Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach

Felmeta Abate Jilo

A Thesis submitted to Department of Computer Science and Engineering,

School of Electrical Engineering and computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June 2023,

Adama, Ethiopia

Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach

Felmeta Abate Jilo

Advisor: **Teklu Urgessa (PhD)**

A Thesis submitted to Department of Computer Science and Engineering,

School of Electrical Engineering and computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June 2023,

Adama, Ethiopia

Declaration

I hereby declare that this Master Thesis entitled "**Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Felmeta Abate	_____	_____
Name of the student	Signature	Date

Recommendation

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach''** prepared under my/our guidance by Felmeta Abate submitted in partial fulfillment of the requirements for the degree of Mater's of Science in Computer Science and Engineering.

Therefore, I/we recommend the submission of a revised version of the thesis to the department following the applicable procedures.

Teklu Urgessa (PhD)	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-advisor	Signature	Date

Approval Page of M.Sc. Thesis

I/we, the advisors of the thesis entitled "**Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach**" and developed by Felmeta Abate, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Teklu Urgessa (PhD)	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Felmeta Abate have read and evaluated the thesis entitled "**Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach**" and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

_____	_____	_____
Chairperson	Signature	Date
_____	_____	_____
Internal Examiner	Signature	Date
_____	_____	_____
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____	_____	_____
Department Head	Signature	Date
_____	_____	_____
School Dean	Signature	Date
_____	_____	_____
Office of Postgraduate Studies, Dean	Signature	Date

ACKNOWLEDGEMENTS

First and foremost, I express my heartfelt gratitude to Almighty Allah for providing me with the wisdom, strength, and to bring this thesis to completion. I would like to extend my heartfelt appreciation to my esteemed advisor, Dr. Teklu Urgessa, for his invaluable guidance, expertise, and unwavering commitment to my academic and professional growth, mentorship, which played a crucial role in the successful completion of this study.

I would also like to acknowledge the dedicated staff members of Oda-Butum University, Mada-Welabu University, and Dilla University department of Afaan Oromoo who participated in the Mean Opinion Score (MOS) evaluation, offering their time and expertise to assess the quality of the synthesized speech. Their contributions were instrumental in obtaining valuable feedback and insights.

Furthermore, I would like to express my gratitude to Mr. Yesuf Hussein, a respected language expert and lecturer at Dilla University's Department of Afaan Oromoo. His extensive knowledge, guidance, and support in the language aspects of our research significantly contributed to the accuracy and fluency of the Afaan Oromoo text synthesized in our model. Finally am deeply indebted to all individuals and institutions that played a role in supporting and facilitating our research. Their contributions were invaluable, and i sincerely grateful for their assistance and collaboration.

TABLE OF CONTENT

Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AAU = Addis Ababa University

AI = Artificial Intelligence

ANN = Artificial Neural Network

AOR = Adjusted Odds Ratio

API = Application Program Interface

BLSTM = Bi-Directional Long Short-Term Memory

BR = Braille Reading

BW = Braille Writing

CA = Character Accuracy

CER = Character Error Rate

CI = Contrast Index

CNN = Convolutional Neural Network

CPU = Central Processing Unit

CUDA = Computer Unified Device Architecture

ERCO = Ethiopian Residents Charity Organization

FFT = Fast Fourier Transform

GPS = Global Positioning System

GPU = Graphics Processing Unit

GRU = Gated Recurrent Unit

GSTs = Global Style Tokens

HEI = Higher Education Institute

HIFI-GAN = High Fidelity Generative Adversarial Network

HMM = Hidden Markov Model

LSTM = Long Short-Term Memory

MFCC = Mel Frequency Cepstral Coefficients

MCD = Mel Cepstral Distortion

MOS = Mean Opinion Score

MPD = Multi-Period Discriminator

MRF = Multi-Receptive Field Fusion

MSD = Multi-Scale Discriminator

NLTK = Natural Language Toolkit

OCR = Optical Character Recognition

OOV = Out-Of-Vocabulary

PCM = Pulse Code Modulation

PDF = Portable Document Format

PIL = Python Image Library

PSU = Power Supply Unit

RAM = Random Access Memory

RNN = Recurrent Neural Network

ReLU = Rectified Linear Unit

SA = Symbol Accuracy

SD = Sensory Disorders

SER = Special Symbol Error Rate

SNE = Special Needs Education

STFT = Short-Time Fourier Transform

SVO = Subject-Object-Verb

Seq2Seq = Sequence To Sequence

TTS = Text To Speech

VI = Visual Impairment

WER = Word Error Rate

WHO = World Health Organization

σT = Standard Deviation of Time Taken to Perform OCR

σAC = Standard Deviation of Accuracy

ABSTRACT

This study aims to develop an Afaan Oromoo Text Reading model for visually impaired individuals by integrating Tesseract Optical Character Recognition (OCR) with Recurrent Neural Network based Tacotron2 model using deep neural network approaches. The research addresses the challenges faced by visually impaired people in accessing written text and proposes a solution to enable them to hear Afaan Oromoo text through synthesized speech. The existing Afaan Oromoo Text-To-Speech studies have limitations in terms of intelligibility and naturalness. Most of these methods rely on traditional techniques such as concatenative synthesis, while the latest study utilizes Griffin-Lim algorithm as a vocoder, which is traditional compared to utilization of more advanced flow-based generative models. This study addresses the need for improved text-to-speech synthesis for the Afaan Oromoo language. To extract Afaan Oromoo text from images, we employ Tesseract OCR engine, which demonstrates accurate text extraction capabilities. To convert extracted text into speech, we used a high-quality text-to-speech synthesis Tacotron2 model. Additionally, we employ WaveGlow and High Fidelity Generative Adversarial Network (HIFI-GAN) vocoders to improve the quality and naturalness of the synthesized speech. Dataset for model training is collected from various sources, including news, religious books, and benchmark dataset is utilized to enhance the Model training with 9000 sentences and transcribed audio datasets. The developed model was evaluated using subjective evaluation mean opinion score (MOS) and objective evaluation Mel Cepstral Distortion (MCD) using dynamic range of root mean square error (RMSE) assessments. Tacotron2 model with WaveGlow vocoder achieved MOS scores of 4.11 for intelligibility and 3.87 for naturalness and MCD evaluation scored 5.29%. On the other hand, Tacotron2 model with HIFI-GAN vocoder achieved MOS scores of 4.39 for intelligibility and 4.35 for naturalness and MCD evaluation scored 3.16%. These results indicate the high quality speech synthesis of Afaan Oromoo Text when using Tacotron2 model with flow-based generative vocoders. This study contributes to the field of assistive technologies by integrating Tesseract OCR, Tacotron2 model, and HIFI-GAN vocoder, which allows visually impaired as well as individuals with reading difficulties to access Afaan Oromoo written text through synthesized speech.

