evaluate the performance of OCR systems of Amharic script. This database contains character and text-line images. The character level database is composed of 80,000 synthetically generated basic Amharic characters while the second dataset is composed of 337,337 text-line images. The text-line dataset is created considering the writing system in Amharic script and the texts are collected from different sources. We believe that the ADOCR database could be used as a baseline database for further Amharic OCR system development by researchers in the area.

In the second contribution of this thesis, deep learning-based OCR methodologies have been assessed for Amharic script recognition. In this thesis, we present the major challenges in the development of the OCR model for indigenous scripts, like Amharic, which have their own alphabets. Accordingly, these challenges are due to the nature of the script itself and/or approaches followed by research attempts made, on Amharic script, so far. All previous attempts were based on classical machine learning techniques and the models were trained with segmented characters. Further, the number of datasets was limited in number and they did not include all the possible characters

used in Amharic script. Moreover, none of these datasets are publicly available. Hence, we have explored state-of-the-art OCR tools and techniques ranging from dataset preparation to model development, variants of artificial neural networks ranging from convolutional neural networks to recurrent networks, sequence-to-sequence learning, and alignment mechanisms ranging from CTC to Attention mechanism. Based on the results from empirical observations and conceptual frameworks from the theoretical foundation, several conclusions can be drawn in this thesis work.

- The shortage of training dataset can be solved with artificial data, however, it should be generated carefully to reflect scanned real-world documents as close as possible. Experiments performed, in this thesis, by training different deep-learning-based OCR models on synthetic data and testing them on real scanned Amharic documents justifies this claim. This thesis presents the procedures for synthetic data generation and real scanned data preparation. Further, the dataset used for Amharic script recognition, in this thesis work, is now made publicly available at `http://www.dfki.uni-kl.de/~belay/` for free.

- This thesis presented novel CNN-based Amharic character recognition techniques. The approach employed a Factored-CNN network that is designed based on the grapheme of basic Amharic characters called Fidel-Gebeta and achieved character recognition accuracies that are significantly better than the performance of existing Amharic characters recognition techniques.

- Stacking CNNs, before the LSTM network layers, as a feature extractor and training the hybrid networks using a CTC objective function in an end-to-end fashion improves the recognition performance than the LSTM-CTC-based network model.

- Attention embedded RNNs, from neural machine translation tasks, are adopted and employed for Amharic text-line image recognition. The recognition efficiency reached near to the performance of CTC-based sequence alignments.

- The integration of the Attention mechanism and CTC objective function makes the sequence-to-sequence learning network perform better, instead of employing each of them separately.

Experimental results show the effectiveness of our approach for the recognition of Amharic scripts. The majority of the recognition errors observed are due to improper annotation among text-line images and the ground truth while

other errors are due to the existence of deformed and/or missing characters in the beginning and/or end of the text-line images during data generation from the given texts. In general, the strategies followed, in this thesis, are promising for the recognition of large digitized Amharic document images. Therefore, the present approaches could be extended, redesigned or new approaches need to be proposed for the OCR of historical and handwritten Amharic documents. Tasks that are not addressed in this thesis and the possibilities to extend the current work are provided in the next section.

8.2 Recommendations for Future Work

To advance the OCR of Amharic script, in this thesis, various deep learning-based schemes are proposed. Based on the findings and progression of the current work, we recommend the following research directions as a possible future work so as to design an efficient OCR system and digitization of Amharic language in general.

- In the present work an attempt is made to develop the OCR model and introduce a database for recognition of Amharic script. It pushes the technology of Amharic script one step towards digitization. To reduce the barriers of this language in the digital world, we need appropriate data that can be used to analyze the linguistics of Amharic. Neural machine translation and speech recognition need notable attention in the future since they have not yet been sufficiently investigated.
- The database introduced in this thesis considers only the two commonly used fonts of Amharic script. Even though a large Amharic database is created with those fonts, there are other fonts rarely used in Amharic writing system that is not included in the training set, which may affect the recognition performance of the OCR model. To facilitate OCR researches and develop versatile OCR for Amharic script, we need to develop a large database of Amharic texts with different font types and styles.
- The OCR models proposed in this thesis are based on various state-of-the-art deep learning-based techniques. The recognition performance of these models is evaluated using a test dataset from the ADOCR database. In this regard, we have achieved promising results. Compared to the over centuries Amharic document collection, which contains images, tables, unknown fonts, and non-standard scripting, the ADOCR

database is not complex enough. We believe that the current model should fit for recognition of data with more complex layouts and it could also be explored for scene Amharic text-image recognition hence it is recommended as a future research direction.

- Training and evaluation of the OCR methodology, in this thesis, are based on synthetic and scanned Amharic documents. However, the same approach can be extended for the case of documents captured with other document digitizing technology like a camera. Digitizing documents with the camera may introduce several issues hence as future work we need to integrate some document preprocessing techniques so as to correct this issue and simplify the task of recognition with the current model.
- In this research an OCR system is developed for synthetically generated and printed Amharic documents. Therefore, it is important to extend the current research work for the recognition of handwritten and historical Amharic documents written on parchments and vellums.

Finally, we believe that the approaches followed in this thesis successfully fulfill the gaps that were presented in Amharic script recognition. We hope it would also serve as a stepping stone for the future researcher who will work in the OCR domain specifically for such indigenous and low-resource scripts.

Appendices

Appendix A

A.1 Attention Mechanism

This part of the thesis is an appendix that presents the details of the Attention mechanism and internal working of Attention techniques. Various architectures of encoder-decoder networks that have been used commonly for sequence-to-sequence tasks, including NMT and OCR are also described in detail. Besides, it gives the necessary mathematical details of Attention-based networks and basic parameters of attention mechanism that one needs to tune when training the attention-based encoder-decoder networks.

The concept of Attention in the neural process has been largely studied in Neuroscience and Computational Neuroscience first [MZ17, IKN98] and consequently, there were many attempts to replicate it in Machine Learning [SVL14, LPM15]. In such fields, a particular aspect of the study is visual attention; thus many animals focus on specific parts of their visual inputs to compute adequate responses. This principle has a large impact on neural computation as we need to select the most pertinent piece of information, rather than using all available information, a large part of it is irrelevant to compute the neural response. A neural network is considered to be an effort to mimic human brain actions in a simplified manner. Attention Mechanism is also an attempt to implement the same action of selectively concentrating on a few relevant things while ignoring others in deep neural networks.

A similar idea, focusing on specific parts of the input, has been followed in deep learning, for speech recognition, neural machine translation, visual question and answering, and text-image recognition. In the recent trends of deep learning Attention mechanisms are one of the most exciting advancements, and that they are here to stay.