Keywords: *Afaan Oromoo, Visually Impaired, Tesseract OCR, Text to Speech, Tacotron2, WaveGlow, HIFI-GAN*

CHAPTER ONE

1. INTRODUCTION

1.1 Background of the Study

Blindness is a condition in which an individual loses the ocular perception. According to recent studies, Approximately 4 million individuals in Ethiopia experience visual impairment, with 1.6% of them being blind and 3.7% having low vision. Visual impairment affects women more significantly than men. Furthermore, approximately 6% of the blind population comprises children(Together Civil Society, 2020).

Furthermore, visually impaired people face many challenges in their every-day life, and a number of these could be relieved to some degree via the use of an aiding device based on artificial vision systems(Lara-alvarez & Bayro-corrochano, 2014). The current trend in technological development is to take advantage of cameras found in smartphones, portable computing devices, and wearable devices that offer several complementing solutions for the user (Lara-alvarez & Bayro-corrochano, 2014).

The issue of equal participation in all aspects of life, particularly education, is a growing global concern. While students with disabilities are the most recent marginalized group to achieve equal opportunity, students with blindness face far more serious challenges in Ethiopian higher education institutions. According to a study conducted at AAU on Academic Challenges of Visually Challenged Students in Addis Ababa University Ethiopia, AAU is still a long way from satisfying visually challenged students for academic success because it has yet to establish a structure to assist them(Mulugeta & Mekuriaw, 2017). Yihun and Belay, in their findings, show that even though some opportunities support inclusive education, these opportunities do not guarantee their implementation effectively(Yihun & Belay, 2020).

The study, which aimed to assess students' "Braille Reading (BR) and Braille Writing (BW) skills and identify challenges impeding those skills," found that students lacked awareness and support in BR and BW activities, the SNE resource center was disorganized, and learners lacked experience using their fingers during BR. Other challenges that contributed to students' poor Braille skills included a lack of Braille-skilled teachers and principals, as well as a school promotion policy(Mitiku, 2020).

The study aimed to address the need for reading written texts for visually impaired individuals in the Afaan Oromoo language. The study aimed to enhance the reading experience of visually impaired individuals and also improve intelligibility and naturalness of Afaan Oromoo text-to-speech synthesis by leveraging deep learning techniques. This field of study was dependent on language, and most efforts had focused on accessible languages, as a result, languages with limited resources received less attention. Additionally, existing Afaan Oromoo TTS systems have employed various techniques, including Hidden Markov Models (HMMs) and RNN based Tacotron2 model using deep learning approach with Griffin-Lim algorithm as a vocoder. However, these approaches have certain limitations.

HMM-based TTS systems struggle to capture the intricate nuances of speech, such as prosody, intonation, and timing, resulting in less natural and expressive synthesized speech(Abraham et al., 2022). Similarly, the Griffin-Lim algorithm, a traditional vocoder, has limitations in generating high-quality speech compared to more advanced flow-based generative models. It lacks explicit control over voice characteristics, such as pitch and prosody, and may not capture fine-grained details accurately(Masuyama et al., 2019). These limitations highlight the need for more advanced techniques in TTS systems to overcome the shortcomings of HMMs and traditional vocoders Griffin-Lim by exploring alternative deep learning-based models and flow-based generative models.

The proposed solution employed two-step to achieve the objectives. Firstly, it employed advanced methods for extracting text from images containing Afaan Oromoo texts, effectively separating the textual content and saving it as a separate text file. This process involved image pre-processing and utilizing deep learning algorithms using optical character recognition (OCR) techniques, Tesseract OCR, to accurately extract the text. Secondly, the study focused on converting the extracted text into Afaan Oromo speech, enabling visually impaired individuals to listen to the content of the document through synthesized speech. This step involved utilizing sophisticated text-to-speech synthesis models, Tacotron2, combined with vocoders WaveGlow and HIFI-GAN, to generate high-quality and natural-sounding speech.

To evaluate the effectiveness of the developed model, the study employed rigorous evaluation measures. The performance of the text extraction process using Tesseract OCR was assessed with the word error rate as a key evaluation metric. The intelligibility and naturalness of the

synthesized speech were evaluated through subjective assessments using the Mean Opinion Score (MOS) metric, providing insights into the intelligibility and naturalness of the generated speech. The Mean Opinion Score (MOS) is a widely used subjective metric used to evaluate the quality of synthesized speech(Leng et al., 2021). It involves multiple human judges who assess each speech utterance and provide their subjective ratings. Additionally, objective evaluation method Mel Cepstral Distortion (MCD) was utilized to measure the similarity between the generated speech and the target speech. The results of this research attempt to bring significant implications for the progress of inclusive technology and the creation of text-reading models customized to the requirements of visually impaired individuals. Additionally, the findings of this study hold promise for individuals facing reading difficulties, such as dyslexia or cognitive impairments. Furthermore, the ability to generate audio content from written text shows a range of possibilities, including the production of audiobooks, podcasts, voiceovers, and various other forms of audio-based media.

1.2 Statement of Problem

Despite significant advancements in assistive technology for the visually impaired, there remained a shortage of text reading materials available in regional languages like Afaan Oromoo(Mulugeta & Mekuriaw, 2017; Yihun & Belay, 2020). This created a major obstacle for visually impaired individuals who relied on written information to participate fully in society and access education, employment, and other essential resources.

Individuals with visual impairments rely on the braille system as their primary method of reading. The braille system consists of 63 character codes, each represented by a combination of 1 to 6 raised dots arranged in specific patterns within cells. Braille characters are embossed onto paper, and individuals read them by carefully running their fingertips over the raised dots. However, mastering braille can be challenging as it requires dedicated learning. Furthermore, not everyone possesses fingertip sensitivity necessary to effectively utilize this physical reading system, and also there are limitations to getting books using Braille in the market(Kate Beck, 2018). It was discovered that the growth, personal dignity, and well-being of Ethiopian visually impaired individuals appear to be affected since they live in a setting with a variety of social, psychological, and educational issues(Tolla, 2022).

The current solutions for this challenge often involved manual transcription of written materials into audio format(Brailleworks.com, 2023; Scotland, 2023), which was time-consuming and expensive. Additionally, many existing text-to-speech solutions were not tailored to regional languages like Afaan Oromoo and existing Afaan Oromoo TTS methods are limited in both their intelligibility and naturalness. Most of these methods rely on traditional techniques such as concatenative synthesis, which struggle to capture the nuances of the language, while the latest study utilizes the traditional Griffin-Lim algorithm as a vocoder, in contrast to the utilization of more sophisticated flow-based generative models. These challenges highlighted the need for a practical, efficient, and effective solution to the issue of reading written text in the Afaan Oromoo language.

This study aimed to address this issue by proposing the development of a deep learning-based text-reading model specifically designed for the Afaan Oromoo language. By using a deep learning approach, the model was able to accurately extract and translate extracted Afaan Oromoo text into an audio format, which aim to offer visually impaired individuals an audio representation of written text that is both understandable and sounds natural.

1.3 Research Questions

This study aimed to address the following research questions.

Q1: How can we improve the quality of speech synthesis?

Q2: How to develop a model that can efficiently read Afaan Oromoo text?

Q3: To what extent can the newly developed model improves the quality of speech synthesis?

1.4 Objectives of the Study

1.4.1 General Objective

The general objective of this study is to develop a model for reading Afaan Oromoo Text that will help visually challenged individuals understand and listen to the content of written text in Afaan Oromoo.

1.4.2 Specific Objectives

The Following specific objectives have to be achieved in-order to fulfill the general objective.

- ✓ To Identify and collect the necessary data for the development of the model;
- ✓ To Identify how to extract text from an image;

✓ To Determine the appropriate deep learning approach that will be used to develop the model;
✓ To Develop a model that can synthesize Afaan Oromoo text into speech;
✓ To Evaluate the performance of the developed model using performance evaluation metrics;

1.5 Significance of the Study

The study "Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach" was significant in several ways. Firstly, it addressed a pressing need for reading text in the Afaan Oromoo language for visually impaired individuals. This was an important issue because it helped to increase their accessibility to information, improve their educational outcomes and overall quality of life. Additional benefit individuals with reading difficulties and to generate audio content from written text. Secondly, the study utilized a deep learning approach, which was a cutting-edge technology in the field of assistive technology.

Additionally, the study was also important for the Afaan Oromoo language and culture. Afaan Oromoo is one of the largest indigenous languages spoken in Ethiopia, but it has not received much attention in the field of assistive technology. The study also showed that technology can play a significant role in preserving language and culture by making it accessible to a wider audience, including visually impaired individuals. As such, the study made a valuable contribution to the promotion and preservation of the Afaan Oromoo language and culture.

Potential Applications of Study Findings:

The findings from our study have the potential to contribute significantly to the development of assistive technologies for visually impaired individuals, enhancing their ability to access and interact with written text. Moreover, the synthesized speech generated by our model can also benefit individuals with reading difficulties, such as dyslexia or cognitive impairments, by providing an alternative means to comprehend written content. Additionally, our model's capability to generate audio content from written text opens up possibilities for creating audiobooks, podcasts, voiceovers, and other forms of audio-based media. Furthermore, by enabling the conversion of written Afaan Oromoo text into synthesized speech, our study findings can contribute to the preservation and dissemination of social and cultural content, making it accessible and engaging for a wider audience.

1.6 Scope and Limitations of the Study

The scope and limitations of the study "Developing Afaan Oromoo Text Reading Model for Visually Impaired People using Deep Learning Approach" were as follows:

Scope:

- The study employed Optical Character Recognition (OCR) techniques to extract Afaan Oromoo text from an image.
- The model was developed to synthesize Afaan Oromoo text, which is extracted from an image containing Afaan Oromoo text.
- The study aimed to evaluate the efficiency of the model by extracting Afaan Oromoo text from the image and converting the text into speech output.

Limitations:

- The study focused on the extraction of Afaan Oromoo text from images using Optical Character Recognition (OCR) techniques, primarily for printed text. However, handwritten text recognition was not specifically addressed in this study.
- The study relied on a small sample of Afaan Oromoo text which is extracted from an image for the evaluation of the model's performance, which may not accurately reflect the model's performance on a larger corpus of text.
- The study also did not consider the potential impact of pronunciation variations, accents, or other language-specific issues that may affect the accuracy of the text-to-speech conversion.
- The research did not include the creation and integration of hardware components; it was only interested in the implementation of text extraction from an image and development of text to speech synthesizer models.

1.7 Thesis Organization

This research study is organized into six chapters to provide a structured framework for examining and presenting the findings. Chapter 1 serves as the introduction, offering an overview of the study's background, objectives, statement of problems, significance, and scope and limitations of the study. Chapter 2 presents a comprehensive literature review, summarizing existing knowledge and identifying gaps in the field of Afaan Oromoo text reading models. In Chapter 3, the research methodology is described, including details on data collection methods and preparation, text extraction from an image and speech synthesis. The proposed solution is outlined in Chapter 4, which includes the proposed solution and architecture, integrating Tesseract OCR, Tacotron2 model, vocoders and Transfer learning. Chapter 5 focuses on the implementation of Tesseract OCR and Afaan Oromoo text-to-speech synthesis model training. The evaluation and discussion of the model's results are presented in Chapter 6, analyzing results and highlighting strengths and limitations. Finally, we concluded the study by summarizing key findings, providing recommendations for further improvements, and discussing the broader implications and potential future research directions.

CHAPTER TWO

2. LITERATURE REVIEW

2.1 Overview

The development of a text-reading model for visually impaired individuals has been an area of research that has received significant attention in recent years(Ganesan et al., 2022; Swain & Sahoo, 2021). Many researchers have focused on converting written text into speech output to make written materials more accessible for visually impaired individuals. However, before the text-to-speech conversion could be performed, the text had to be extracted from the printed document through a process known as image-to-text conversion. This process involved using optical character recognition (OCR) techniques to extract the text from an image.