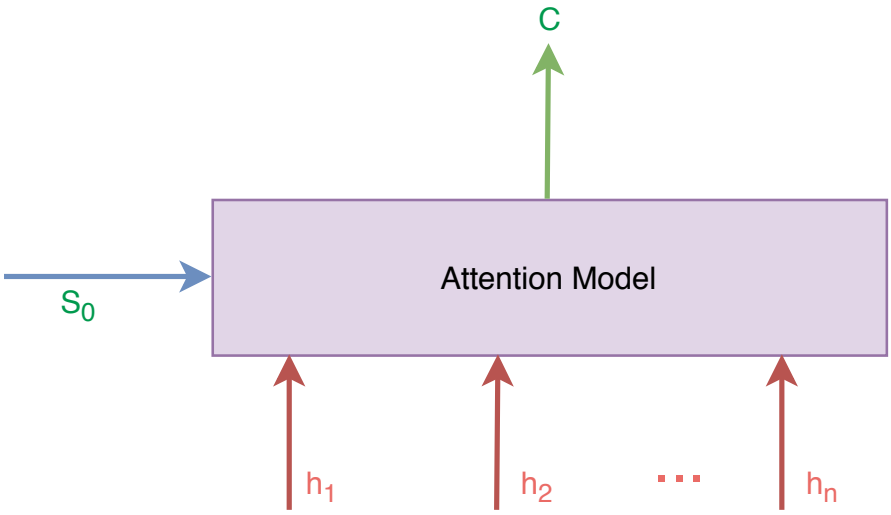


Figure A.1: **General setting of Attention model.** *The initial state s_0 is the beginning of the generated sequence, the h_i are the representations of the parts of the input sequence, and the output is a representation of the filtered sequence, with a filter putting the focus of the interesting part for the sequence currently generated.*

A.1.1 Generic Model of Attention Mechanism

As illustrated in Figure A.1 the generic setting of Attention model takes a sequence with a length of n arguments $h_1, ..., h_n$ (in the preceding examples, the h_i would be the hidden states of the encoder network), and a state s_0. It return a vector C which is supposed to be the summary of the h_i, focusing on information linked to the state s_0. More formally, it returns weighted arithmetic mean of the h_i, and the weights are chosen according to the relevance of each h_i given the state s_0.

The Attention model depicted in Figure A.1 looks a black-box and it doesn't clearly show how the summarized information C is computed; thus the whole structure of the Attention model need to be discussed in detail and the model should be redrawn by expanding the black-box design.

The intuition of mathematical computation, in the Attention model, is to produce a vector called context vector c; thus the network computes $f_1, f_2, ..., f_n$ with a tanh activation layer. f_i is the aggregated value of h_i and s_0, where each f_i is computed, independently, without looking at the other h_j for j is

not equal to i and it is given as,

$$f_i = (W_{s_0 f} s_0 + W_{h s_0} h_i) \tag{A.1}$$

where W is weight matrices to be learned, later, while training the whole model layers. Then each weight metrics of f_i is computed using Softmax function which is also given as,

$$softmax(f_1, ..., f_n) = \left(\frac{exp(f_i))}{\sum_{k=1}^{n} exp(f_k)} \right) i, \text{for } i = 1, ..., n \tag{A.2}$$

where the s_i are the softmax of the f_i projected on a learned direction and $\sum_{i=1}^{n} s_i = 1$. The output c is the weighted arithmetic mean of all the h_i, where the weight represents the relevance for each variable according to the initial hidden state s_0 and computed as,

$$c = \sum_{i=1}^{n} h_i s_i \tag{A.3}$$

A.1.2 Types of Attention and Seq2seq Model

The motivation behind the Attention mechanism is the problem of forgetting long source sequences during translation. Seq2seq model aims to transform an input sequence to a target one and both sequences can be of arbitrary lengths [SVL14]. Examples of transformation tasks include machine translation between multiple languages in either text or audio, question-answer dialog generation, or even image-to-text conversion in OCR. The seq2seq model normally has an encoder-decoder architecture, composed of two processes.

- **Encoding processes**: It takes a variable-length input sequence and compresses the information into a context vector of a fixed length. This representation is expected to be a good summary of the meaning of the whole input sequences.
- **Decoding process**: It is initialized with the context vector to emit the transformed output sequences. In the seq2seq model, the encoder and the decoder networks are linked with the hidden state information which is sent from the last hidden state of the encoder network to the decoder as an initial hidden state of the decoder network.

In seq2seq architecture both the encoder and decoder are recurrent neural networks, which are usually either LSTM or GRU units [CVMG+14,

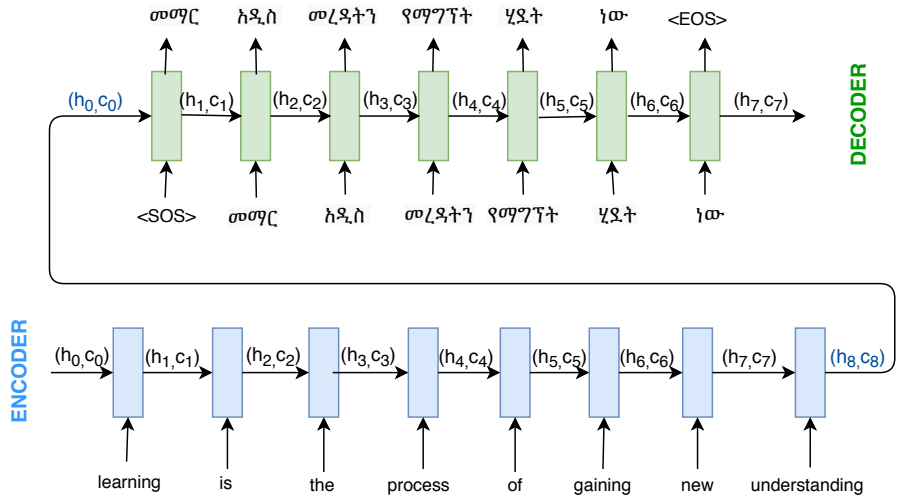


Figure A.2: **Seq2seq model.** *This encoder-decoder model, translating the sentence "Learning is the process of gaining new understanding" to its equivalent Amharic sentence*

BCB14]. An example of seq2seq model illustrated in Figure A.2, where the encoder reads a sequence input with variable lengths, e.g., English words (**Learning is the process of gaining new understanding**), and the decoder produces a sequence output, e.g., corresponding Amharic words, (**መማር አዲስ መረዳትን የሚገኘት ሂደት ነዉ**). As shown in Figure A.2, the encoder network (an LSTM) takes a sequence of an English word as input and encapsulates the information as the internal state vectors which are later used by the decoder. Since LSTM takes only one element at a time, so if the input sequence is of length T, then LSTM takes T time steps to read the entire sequence (i.e in this example the English sentence has 8 words, then the LSTM will read this sequence word by word in 8 time-steps).

The hidden state and cell state (h_0 and c_0) of the encoder are initialized randomly with a smaller value or it can be zero. The dimensions of both states should be the same as the number of units in the LSTM cell. The encoder will read the English sentence word by word and store the final internal states, an intermediate vector, of the LSTM generated after the last time step and since the output will be generated once the entire sequence is read, therefore outputs of the Encoder at each time step have no use and so they are discarded. The final states of the encode (i.e in this case h_8 and c_8) are the relevant information of the whole English sentence "**Learning is the process of gaining new understanding**".

The decoder works in a very similar fashion as the encoder. However, dur-

ing the training and testing phase, the decoder behaves differently unlike the encoder which is the same in both cases. The start and end of the sequence, for the decoder, are marked with $< SOS >$ and $< EOS >$ respectively. During training the initial states (h_0 and c_0) of the decoder is set to the final states of the encoder (h_8 and c_8). The decoder is trained to generate the output based on the information gathered by the encoder by taking $< SOS >$ as the first input to generate the first Amharic word and at last, the decoder learns to predict the, $< EOS >$. The input at each time step is actual output which is usually called the teacher forcing technique. However, the loss is calculated on the predicted outputs from each time step and the errors are backpropagated through time to update the parameters of the model.

Unlike the training phase which uses the actual output as input at each time step, the test phase uses the protected output produced at each time step as input in the next time step. The final states of the decoder are discarded as we got the output hence it is of no use in both the training and testing stages.