There have been numerous studies and works conducted in the past on image-to-text conversion and its application to assistive technology for visually impaired individuals. These studies explored various OCR techniques, including deep learning-based approaches, to achieve high accuracy in text extraction. Additionally, there have also been studies that integrated OCR and text-to-speech synthesis to create comprehensive text-reading models for visually impaired individuals. It was important to review the existing literature on image-to-text conversion and its application to assistive technology for visually impaired individuals to gain a deeper understanding of the field and the techniques that were developed. This information was valuable in guiding the development of the proposed text reading model for the Afaan Oromoo language, taking into account the unique challenges faced by visually impaired individuals.

2.2 Visual Impairment and Blindness

Visual impairment and blindness are conditions that can greatly impact an individual's quality of life. According to World Health Organization (WHO) report on October 2019, “Currently, approximately 2.2 billion individuals globally are affected by vision impairment, with around 1 billion of them experiencing preventable or unresolved vision impairments”(World Health Organization, 2019). Visual impairment can result from a variety of causes, including eye diseases, eye injuries, and congenital conditions. Blindness and visual impairment can lead to a range of difficulties, including difficulty with daily activities such as reading, navigating, and performing household tasks. In addition, visually impaired individuals may experience social

isolation, reduced employability, and limited access to information and education(Rokach et al., 2021).

A study was conducted to investigate the prevalence of visual impairment and identify related factors among school-aged children in Ethiopia. The study involved 718 students, resulting in a response rate of 89.3%. The results showed that 7.24% of the school-age children had visual impairment, with 3.9% being classified as having low vision and 3.34% having severe visual impairment. Some factors that were found to be associated with a visual impairment include: being female (with an adjusted odds ratio of 2.28), being between the ages of 10 and 13 years (AOR of 2.94) or 14 to 18 years (AOR of 4.06), being a student in a private school (AOR of 2.61), watching television for 2 to 4 hours per day (AOR of 3.56), watching television at a distance of less than 1 meter (AOR of 7.65), mobile exposure for 2 to 4 hours per day (AOR of 2.61), and medical visits while experiencing symptoms (AOR of 11.32). On the other hand, not having a medical visit experience was also found to be associated with visual impairment (AOR of 3.84) (Bezabih et al., 2017).

Another study conducted at Ambo Town aimed to determine the extent of visual impairment and its associated factors among primary school children. Out of the 838 students enrolled in the study, 93.07% participated, giving a total of 838 students. The results showed that 69 students, which is 8.8% (95%CI: 6.81-10.79) had visual impairment. The main cause of the impairment was found to be a refractive error in 46 (54.76%) students. The study identified several factors that were significantly associated with visual impairment among primary school children, such as grade level (AOR=2.375 95%CI: 1.079-5.226), exposure to mobile devices (AOR=2.44 95%CI: 1.401-5.01), awareness of child's eye problems (AOR=4.503 95%CI; 2.12-9.55), and a family history of visual impairment (AOR=2.41 95%CI: 1.071-5.42) (Legese et al., 2023).

2.3 Challenges of Visually Impaired People

Among the various educational strategies, inclusive education enables all students to learn in formal educational settings. This approach offers tailored teaching within suitable classes, addressing the educational requirements of typical students as well as those with special needs in an age-appropriate manner (Yihun & Belay, 2020). The results of the authors' research suggest that a small portion of the Ethiopian population takes advantage of government and non-

government services, as indicated by prior studies. Additionally, there is a relatively low provision of education for children with disabilities of school age.

In Ethiopia, a school has few opportunities and had a difficult time implementing inclusive education(Mulugeta & Mekuriaw, 2017). Lack of instructional resources and a skilled workforce are two issues that school faces in implementing inclusive education. Furthermore, the school lacks a systematic approach to conducting regular and scientifically-based assessments to identify the specific needs and extent of visual impairment among students. The library and resource room suffer from a significant scarcity of Braille materials, including handouts, modules, and reference books, along with other essential educational resources(Mulugeta & Mekuriaw, 2017). On another study, which aimed to assess students' "Braille Reading (BR) and Braille Writing (BW) skills and identify challenges impeding those skills," The study revealed that students were deficient in their understanding and support for Braille (BR) and raised-dot writing (BW) activities. Additionally, the special needs education (SNE) resource center was found to be disorganized, and learners lacked proficiency in using their fingers during Braille reading (BR). The findings indicated that students' Braille (BR) and Braille writing (BW) skills were significantly lacking, and their reading speed was notably slow. Furthermore, students demonstrated poor skills in both Afaan Oromo and English in relation to braille writing (BW)(Mitiku, 2020).

The study findings focused on exploring the experiences and difficulties encountered by students with visual impairments (SVI) in higher education institutions (HEIs), revealed several major challenges faced by SVI in HEIs. Firstly, the students frequently faced departmental assignments not aligned with their interests, and secondly, HEI curricula were inflexible and failed to accommodate the needs of SVI students. Access to accessible curricular materials was limited, along with a lack of utilization of assistive technology (Gebrehiwot, 2015). Additionally, assessment practices were deemed unfair towards SVI students, and the learning environment was not conducive to their needs. The support provided by institutions for SVI students were inadequate and disorganized, while instructors showed minimal effort in accommodating the learning needs of SVI students during classroom instruction. It should be noted that the findings showed no significant differences in perceptions among SVI of the two institutions studied(Gebrehiwot, 2015). Overall, the study highlights the need for HEIs to take a more proactive and inclusive approach toward supporting SVI in their academic pursuits.

(Omede, 2015)This study highlights the importance of addressing the specific educational needs of visually impaired students. The findings suggest that access to computer applications, optical aids, Braille writing materials, and physical facilities are essential to support their learning. Additionally, the availability of qualified personnel, funding, and library resources plays a crucial role in ensuring that visually impaired students receive a quality education. These findings underline the need for institutions to prioritize the support and resources needed to provide a fully inclusive education experience for visually impaired students.