A critical and apparent disadvantage of a fixed-length context vector, created in the encoding process of the seq2seq model, is the incapability of remembering long sentences. Often it has forgotten the first part once it completes processing the whole input; thus Attention mechanism was introduced to solve this issue. Even though the first model of Attention mechanisms is introduced by Bahdanau [BCB14], nowadays, there are various types of Attention mechanisms proposed by many researchers from the deep learning community. The underlying principles of Attention are the same in these all types of Attention mechanisms that were proposed so far, the differences lie mainly in their architectures and computations.

Attention can be general and self-attention, hard and soft-attention, global and local-attention. General-attention manages and quantifying the interdependence between the input and output elements [BCB14], where self-attention and interdependency within the input elements and relating different positions of a single sequence to compute a representation of the same sequence [CDL16].

The type of Attention that attends to the entire input state space and uses all the encoder hidden states is also known as global-attention. In contrast, local-attention attends the part of the input sequence or patch of an image and uses only a subset of the encoder hidden states [LPM15, XBK$^+$15]. Considering the neural machine translation tasks, the global-attention has a drawback that it has to attend to all words on the source side for each target word, which is expensive and can potentially render it impractical to trans-

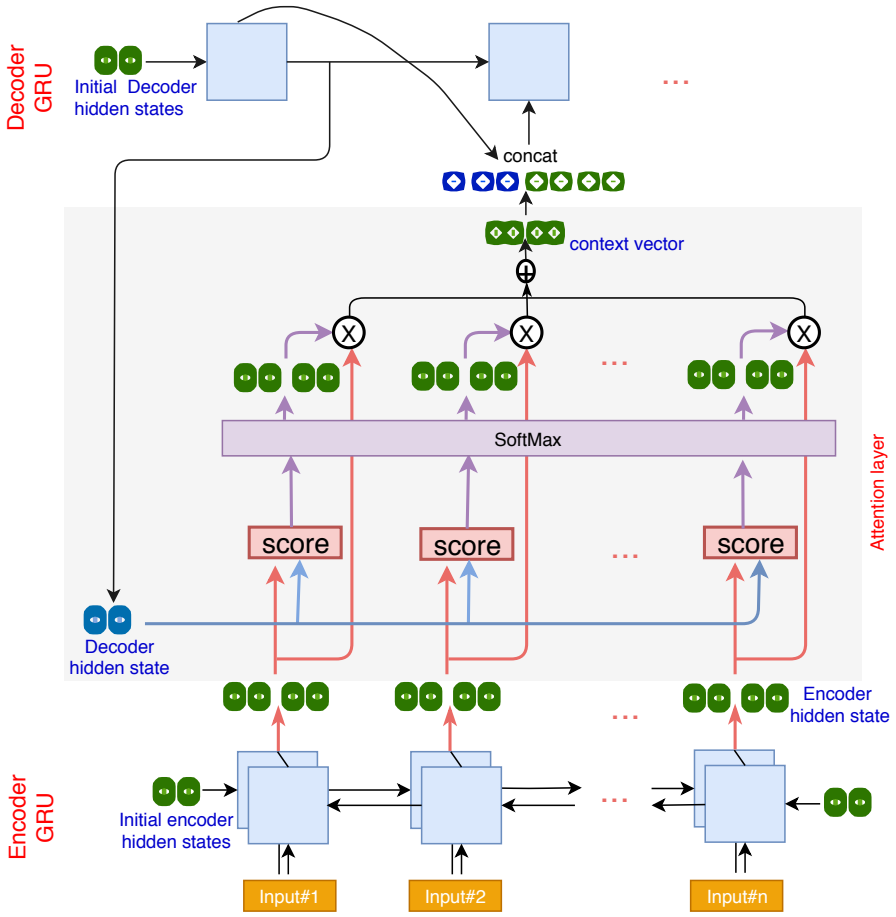


Figure A.3: **A typical architecture of Soft attention .**

late longer sequences such as paragraphs or documents; thus, to address this deficiency, a local-attentional mechanism is proposed that can choose to focus only on a small subset of the source positions per target word. Besides, soft-attention refers to the global-attention [LPM15] approach in which weights are placed softly and attends overall patches in the input sequence. The hard-attention [XBK+15], on the other hand, selects one patch of the image/part of the input sequence to attend at a time. While less expensive at inference time, the hard-attention model is non-differentiable and requires more complicated techniques such as variance reduction or reinforcement learning to train [XBK+15]. In this case, the local-attention mechanism proposed by Luong [LPM15] is selectively focused on a small window of context and is differentiable and it avoids the expensive computation incurred in the soft-attention and at the same time, is easier to train than the hard-attention approach proposed by Xu [XBK+15].

Further, the Attention mechanism proposed by Bahdanau [BCB14] is also a soft-attention and differentiable as well. However, it is expensive in longer sequence inputs. Global-attention is somehow similar to soft-attention while the local one is a blend between hard and soft-attention of Xu[XBK$^+$15] and improvement over the hard-attention to make it differentiable. Therefore, Attention mechanisms generally can be classified as Global/soft as presented in the work of Xu [XBK$^+$15] and Bahdhnau [BCB14] or Local/Hard attention as proposed by Xu [XBK$^+$15] and Luong [LPM15]. The basic architectural differences and working procedures of soft and hard attention, based on the work of Bahidanu and Luong are illustrated in Figure A.3 and A.4 respectively.

In soft attention, [BCB14], the encoder is a bidirectional gated recurrent unit (BiGRU) while the decoder is a GRU whose initial hidden state is a vector modified from the last hidden state of the backward encoder GRU. The general steps followed in this Attention mechanism can be presented as follows:

1. **Initialize hidden states of the encoder network**: The initial encoder hidden states are a random small number or zero and then the encoder initialize hidden states of each element in the input sequence.

2. **Initialize the decoder hidden state**: The initial hidden state of the decoder network is a modified vector of the last hidden state of backward encoder GRU or we can also initialize the decoder hidden states randomly as we did in the encoder network.

3. **Compute alignment scores**: Alignment score is calculated between the previous decoder's hidden state at $t-1$ and each of the encoder's hidden states at time-steps t. In this Attention mechanism, the last encoder hidden state can be used as the initial hidden state of the decoder, and the alignment score function used in this Attention model is called additive or concatenation (i.e Since the decoder hidden state and encoder outputs will be passed through their linear layer and have their trainable weights, the current encoder hidden state and the previous decoder hidden states are first added/concatenated and then $tanh$ is applied). Alignment score computation is given as,

$$s_{align} = W_{shared}.tanh(W^{enc} \cdot h^{enc} + W^{dec} \cdot h^{dec}) \qquad \text{(A.4)}$$

4. **Softmaxing the alignment scores**: Each computed alignment score runs through a softmax layer.

5. **Computing the Context Vector**: The encoder hidden states and their respective soft-maxed alignment scores are multiplied and then summed up the results to form the context vector.

6. **Decoding the Output**: the context vector is concatenated with the previous decoder output and fed into the Decoder for that time step along with the previous decoder hidden state to produce a new output. The process from steps 3 to 6 is repeated for each time step of the decoder until completing the specified maximum length of the output.

The second type of Attention is based on the work of Luong [LPM15] and is often referred to as local/ Multiplicative Attention. In this attention mechanism, the encoder is a two-stacked long short-term memory (LSTM) network while the decoder also has the same architecture, whose initial hidden states are the last hidden states of the encoder network. The input sequences are feed to the encoder in reverse order. Some of the major steps, based on the work of Luong [LPM15], followed in this attention are described as follows.

1. **Initialize the encoder hidden states**: As done in the work of Bahdanau, hidden states of the encoder unit are initialized randomly.

2. **Initialize the decoder hidden states**: Once the initial hidden states of the encoder unit are produced then the last hidden states of both encoder LSTM are passed to the decoder unit as initial hidden states of the decoder.

3. **Decoder RNN configuration**: Since it has two LSTM network layers, the previous decoder RNN hidden state and decoder output is passed through the second decoder RNN to generate a new hidden state for that time step.

4. **Compute alignment scores**: Using the new decoder hidden state and the encoder hidden states, alignment scores are computed. In this case, the Luong Attention uses three types of scoring functions that use the encoder outputs and the decoder hidden state produced in the previous step to calculate the alignment scores. The computation for each of the scoring function is give as follows:

    - **Dot**: In this scoring function the alignment score s_{align} is computed by taking the hidden states of the encoder h^{enc} and multiply them with the hidden state of the decoder h^{dec}. it is given as,

$$s_{align} = h^{enc} \cdot h^{dec} \tag{A.5}$$

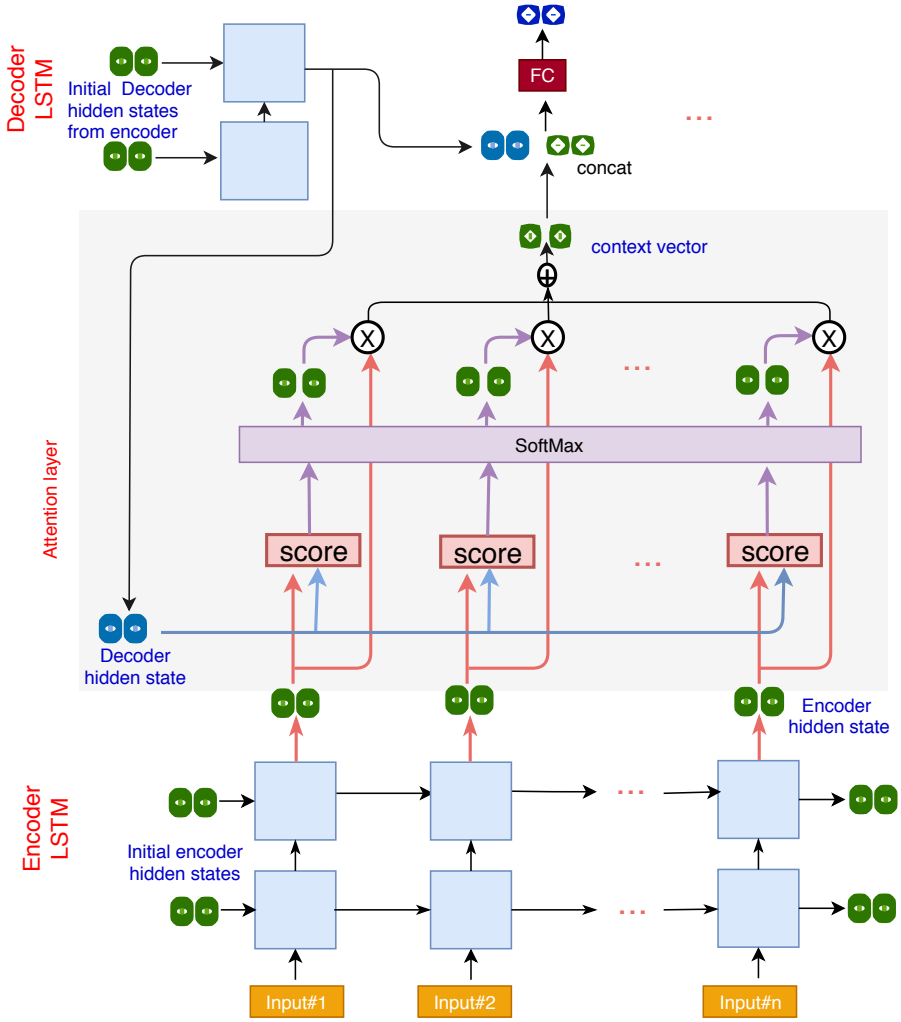


Figure A.4: **Attention model architecture based on the work of Luong.**

- **General**: This scoring function is similar to the dot function, except that a weight matrix is applied on their product and it is give as,

$$s_{align} = W(h^{enc} \cdot h^{dec}) \tag{A.6}$$

- **Concat**: The function **Concat** scoring function of Luong's Attention is slightly similar to Bahdanau's one, whereby the decoder hidden state is added to the encoder hidden states. However, In Luong's Attention the decoder hidden state and encoder hidden states have shared weights since both of them are added together first before being passed through a Linear layer and then this weight matrix will be applied on the $tanh$ activated outputs.

It can be computed as,

$$s_{align} = W.tanh(W_{shared}(h^{enc} + h^{dec})) \tag{A.7}$$

5. **Softmaxing the alignment scores**: the alignment scores for each encoder hidden state are passed through Softmax function.
6. **Computing the context vector**: To compute the context vector, the encoder hidden states and their corresponding alignment scores are multiplied and then summed up.
7. **Producing the output**: The concatenated context vector and the decoder hidden state generated in step 3 are passed through a feed-forward neural network to produce a new final output.

Steps 3 to 8 are repeated themselves till the end of the maximum sequence length.

The Attention mechanism proposed by Luong is different from the one presented by Bahdanau with some features. For example, attention calculation in Bahdanau requires the output of the decoder from the prior time step while Luong Attention uses the output from the encoder and decoder for the current time step only. Besides, Loung uses a reversed input sequence instead of bidirectional inputs, LSTM instead of GRU elements and unlike Bahdanau's Attention, Luong uses the dropout rate.

A.1.3 Basic Parameters of Attention-based Networks

As done in all neural network-based models, there are various network parameters that one needs to tune and use so that the Attention-based networks perform better on a given task. In this case, all parameters of the other module (the encode and decoder modules) should have selected and tuned as usual, however, in this section, in addition to tunable parameters used in this module, the configuration of the Attention module are also described.

Parameters for Encoder and Decoder Networks: The number of hidden layers, the number of LSTM cells, and even the activation function in an individual layer are the important parameters that need to be properly selected and tuned. However, we need also consider the effect of employing more layers. since the learning capability of a network increases as it learns more compact representation, but it may also make the training more difficult. We need also properly select the type of Attention (either soft or hard-Attention)

and training procedure (either teacher-forcing or injection learning). Finally, we need to tune the type of scoring function, number of neurons, and size of fully connected network layers in the Attention layer.