According to a study conducted by the International Labour Organization (ILO) in Ethiopia, it was revealed that a significant proportion of individuals with disabilities, approximately 95%, live in poverty. These individuals heavily rely on family assistance and resort to begging as a means of livelihood. Another study conducted in the Oromia region found that 55% of the surveyed individuals with disabilities depend on their families, neighbors, and friends for their sustenance. These findings highlight the challenging socio-economic conditions faced by people with disabilities in Ethiopia, emphasizing the need for inclusive policies and support systems to improve their quality of life.[1]

Summary of Visual Impairments and Challenges

The study found that the prevalence of visual impairment among primary school children in Ambo Town was 8.8%, with refractive error being the main cause. High school grade level, mobile exposure, family history, and lack of awareness about eye problems were significant factors associated with visual impairment. To address this issue, regular monitoring and screening of schoolchildren is crucial, as well as increasing community awareness and access to eye health services (Legese et al., 2023). In a separate study, the result showed that 7.24% of school-age children in Ethiopia had visual impairment, with low vision being more prevalent. The study identified female gender, age (10-13 years and 14-18 years), being a private school student, television exposure, mobile exposure, and lack of medical visits as associated factors with visual impairment. To reduce the prevalence of visual impairment, community awareness and access to affordable eye health services are important (Bezabih et al., 2017).

The studies reviewed suggest that inclusive education has the potential to benefit all students, including those with special needs. However, the implementation of inclusive education in

[1] International Labour Organization, https://www.ilo.org/wcmsp5/groups/public/@ed_emp/@ifp_skills/documents/publication/wcms_112299.pdf

Ethiopia faces several challenges, including a shortage of instructional resources, a lack of qualified professionals, and a lack of professional growth and development opportunities. The study results indicate that students with visual impairments face particular challenges in accessing education, including limited access to curricular materials in accessible formats and limited use of assistive technology. Additionally, the study results show that the assessment practices in higher education institutions are often unfair to visually impaired students and the learning environment is not friendly to them. The support received by visually impaired students from their institutions was found to be inadequate and disorganized, and instructors made little effort to accommodate their learning needs. Another finding highlight the need for higher education institutions to prioritize the support and resources needed to provide a fully inclusive education experience for visually impaired students.

2.4 Artificial Neural Networks

Artificial Neural Networks (ANNs) mimic the structure and functioning of the human brain and are a type of artificial intelligence. They comprise interconnected nodes known as artificial neurons, which collaborate to process and analyze data. ANNs can be used to solve complex problems and perform tasks that were previously only possible through human efforts, such as recognizing patterns and making predictions(Gourav, 2021).

One of the key applications of ANNs is in the field of Optical Character Recognition (OCR) and Text-to-Speech (TTS) systems(Pancholi & Patel, 2021). In OCR systems, ANNs are used to recognize and transcribe text from images and scanned documents. The system analyzes the image, breaking it down into individual characters, and then uses ANNs to identify and classify each character based on patterns and shapes. This allows the system to accurately transcribe text from any image or document, even if the text is distorted or written in a different font. Similarly, TTS systems use ANNs to convert text into spoken words. The system first processes the text input to understand its structure, such as the words, sentences, and prosody (the rhythm and melody of speech)(Alam et al., 2020; Tan et al., 2021). The processed text is then used to train an ANN, which generates an audio output that mimics the natural rhythm and tone of human speech. This allows the TTS system to produce a synthetic voice that sounds similar to a human voice, making it easier for visually impaired individuals to understand written text.

2.5 Convolutional Neural Network

A Convolutional Neural Network (CNN) is a specialized type of Artificial Neural Network primarily used for analyzing grid-structured data, such as images or audio. CNNs are specifically designed to process and extract meaningful features from these types of data. It is used in computer vision and natural language processing tasks and works by passing a filter over the data, called convolution, and producing a feature map that can be used to classify or recognize patterns in the data(O'Shea & Nash, 2015). A convolutional neural network (CNN or ConvNet) is one of the earliest models of hierarchy, known for its ability to learn intricate patterns in images across multiple layers using a series of two-dimensional convolutional operations. CNNs are specifically created to handle multi-dimensional data that is organized as several arrays or tensors(Alam et al., 2020).

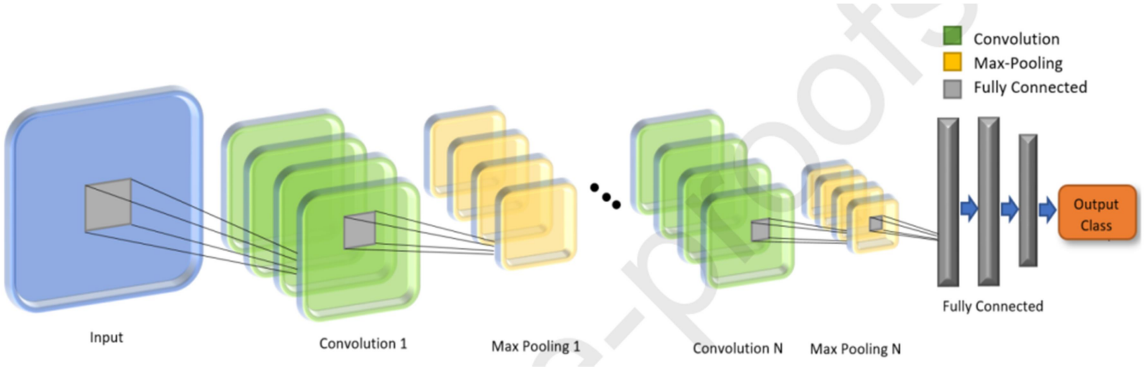


Figure 1: Architecture of Convolutional Neural Network(Alam et al., 2020)

The above architecture of CNN shows that the input layer receives the raw input data, followed by a series of convolutional layers and max pooling layers that extract and down-sample features. This process is repeated in multiple blocks to learn more complex features. The fully connected layers then process these features and perform the final classification based on the learned representations. The output layer produces the probabilities for each class, indicating the predicted class for the input.

For Optical Character Recognition (OCR) systems, CNNs are often used to identify and extract text from an image. The network is trained on large datasets of images with text to learn the patterns of text and how to recognize it. The CNN processes the input image and uses pooling layers to reduce the dimensionality and simplify the feature map. Finally, the feature map is fed into a fully connected layer that uses the learned patterns to predict the text in the image(Alkaddo

& Albaqal, 2022). For Text-to-Speech (TTS) systems, CNNs can also be used to generate speech from text. The network is trained on large datasets of text and speech pairs to learn how to generate speech from text. The input text is processed through an encoder network that converts the text into a compact representation, which is then processed by a decoder network to generate an audio signal. The CNN approach is proven to be effective in synthesizing speech with high quality and naturalness, making it an attractive choice for TTS systems(Ping et al., 2017).