Bibliography

[AB08] Yaregal Assabie and Josef Bigun. Writer-independent offline recognition of handwritten ethiopic characters. *Proc.* 11^{th} *ICFHR*, pages 652–656, 2008. *14*

[AB09] Yaregal Assabie and Josef Bigun. Hmm-based handwritten amharic word recognition with feature concatenation. In *2009* 10^{th} *International Conference on Document Analysis and Recognition*, pages 961–965. IEEE, 2009. *38, 62, 63*

[AB11] Yaregal Assabie and Josef Bigun. Offline handwritten amharic word recognition. *Pattern Recognition Letters*, 32(8):1089–1099, 2011. *72*

[AEP07] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in neural information processing systems*, pages 41–48, 2007. *64*

[AHB+18] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018. *27*

[Ale97] Worku Alemu. The application of ocr techniques to the amharic script. Master's thesis, School of Information Science, Addis Ababa University, Addis Ababa, 1997. *32, 38, 62, 63, 72, 73*

[All89] Alan Allport. Visual attention. 1989. *22*

[All95] May Allam. Segmentation versus segmentation-free for recognizing arabic text. In *Document Recognition II*, volume 2422, pages 228–235. International Society for Optics and Photonics, 1995. *17*

[ALT18] Direselign Addis, Chuan-Ming Liu, and Van-Dai Ta. Printed ethiopic script recognition by using lstm networks. In *2018 International Conference on System Science and Engineering (ICSSE)*, pages 1–6. IEEE, 2018. *38, 87, 98*

[AM90] Yaser S Abu-Mostafa. Learning from hints in neural networks. *Journal of complexity*, 6(2):192–198, 1990. *64*

[AMKS15] R Anil, K Manjusha, S Sachin Kumar, and KP Soman. Convolutional neural networks for the recognition of malayalam characters. In *Proceedings of the 3^{rd} International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*, pages 493–500. Springer, 2015. *55*

[AMT+18] Solomon Teferra Abate, Michael Melese, Martha Yifiru Tachbelie, Million Meshesha, Solomon Atinafu, Wondwossen Mulugeta, Yaregal Assabie, Hafte Abera, Binyam Ephrem Seyoum, Tewodros Abebe, et al. Parallel corpora for bi-lingual english-ethiopian languages statistical machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3102–3111, 2018. *4*

[Aro85] Mark Aronoff. Orthography and linguistic theory: The syntactic basis of masoretic hebrew punctuation. *Language*, pages 28–72, 1985. *32*

[Asn12] Biniam Asnake. Retrieval from real-life amharic document images. Master's thesis, School of Information Science, Addis Ababa University, 2012. *31*

[Ass02] Yaregal Assabie. Optical character recognition of amharic text: an integrated approach. Master's thesis, Addis Ababa University, Addis Ababa, 2002. *38, 54, 55, 56, 60, 73*

[AT17] Akm Ashiquzzaman and Abdul Kawsar Tushar. Handwritten arabic numeral recognition using deep learning neural networks. In *2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 1–4. IEEE, 2017. *54, 55*

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. *22, 23, 24, 26, 90, 92, 116, 117, 119*

[BCFX14] Jinfeng Bai, Zhineng Chen, Bailan Feng, and Bo Xu. Image character recognition using deep convolutional neural network learned from different languages. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 2560–2564. IEEE, 2014. *54, 55*

[BHL+19a] B. H. Belay, T. Habtegebrial, M. Liwicki, G. Belay, and D. Stricker. Amharic text image recognition: Database, algorithm, and analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1268–1273. IEEE, Sep. 2019. *3, 43, 86, 87, 90, 91, 98, 101*

[BHL+19b] Birhanu Belay, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, and Didier Stricker. Factored convolutional neural network for amharic character image recognition. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2906–2910. IEEE, 2019. *19, 32, 72, 98*

[BHM+20] Birhanu Belay, Tewodros Habtegebrial, Million Meshesha, Marcus Liwicki, Gebeyehu Belay, and Didier Stricker. Amharic ocr: An end-to-end learning. *Applied Sciences*, 10(3):1117, 2020. *32, 91, 93, 98*

[BHS18] Birhanu Hailu Belay, Tewodros Amberbir Habtegebrial, and Didier Stricker. Amharic character image recognition. In *2018 IEEE* 18^{th} *International Conference on Communication Technology (ICCT)*, pages 1179–1182. IEEE, 2018. *62, 67, 72, 98*

[Blu16] Théodore Bluche. Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. In *Advances in Neural Information Processing Systems*, pages 838–846, 2016. *92*

[Bre08] Thomas M Breuel. The ocropus open source ocr system. In *Document Recognition and Retrieval XV*, volume 6815, page 68150F. International Society for Optics and Photonics, 2008. *3, 44, 55, 56*

[Bre17] Thomas M Breuel. High performance text recognition using a hybrid convolutional-lstm implementation. In *2017* 14^{th} *IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 11–16. IEEE, 2017. *72, 98*

[BSS17] Erik Bochinski, Tobias Senst, and Thomas Sikora. Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3924–3928. IEEE, 2017. *19*

[BT14] Henry S Baird and Karl Tombre. The evolution of document image analysis. In *Handbook of document image processing and recognition*, pages 63–71. Springer, 2014. *1*

[BUHAAS13] Thomas M Breuel, Adnan Ul-Hasan, Mayce Ali Al-Azawi, and Faisal Shafait. High-performance ocr for printed english and fraktur using lstm networks. In *2013* 12^{th} *International Conference on Document Analysis and Recognition*, pages 683–687. IEEE, 2013. *54, 55*

[CAC+04] Luke Cole, David Austin, Lance Cole, et al. Visual object recognition using template matching. In *Australian conference on robotics and automation*, 2004. *16*

[CBC19] SA Chekole, F Bekele, and B Chekol. Archival management and preservation in ethiopia: The case of east gojjam. *J Tourism Hospit*, 8(408):2167–0269, 2019. *5*

[CCB15] Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11):1875–1886, 2015. *90*

[CDL16] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016. *117*

[CdSBJB+06] Paulo Rodrigo Cavalin, Alceu de Souza Britto Jr, Flávio Bortolozzi, Robert Sabourin, and Luiz E Soares Oliveira. An implicit segmentation-based method

for recognition of handwritten strings of characters. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 836–840, 2006. *15*

[CH03] John Cowell and Fiaz Hussain. Amharic character recognition using a fast signature based algorithm. In *Information Visualization, 2003. IV 2003. Proceedings. 7th International Conference on*, pages 384–389. IEEE, 2003. *14, 72, 98*

[Cho14] Amit Choudhary. A review of various character segmentation techniques for cursive handwritten words recognition. *Int. J. Inf. Comput. Technol*, 4(6):559–564, 2014. *14*

[CL96] Richard G Casey and Eric Lecolinet. A survey of methods and strategies in character segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 18(7):690–706, 1996. *14*

[Cou03] Florian Coulmas. *Writing systems: An introduction to their linguistic analysis*. Cambridge University Press, 2003. *31, 32*

[CPT+19] J. Cai, L. Peng, Y. Tang, C. Liu, and P. Li. Th-gan: Generative adversarial network based transfer learning for historical chinese character recognition. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 178–183. IEEE, Sep. 2019. *3*

[CRA13] Amit Choudhary, Rahul Rishi, and Savita Ahlawat. A new character segmentation approach for off-line cursive handwritten words. *Procedia Computer Science*, 17:88–95, 2013. *15*

[CV18] Arindam Chowdhury and Lovekesh Vig. An efficient end-to-end neural model for handwritten text recognition. *arXiv preprint arXiv:1807.07965*, 2018. *27, 98*

[CVMG+14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. *116*

[DCBC15] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 845–850, 2015. *63*

[Dem11] Fitsum Demissie. Developing optical character recoginition for ethiopic scripts, 2011. *60*

[DHK13] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8599–8603. IEEE, 2013. *63*

[DJH01] O De Jeses and Martin T Hagan. Backpropagation through time for a general class of recurrent network. In *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, volume 4, pages 2638–2643. IEEE, 2001. *19*

[DLY+19] Amit Das, Jinyu Li, Guoli Ye, Rui Zhao, and Yifan Gong. Advancing acoustic-to-word ctc model with attention and mixed-units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12):1880–1892, 2019. *90*

[DT+14] David Doermann, Karl Tombre, et al. *Handbook of document image processing and recognition*. Springer, 2014. *1*

[DZN16] Patrick Doetsch, Albert Zeyer, and Hermann Ney. Bidirectional decoder networks for attention-based end-to-end offline handwriting recognition. In *2016* 15^{th} *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 361–366. IEEE, 2016. *92*

[EF20] Gary F. Simons Eberhard, David M. and Charles D. Fennig. *Ethnologue: Languages of the World*. SIL International, Dallas, Texas, United States, twenty-third edition, 2020. *29*

[Eik93] Line Eikvil. Optical character recognition. *citeseer. ist. psu. edu/142042. html*, 1993. *14, 16*

[ETK16] Mohamed Elleuch, Najiba Tagougui, and Monji Kherallah. A novel architecture of cnn based on svm classifier for recognising arabic handwritten script. *International Journal of Intelligent Systems Technologies and Applications*, 15(4):323–340, 2016. *62*

[GCBG08] Emmanuèle Grosicki[1], Matthieu Carre, Jean-Marie Brodin, and Edouard Geoffrois[1]. Rimes evaluation campaign for handwritten mail processing. 2008. *43*

[GDS10] Debashis Ghosh, Tulika Dube, and Adamane Shivaprasad. Script recognition—a review. *IEEE Transactions on pattern analysis and machine intelligence*, 32(12):2142–2161, 2010. *54, 55*

[GFGS06] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the* 23^{rd} *international conference on Machine learning*, pages 369–376, 2006. *20, 21, 74, 84*

[Gla17] Tobias Glasmachers. Limits of end-to-end learning. *arXiv preprint arXiv:1704.08305*, 2017. *86*

[GLB+08] Alex Graves, Marcus Liwicki, Horst Bunke, Jürgen Schmidhuber, and Santiago Fernández. Unconstrained on-line handwriting recognition with recurrent neural networks. In *Advances in neural information processing systems*, pages 577–584, 2008. *72, 90, 98, 99*

[GMH13] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013. *19*

[Gra08] A Graves. Supervised sequence labelling with recurrent neural networks [ph. d. dissertation]. *Technical University of Munich, Germany*, 2008. *102*

[GSR15] Shivansh Gaur, Siddhant Sonkar, and Partha Pratim Roy. Generation of synthetic training data for handwritten indic script recognition. In *2015* 13^{th} *International Conference on Document Analysis and Recognition (ICDAR)*, pages 491–495. IEEE, 2015. *72*

[GSTBJ19] Mesay Samuel Gondere, Lars Schmidt-Thieme, Abiot Sinamo Boltena, and Hadi Samer Jomaa. Handwritten amharic character recognition using a convolutional neural network. *arXiv preprint arXiv:1909.12943*, 2019. *38, 72*

[GVK19] Rajib Ghosh, Chirumavila Vamshi, and Prabhat Kumar. Rnn based online handwritten word recognition in devanagari and bengali scripts using horizontal zoning. *Pattern Recognition*, 92:203–218, 2019. *81, 91*

[GVZ16] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324, 2016. *42*

[HA13] Abraham Hailu and Yaregal Assabie. Itemsets-based amharic document categorization using an extended a priori algorithm. In *Language and Technology Conference*, pages 317–326. Springer, 2013. *32*

[HA17] Seid Ali Hassen and Yaregal Assabie. Recognition of double sided amharic braille documents. *International Journal of Image, Graphics and Signal Processing*, 9(4):1, 2017. *62*

[Hag15] Tesfahun Hagos. Ocr system for the recognition of ethiopic real life documents. Master's thesis, Bahir Dar Institute of Technology, 2015. *3, 55*

[Han17] Awni Hannun. Sequence modeling with ctc. *Distill*, 2(11):e8, 2017. *21*

[Her82] H Herbert. The history of ocr, optical character recognition. *Manchester Center, VT: Recognition Technologies Users Association*, 1982. *1*

[HKR15] Samer Hijazi, Rishi Kumar, and Chris Rowen. Using convolutional neural networks for image recognition. *Cadence Design Systems Inc.: San Jose, CA, USA*, pages 1–12, 2015. *81*

[HM16] Birhanu Hailu and Million Meshesha. Applying image processing for malt-barley seed identification. In *Conference: Ethiopian the 9th ICT Annual Conference*, 2016. *18*

[HML+18] Shizhong Han, Zibo Meng, Zhiyuan Li, James O'Reilly, Jie Cai, Xiaofeng Wang, and Yan Tong. Optimizing filter size in convolutional neural networks

for facial action unit recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5070–5078, 2018. *19*

[HOT06] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. *18*

[HS95] Ke Han and Ishwar K Sethi. Off-line cursive handwriting segmentation. In *Proceedings of* 3^{rd} *International Conference on Document Analysis and Recognition*, volume 2, pages 894–897. IEEE, 1995. *14*

[HS06] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006. *18*

[HZMJ15] Meijun He, Shuye Zhang, Huiyun Mao, and Lianwen Jin. Recognition confidence analysis of handwritten chinese character with cnn. In *2015* 13^{th} *International Conference on Document Analysis and Recognition (ICDAR)*, pages 61–65. IEEE, 2015. *55*

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. *18, 72*

[IFD+19] R. R. Ingle, Y. Fujii, T. Deselaers, J. Baccash, and A. C. Popat. A scalable handwritten text recognition system. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 17–24. IEEE, Sep. 2019. *3*

[IKN98] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998. *113*

[Jam07] William James. *The principles of psychology*, volume 1. Cosimo, Inc., 2007. *22*

[KJEY20] Mohamed Ibn Khedher, Houda Jmila, and Mounim A El-Yacoubi. Automatic processing of historical arabic documents: A comprehensive survey. *Pattern Recognition*, 100:107144, 2020. *2, 54*

[KSBH16] Pavel Kisilev, Eli Sason, Ella Barkan, and Sharbell Hashoul. Medical image captioning: learning to describe medical image findings using multi-task-loss cnn. *Deep Learning and Data Labeling for Medical Applications:in proceeding of International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2016. *63*

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. *18*

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. *18, 43*

[LFZ+15] Tianyi Liu, Shuangsang Fang, Yuehui Zhao, Peng Wang, and Jun Zhang. Implementation of training convolutional neural networks. *arXiv preprint arXiv:1506.01195*, 2015. *18*

[LGF+07] Marcus Liwicki, Alex Graves, Santiago Fernàndez, Horst Bunke, and Jürgen Schmidhuber. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In *Proceedings of the* 9^{th} *International Conference on Document Analysis and Recognition, ICDAR 2007*, 2007. *19, 20, 72*

[LNNN17] Nam-Tuan Ly, Cuong-Tuan Nguyen, Kha-Cong Nguyen, and Masaki Nakagawa. Deep convolutional recurrent network for segmentation-free offline handwritten japanese text recognition. In *2017* 14^{th} *IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 7, pages 5–9. IEEE, 2017. *72, 91, 98*

[LO16] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2231–2239, 2016. *98*

[LPM15] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015. *19, 23, 24, 90, 92, 113, 117, 118, 119, 120*

[LTD+17] Linghui Li, Sheng Tang, Lixi Deng, Yongdong Zhang, and Qi Tian. Image caption with global-local attention. In *Proceedings of the thirty-first AAAI conference on artificial intelligence*, pages 4133–4139, 2017. *27*

[LV12] Hong Lee and Brijesh Verma. Binary segmentation algorithm for english cursive handwriting recognition. *Pattern Recognition*, 45(4):1306–1317, 2012. *14*