2.6 Recurrent Neural Network

A Recurrent Neural Network (RNN) is a type of artificial neural network that is designed to process sequential data. It is a network of artificial neurons that are designed to recognize patterns in sequences of data, such as time series data, speech, text, or any other type of sequential data(Aishwarya.27, 2023).

RNNs leverage sequential information, unlike traditional neural networks that assume inputs and outputs are independent. They consider prior information to enhance their processing capabilities. They are referred to as recurrent because they repeat the same computation for every element in a sequence and the output depends on previous computations. RNNs can be viewed as having a memory that keeps track of what has been computed thus far. Although, in theory, RNNs are capable of utilizing information from very long sequences, in reality, they are restricted to only considering a few previous steps(Britz A., 2017). A typical RNN can be depicted as follows.

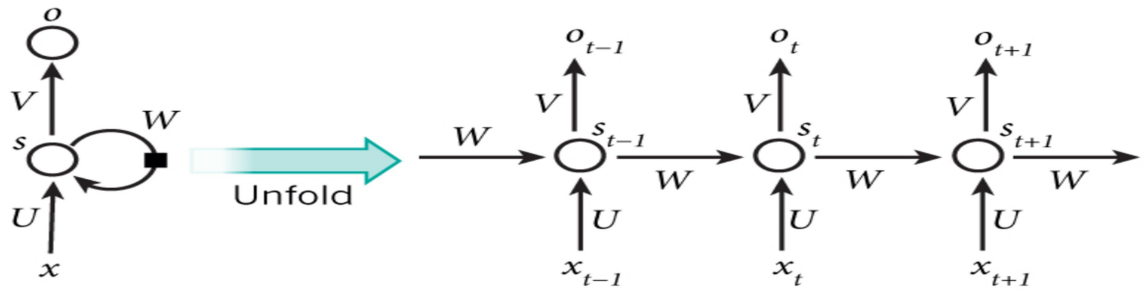


Figure 2: A recurrent neural network and the unfolding in time of the computation involved in its forward computation(Liu et al., 2015)

Figure 2 above demonstrates how a feed-forward network can be created by unfolding an RNN over time. xt is the input at time step t, such as a one-hot vector encoding a word in a sentence, and st is the hidden state at time step t. This is also known as the network's memory, and it is

dependent on the input and previous concealed state at time step t. where ot represents the output at time step t in the equation ot=softmax(Vst)(Rajaram et al., 2018). An important characteristic of RNNs is their hidden state, serving as a memory to retain information from previous time steps and use it for future predictions. The hidden state is updated at each time step using the input and the previous hidden state. This allows the network to maintain context and capture patterns in the sequential data over time(DiPietro & Hager, 2020). RNNs have been widely used in a variety of applications, including natural language processing, speech recognition, and optical character recognition (Iryna, 2021). In OCR, RNNs can be used to recognize text in an image by processing the image pixel by pixel, while maintaining the context of the surrounding pixels. This allows the network to recognize patterns in the image and make predictions about the characters that are present.

Similarly, RNNs have been used in Text-to-Speech (TTS) systems to generate speech from text. In TTS, the input is a sequence of characters or words, and the network generates speech by predicting the spectral parameters of the speech waveform at each time step(Karita et al., 2019). The hidden state of the network allows it to capture the context of the words and the relationships between them, which is crucial in generating natural-sounding speech.

2.7 Image-to-Text Conversion

Image-to-text conversion using OCR (Optical Character Recognition) refers to the process of converting images of text into editable text documents. Tesseract OCR is an open-source OCR engine that has become one of the most popular OCR technologies for image-to-text conversion. The system works by first preprocessing the image to make it more suitable for text recognition. This involves processes such as noise reduction, removing any noise or blur, and enhancing the image for better text recognition(Patel et al., 2012; Smith, 2005). Once the image has been preprocessed, OCR Tesseract uses its deep learning algorithms to recognize the text in the image. It divides the image into small segments and identifies each segment as either text or non-text. The recognized text is then processed to remove any errors or inaccuracies. The final output is a text document that can be edited and saved for further use. Tesseract OCR is highly accurate and supports multiple languages, making it an ideal solution for image-to-text conversion. Additionally, it is constantly evolving and improving, making it a reliable technology for image-to-text conversion in various applications.

A recent research paper introduces an Android app that combines the Tesseract OCR engine, the Bing translator, and the phone's native speech output feature. The application has been tested with users from different language backgrounds, and the results indicate that it has a wide range of applications, including travel and accessibility for visually impaired individuals. With this application, travelers visiting foreign countries can understand messages in different languages, and visually impaired users can access important messages from printed texts through the speech output feature(Ramiah et al., 2015).

In the field of OCR, digital cameras have emerged as a popular choice for image acquisition due to their speed, versatility, mobility, non-intrusive nature, and affordability. However, their use in OCR applications is limited by certain factors such as geometrical distortions. This study focuses on the preprocessing step involved in OCR, specifically with images captured using digital cameras. The effectiveness of the proposed preprocessing technique was evaluated through experiments using FineReader 7.0 as the OCR tool. The results of the experiments demonstrated the significance of image preprocessing in enhancing the performance of OCR applications(Bieniecki et al., 2007). The goal of the study was to evaluate the impact of different factors on the accuracy of OCR recognition using FineReader 7.0 software. The factors considered were uneven lighting, noise, resolution, and geometric deformations.

The results showed that noise was the most detrimental factor but with 10% noise and a high resolution of 600 dpi, the accuracy was still acceptable. The median filter was found to be inappropriate for this scenario and resulted in further accuracy loss. The accuracy was most sensitive to geometric deformations and a rotation of even 10 degrees could result in total OCR processing failure. However, the results also showed that upsampling a low-resolution image could result in significant accuracy improvement. The conclusion drawn from these experiments was the importance of image preprocessing in OCR applications and the sensitivity of OCR recognition to geometric deformations(Bieniecki et al., 2007).

(Pawar et al., 2018)In this paper, the utilization of Optical Character Recognition (OCR) and the Tesseract OCR Engine is studied to extract textual information from images. The authors aim to explore the viability of this technology for converting images into editable or digitally stored data. Additionally, the authors also examine the ability of OCR to convert the recognized text into audio format for the benefit of visually impaired individuals. Experiments and studies were

conducted to evaluate the performance of the Tesseract OCR Engine in recognizing text from images and converting it into audio. The results indicate that the system is capable of accurately extracting textual information from images and converting it into audio format. The authors conclude that this system provides a valuable solution for digitizing text from images and making it accessible to visually impaired individuals. The study highlights the importance of OCR technology in modern society and the role it plays in bridging the gap between image-based information and digital data(Pawar et al., 2018).