[MB02] U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46, 2002. *42*

[MBP15] Durjoy Sen Maitra, Ujjwal Bhattacharya, and Swapan K Parui. Cnn based common approach to handwritten character recognition of multiple scripts. In *Document Analysis and Recognition (ICDAR), 2015* 13^{th} *International Conference on*, pages 1021–1025. IEEE, 2015. *62*

[Mes08] Million Meshesha. Recognition and retrieval from document image collections. *Ph. D. Dissertation, International Institute of Information Technology (IIIT)*, 2008. *3, 5, 14, 29, 31, 37, 54, 62, 63*

[Mey06] Ronny Meyer. Amharic as lingua franca in ethiopia. *Lissan: Journal of African Languages and Linguistics*, 20(1/2):117–132, 2006. *3, 30*

[MF06] R Manmatha and Shaolei Feng. A hierarchical, hmm-based automatic evaluation of ocr accuracy for a digital library of books. In *Proceedings of the* 6^{th}

ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'06), pages 109–118. IEEE, 2006. *18*

[Mil00] M Million. A generalized approach to optical character recognition of amharic texts. Master's thesis, School of Information Science, Addis Ababa University, Addis Ababa, 2000. *38*

[MJ05] Million Meshesha and CV Jawahar. Recognition of printed amharic documents. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pages 784–788. IEEE, 2005. *29*

[MJ07] Million Meshesha and CV Jawahar. Optical character recognition of amharic documents. *African Journal of Information & Communication Technology*, 3(2), 2007. *14, 30, 32, 54, 55, 56, 60, 61, 62, 72, 73, 98*

[ML15] Ronaldo Messina and Jerome Louradour. Segmentation-free handwritten chinese text recognition with lstm-rnn. In *2015* 13^{th} *International Conference on Document Analysis and Recognition (ICDAR)*, pages 171–175. IEEE, 2015. *72, 98*

[MM18] Getahun Tadesse Mekuria and Getahun Tadesse Mekuria. Amharic text document summarization using parser. *International Journal of Pure and Applied Mathematics*, 118(24), 2018. *4, 30*

[MSLB98] John Makhoul, Richard Schwartz, Christopher Lapre, and Issam Bazzi. A script-independent methodology for optical character recognition. *Pattern Recognition*, 31(9):1285–1294, 1998. *18*

[MZ17] Tirin Moore and Marc Zirnsak. Neural mechanisms of selective visual attention. *Annual review of psychology*, 68:47–72, 2017. *113*

[Nag92] George Nagy. At the frontiers of ocr. *Proceedings of the IEEE*, 80(7):1093–1100, 1992. *17*

[Nag00] George Nagy. Twenty years of document image analysis in pami. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (1):38–62, 2000. *14*

[NAL] The ethiopian national archive and liberary agency. `http://www.nala.gov.et/home`. Accessed on 10 Feburary 2021. *5*

[NWC+11] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 5, 2011. *43*

[Pal05] Mahesh Pal. Random forest classifier for remote sensing classification. *International journal of remote sensing*, 26(1):217–222, 2005. *18*

[PC00] U Pal and BB Chaudhuri. Automatic recognition of unconstrained off-line bangla handwritten numerals. In *International Conference on Multimodal Interfaces*, pages 371–378. Springer, 2000. *31*

[PC04] U Pal and BB Chaudhuri. Indian script character recognition: a survey. *pattern Recognition*, 37(9):1887–1899, 2004. *14*

[PPCA13] Christos Papadopoulos, Stefan Pletschacher, Christian Clausner, and Apostolos Antonacopoulos. The impact dataset of historical document images. In *Proceedings of the* 2^{nd} *international workshop on historical document imaging and processing*, pages 123–130, 2013. *43*

[pre] The ethiopian press agency. will amharic be au's lingua franca ? `https://www.press.et/english/?p=2654#1`. Accessed on 10 November 2019. *30*

[PV17] Jason Poulos and Rafael Valle. Character-based handwritten text transcription with attention networks. *arXiv preprint arXiv:1712.04046*, 2017. *27, 98*

[PZLL20] Xushan Peng, Xiaoming Zhang, Yongping Li, and Bangquan Liu. Research on image feature extraction and retrieval algorithms based on convolutional neural network. *Journal of Visual Communication and Image Representation*, 69:102705, 2020. *19*

[RBO17] Christophe Rigaud, Jean-Christophe Burie, and Jean-Marc Ogier. Segmentation-free speech text recognition for comic books. In *2017* 14^{th} *IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 3, pages 29–34. IEEE, 2017. *72*

[RMS09] Amjad Rehman, Dzulkifli Mohamad, and Ghazali Sulong. Implicit vs explicit based script segmentation and recognition: a performance comparison on benchmark database. *Int. J. Open Problems Compt. Math*, 2(3):352–364, 2009. *15*

[RRB18] Betselot Yewulu Reta, Dhara Rana, and Gayatri Viral Bhalerao. Amharic handwritten character recognition using combined features and support vector machine. In *2018* 2^{nd} *International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 265–270. IEEE, 2018. *38, 62, 63, 72, 73*

[RSWP18] Christian Reul, Uwe Springmann, Christoph Wick, and Frank Puppe. State of the art optical character recognition of 19^{th} century fraktur scripts using open source engines. *arXiv preprint arXiv:1810.03436*, 2018. *73*

[RTAS20] António H Ribeiro, Koen Tiels, Luis A Aguirre, and Thomas Schön. Beyond exploding and vanishing gradients: analysing rnn training using attractors and smoothness. In *International Conference on Artificial Intelligence and Statistics*, pages 2370–2380, 2020. *19*

[SA15] Maad Shatnawi and Sherief Abdallah. Improving handwritten arabic character recognition by modeling human handwriting distortions. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 15(1):1–12, 2015. *56*

[SG97] Zhixin Shi and Venu Govindaraju. Segmentation and recognition of connected handwritten numeral strings. *Pattern Recognition*, 30(9):1501–1504, 1997. *14*

[SIK+09] Fouad Slimane, Rolf Ingold, Slim Kanoun, Adel M Alimi, and Jean Hennebert. A new arabic printed text image database and evaluation protocols. In *2009 10^{th} International Conference on Document Analysis and Recognition*, pages 946–950. IEEE, 2009. *42*

[SJS13] P Sibi, S Allwyn Jones, and P Siddarth. Analysis of different activation functions using back propagation neural networks. *Journal of theoretical and applied information technology*, 47(3):1264–1268, 2013. *18*

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. *18*

[SS13] Nazly Sabbour and Faisal Shafait. A segmentation-free approach to arabic and urdu ocr. In *Document Recognition and Retrieval XX*, volume 8658, page 86580N. International Society for Optics and Photonics, 2013. *42*

[SSSS17] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *Proceedings of the 34^{th} International Conference on Machine Learning-Volume 70*, pages 3067–3075. JMLR. org, 2017. *86*

[STA+18] Palaiahnakote Shivakumara, Dongqi Tang, Maryam Asadzadehkaljahi, Tong Lu, Umapada Pal, and Mohammad Hossein Anisi. Cnn-rnn based method for license plate recognition. *CAAI Transactions on Intelligence Technology*, 3(3):169–175, 2018. *91*

[SUHP+15] Fotini Simistira, Adnan Ul-Hassan, Vassilis Papavassiliou, Basilis Gatos, Vassilis Katsouros, and Marcus Liwicki. Recognition of historical greek polytonic scripts using lstm networks. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 766–770. IEEE, 2015. *73*