The authors conducted an investigation titled "Optical Character Recognition by Open Source OCR Tool Tesseract," where they offer a thorough examination of Optical Character Recognition (OCR) and its practical application utilizing the open-source OCR tool, Tesseract. The paper begins by introducing the concept of OCR and its importance in the digitalization of text data. The authors then delve into the history of Tesseract, an open-source OCR tool that has been widely adopted by the OCR community. The architecture of Tesseract is also discussed in detail, including its experiment results on various types of images(Patel et al., 2012). The authors conclude the paper by performing a comparative study between Tesseract and a commercial OCR tool, Transym OCR. The study focuses on the extraction of vehicle number plates and compares the two tools based on various parameters, including accuracy and efficiency. The results of this comparative study provide valuable insights into the strengths and weaknesses of both Tesseract and Transym OCR, helping users to make informed decisions about which tool to use for specific OCR tasks. Overall, the paper provides a comprehensive review of OCR and its implementation using Tesseract, making it a valuable resource for researchers and practitioners in the field(Patel et al., 2012).

Another study conducted by(V. S & A, 2015), the focus is on the comparison of various Optical Character Recognition (OCR) tools in terms of accuracy and error rate. OCR technology is used to convert scanned images into editable text format, which is a useful and popular method in different applications. The effectiveness of OCR is dependent on the pre-processing and segmentation algorithms used, as well as the quality of the input image(V. S & A, 2015). The primary objective of the study was to determine the accuracy and error rate of various OCR tools. To measure these parameters, the Character Accuracy (CA) and Character Error Rate (CER) were calculated using a specific formula. The formula takes into consideration the number of characters in the input image and the number of characters in the resultant text document.

Table 1: Tesseract and Transym OCR Comparison(Patel et al., 2012)

Feature	Tesseract OCR	Transym OCR
Free	Yes	No (Trial Version is available)
Open Source	Yes	No
License	Apache	Proprietary
Online	No	No
Operating System	Window, MAC, Linux	Windows
Latest Stable version	3.01	3
Release Year	2010	2008
DLL Available	Yes	No
Accuracy (For extracting character from vehicle number plate)	61% (color images) 70% (gray scale images)	47%
Average Time	1 second (color images) 0.82 Seconds gray scale images)	6.75 seconds
σAC	34.21(For color Images) 24.64(For Gray Scale Images)	40
σT	0.63 (For color images) 0.61(For gray scale images)	

σAC = Standard deviation of Accuracy

σT = Standard deviation of Time taken to perform OCR

2.8 Deep Learning Approaches for Image-to-Text Conversion

Deep learning is a field of artificial intelligence that has recently been applied to the problem of image-to-text conversion. This involves using neural networks to analyze images and generate corresponding text(Iryna, 2021). The advantage of using deep learning is that it can handle more complex images, recognize patterns in images more accurately, and produce more human-like text. As a result, deep learning approaches have proven to be very effective for image-to-text conversion and have become increasingly popular in recent years. This field is still rapidly evolving, and there is a lot of room for growth and innovation. Nevertheless, deep learning approaches are already delivering remarkable results and have the potential to revolutionize the field of image-to-text conversion(He & Deng, 2017). Some of the popular Deep Learning approaches used in Image to Text Conversion are:

2.8.1 Convolutional Neural Networks (CNN)

CNNs are deep neural networks used for image recognition tasks. They are trained to identify patterns and features in an image and extract information from it. They can be used to extract text information from an image by training them on a large dataset of text images and recognizing the features that are common to all the images(He & Deng, 2017).

2.8.2 Recurrent Neural Networks (RNN)

RNNs are another type of deep neural network that is used for image-to-text conversion. They are trained to recognize sequences of characters in an image and extract the text information from it. The network uses a hidden state to keep track of the previous characters in the sequence and can identify patterns in the image that are common to all the characters(Abhishek, 2020).

2.9 Basic Architecture of Optical Character Recognition System

The architecture of an Optical Character Recognition (OCR) system typically consists of several stages, each of which is responsible for a different aspect of the text recognition process. The main stages in an OCR system (A. S. & G, 2015) include:

Pre-processing: This stage is responsible for preparing the input image for text recognition. The pre-processing stage typically involves image normalization, color correction, noise reduction, and the segmentation of the image into individual text regions.

Feature extraction: In this stage, the OCR system extracts relevant features from the pre-processed image, such as the shapes, edges, and contours of the characters. These features are

used to identify the different characters in the image and to determine the relationships between them.

Character recognition: This stage is responsible for recognizing the individual characters in the image. The OCR system uses the features extracted in the previous stage to match the characters with an artificial neural network trained to recognize characters.

Text recognition: In this stage, the OCR system uses the recognized characters to form words and sentences, and to produce the final text output. The text recognition stage may also involve the correction of errors, such as missing characters or misspelled words, and the improvement of the overall text quality. The final stage of an OCR system is responsible for post-processing the output text, such as converting it into a specific format, or saving it to a separate file. the basic stages described above are typically present in most OCR systems(A. S. & G, 2015).

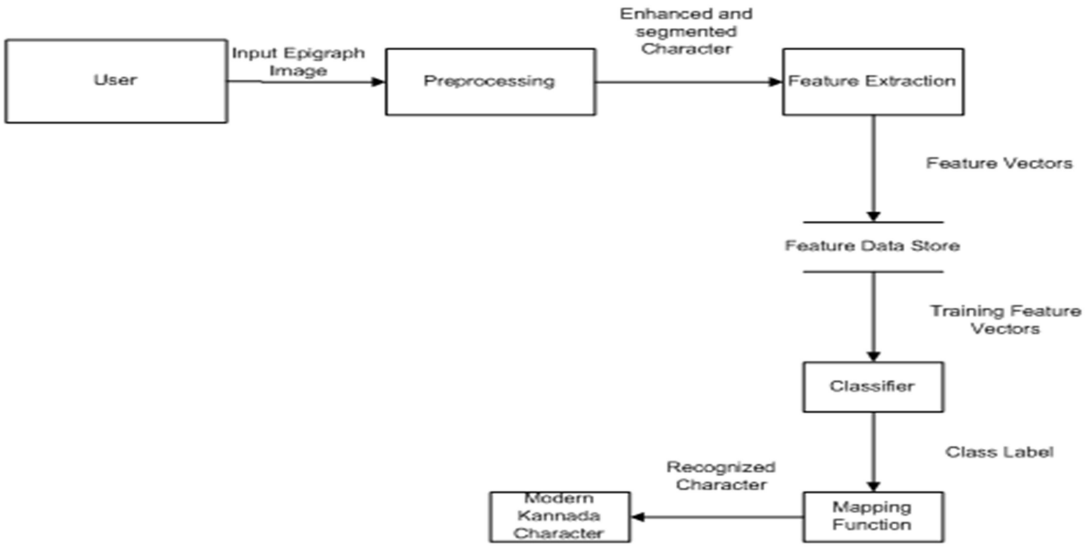


Figure 3: Architecture of OCR System (A. S. & G, 2015)