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014. *24, 113, 115*

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. *18, 55*

[Tef99] Dereje Teferi. Optical character recognition of typewritten amharic text. Master's thesis, School of Information Science, Addis Ababa University, Addis Ababa, 1999. *38, 54, 55, 60, 62, 63, 72, 73*

[Tes10] Abay Teshager. Amharic character recognition system for printed real-life documents. Master's thesis, School of Information Science, Addis Ababa University, Addis Ababa, 2010. *32*

[TT18] Zhiqiang Tong and Gouhei Tanaka. Reservoir computing with untrained convolutional neural networks for image recognition. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1289–1294. IEEE, 2018. *19*

[UH16] Adnan Ul-Hasan. Generic text recognition using long short-term memory networks. *Ph. D. Dissertation, Technical University of Kaiserslautern*, 2016. *17*

[UHAR+13] Adnan Ul-Hasan, Saad Bin Ahmed, Faisal Rashid, Faisal Shafait, and Thomas M Breuel. Offline printed urdu nastaleeq script recognition with bidirectional lstm networks. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1061–1065. IEEE, 2013. *73*

[VMN+16] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. In *arXiv preprint arXiv:1601.07140*, 2016. *43*

[Wer90] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. *19*

[WHK+17] Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253, 2017. *90*

[Wio06] Anais Wion. The national archives and library of ethiopia: six years of ethio-french cooperation (2001-2006). 2006. *4*

[WSW+17] Qi Wu, Chunhua Shen, Peng Wang, Anthony Dick, and Anton van den Hengel. Image captioning and visual question answering based on attributes and external knowledge. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1367–1381, 2017. *20*

[WYCL17] Yi-Chao Wu, Fei Yin, Zhuo Chen, and Cheng-Lin Liu. Handwritten chinese text recognition using separable multi-dimensional recurrent neural network. In *2017* 14^{th} *IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 79–84. IEEE, 2017. *72, 98*

[XBK+15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015. *23, 24, 117, 118, 119*

[YBJL12] Aiquan Yuan, Gang Bai, Lijing Jiao, and Yajie Liu. Offline handwritten english character recognition based on convolutional neural network. In *2012* 10^{th} *IAPR International Workshop on Document Analysis Systems*, pages 125–129. IEEE, 2012. *54, 55, 91*

[YNDT18] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018. *19*

[YZX+18] Min Yang, Wei Zhao, Wei Xu, Yabing Feng, Zhou Zhao, Xiaojun Chen, and Kai Lei. Multitask learning for cross-domain image captioning. *IEEE Transactions on Multimedia*, 2018. *63*

[ZCW10] Shusen Zhou, Qingcai Chen, and Xiaolong Wang. Hit-or3c: an opening recognition corpus for chinese characters. In *Proceedings of the* 9^{th} *IAPR International Workshop on Document Analysis Systems*, pages 223–230, 2010. *43*

[ZDD18] Jianshu Zhang, Jun Du, and Lirong Dai. Track, attend, and parse (tap): An end-to-end framework for online handwritten mathematical expression recognition. *IEEE Transactions on Multimedia*, 21(1):221–233, 2018. *20, 27, 98*

[ZHZ17] Haifeng Zhao, Yong Hu, and Jinxia Zhang. Character recognition via a compact convolutional neural network. In *2017 International conference on digital image computing: techniques and applications (DICTA)*, pages 1–6. IEEE, 2017. *55, 56*

[ZLLT14] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision*, pages 94–108. Springer, 2014. *63*

[ZY17] Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017. *64*

[ZYDS17] Geoffrey Zweig, Chengzhu Yu, Jasha Droppo, and Andreas Stolcke. Advances in all-neural speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4805–4809. IEEE, 2017. *20*

Curriculum Vitae

Full Name: **Birhanu Hailu Belay**

Work Experience

October 2017–Now	**PhD Student TU Kaiserslautern, DFKI, Kaiserslautern, Germany**
January 2013 – September 2017	**Assistant Lecture, Bahir Dar Institute of Technology, Ethiopia**
September 2011 – December 2012	**Graduate Assistant, Deberetabor University, Ethiopia**
September 2010 – August 2011	**Graduate Assistant, Arbaminch University, Ethiopia**

Education

October 2017 – Now	**PhD in Computer Science**, TU Kaiserslautern, Germany Thesis: *"Deep Learning for Amharic Text-image Recognition: Algorithm, Dataset and application"*
September 2013 – October 2015	**Master of Science in Computer Science**, Bahir Dar Institute of Technology, Ethiopia Thesis: *"Applying Image processing techniques for Malt-barley Seed Identification"*
September 2007 – July 2010	**Bachelor of Science Information Communication Technology**, Wollo University, Ethiopia Thesis/project: *"Patient Registration and Medical Records Management System"*

Computer Skill

Programming	Python, Java, C++
Web development	CSS, HTML, PHP, Javascript, JSP
Database	Microsoft SQL Server, Microsoft Access
Development Environments	Matlab, Macromedia Dreamweaver 8, VisualStudio
Operating Systems	Linux and Windows operating system
Documentation softwares	MS Office products, Latex
Deep learning frameworks	Keras, Tensorflow

Publication list

Journal articles

1. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Million Meshesha, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Amharic OCR: An End-to-End Learning". *MDPI-Applied Sciences. 2020; 10(3):1117.*

2. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Gebeyehu Belay, Million Meshesha, Marcus Liwicki, Didier Stricker, "Learning by Injection: Attention Embedded Recurrent Neural Network for Amharic Text-image Recognition". *Preprints, 2020*

Conference papers

3. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Blended Attention-CTC for Amharic Text-image Recognition: Sequence-to-Sequence Learning", 10^{th} International Conference on Pattern Recognition Application and Methods (ICPRAM 2021)

4. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Gebeyehu Belay, Didier Stricker, "Using Automatic Features for Text-image Classification in Amharic Documents", 9^{th} International Conference on Pattern Recognition Applications and Methods. International Conference on Pattern Recognition Applications and Methods (ICPRAM-2020) February 22-24 Valletta, Malta 2020.

5. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Factored Convolutional Neural Network for Amharic Character Image Recognition", 26^{th} IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 2906–2910.

6. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Marcus Liwicki, Gebeyehu Belay, Didier Stricker, "Amharic Text Image Recognition: Dataset, Algorithm and Analysis", 15^{th} International Conference Document Analysis and Recognition (ICDAR), Sydney, Australia, 20–25 September 2019; pp. 1268–1273.

7. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Gebeyehu Belay, Didier Stricker, "Multi-font Printed Amharic Character Image Recognition: Deep Learning Techniques", 6th EAI International Conference on Advances of Science and

Technology: ICAST 2018, Bahir Dar, Ethiopia, October 5-7, 2018, Proceedings, vol. 274, p. 322. Springer, 2019

8. **Birhanu Hailu Belay**, Tewodros Habtegebrial, Didier Stricker, "Amharic character image recognition", 18^{th} IEEE International Conference on Communication Technology (ICCT), Chongqing, China, 8–11 October 2018; pp. 1179–1182

9. **Birhanu Hailu Belay**, Million Meshesha: "Application of Image Processing Techniques for Malt-barley Seed Identification", 9^{th} ICT Annual Conference 2016 (EICTAC 2016)

10. **Birhanu Hailu Belay**, Tesfa Tegegne, "Ethiopian Roasted Coffee Classification Using Imaging Techniques", 3^{rd} International Conference on the Advancement of Science and Technology (ICAST2015), pp. 23-29, Bahir Dar, Ethiopia, 2015.