The figure above show that systematic flow of OCR system, starting with preprocessing to enhance image quality, followed by feature extraction to capture important characteristics, classification using an ANN to identify characters, and finally post-processing to present the results in a modern and accessible format.

2.10 Text-to-Speech Synthesis

Text-to-Speech (TTS) synthesis is a process of generating spoken language from written text inputs. TTS systems are widely used in various applications such as voice-enabled devices, speech-based navigation systems, and screen readers for visually impaired individuals. The basic

process of TTS synthesis involves several steps, including text processing, acoustic modeling, and speech synthesis(Eric et al., 2023).

Text processing is the first step, where the text input is preprocessed and converted into a machine-readable format. This includes tasks such as text cleaning, normalization, and tokenization. The processed text is then used to generate the acoustic features, which are statistical models that map text to speech parameters such as pitch, duration, and energy. The final step is speech synthesis, where the acoustic features are used to generate speech signals that are then played back to the user as speech(Reichel & Ptzinger, 2006).

In the paper "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions", the authors aimed to present a state-of-the-art TTS system called Tacotron 2(Shen et al., 2018). The objective was to improve upon previous TTS systems, such as Tacotron and WaveNet, by incorporating new advancements and generating natural-sounding speech from text using neural networks. The authors explained that the system maps a sequence of letters to a sequence of audio features using a sequence-to-sequence model optimized for TTS. These audio features represented as an 80-dimensional spectrogram, capture pronunciation, volume, speed, and intonation and are computed every 12.5 milliseconds. The spectrogram is then converted to a 24 kHz waveform using a WaveNet Vocoder. The paper presents Tacotron 2 as a major advancement in TTS research. The authors showed that their system could generate natural-sounding speech from text using neural networks trained with only speech examples and text transcripts. Despite its achievements, the authors acknowledged that there are still limitations and opportunities for further improvement such as improving pronunciation and real-time generation, as well as incorporating the ability to control the generated speech.

The authors of the paper “Towards End-to-End Prosody Transfer for Expressive Speech Synthesis with Tacotron” aimed to improve upon the existing Tacotron architecture by incorporating prosody transfer capabilities. To achieve this, they introduced a prosody encoder, which is an additional component in the Tacotron architecture. The prosody encoder computes a low-dimensional embedding from a clip of human speech that captures the prosody, or the rhythm and melody of the speech(Skerry-Ryan et al., 2018). During inference, the prosody embedding is used to transfer the prosody of the reference speech to the generated speech, even if the voice of the reference speaker is not present in the training data of Tacotron. The authors

demonstrated that the prosody transfer was successful and that the generated speech exhibited the prosody of the reference speech, even for speakers whose voice was not in the Tacotron training data(Skerry-Ryan et al., 2018).

The authors of the paper, "Style Tokens: Unsupervised Style Modeling, Control, and Transfer in End-to-End Speech Synthesis"(Y. Wang et al., 2018), introduced a novel approach to modeling the variations in speech styles. They proposed the use of Global Style Tokens (GSTs) to represent the prosody of a speech clip as a linear combination of basis embeddings. This approach allows for the unsupervised learning of text-independent variations in a speaker's style without the need for explicit style labels. During inference, the combination weights for the GSTs can be selected or modified, giving the ability to control and transfer the speech style without relying on reference audio clips.

The authors demonstrated the effectiveness of their method through a series of experiments that showed that the GSTs can effectively capture variations in speaking styles, including accent, speed, intonation, and more. They also showed that the model can generate speech in the style of different speakers and control the style of the generated speech by modifying the combination weights for the GSTs(Y. Wang et al., 2018). Overall, the paper presents a valuable contribution to the field of speech synthesis by introducing a new and effective approach to unsupervised style modeling and transfer in end-to-end speech synthesis systems.

(Sun et al., 2020)The paper describes a new approach to model and control prosody attributes in speech synthesis. The authors add a hierarchical latent representation of prosody attributes to the Tacotron 2 framework, which enables them to control and interpret the prosody of the generated speech at each level. The conditioning between levels and across dimensions is imposed through auto-regressive factorization. Experiments are conducted to evaluate the effectiveness of the proposed method, with results showing improved reconstruction performance, qualitative interpretation, and disentanglement of nominated prosody attributes. The experiments also demonstrate the effectiveness of phone and word-level prosody control and sampling, including energy, duration, and F_0 control and sampling, as well as silence control. The results are presented for three speakers (one male and two female) and show the effectiveness of the proposed approach in the Tacotron 2 framework.

In the "Generating Natural and Diverse Text-to-Speech Samples Using A Quantized Fine-grained VAE and Auto-regressive Prosody Prior"(Vae et al., 2020) paper, an approach was proposed for generating natural and diverse audio samples for TTS systems like Tacotron-2. The approach uses vector quantization on a fine-grained VAE to extract prosody features at the phoneme level and uses an auto-regressive before addressing prosody discontinuity. The approach was evaluated on five different models using three different speakers (male and female) and different text samples and scales (1.0, 0.2, and 0.0). The paper presents the audio files and results of the experiments, which show the effectiveness of the proposed approach in generating natural and diverse TTS samples.

2.10.1 Deep Learning Approaches for Text-to-Speech Synthesis

Text-to-Speech (TTS) synthesis is the process of generating speech from text. Deep learning approaches for TTS synthesis use neural networks to convert text input into speech output. The main architecture for TTS synthesis using deep learning consists of two main components:

There are two main deep-learning approaches for TTS synthesis:

1. **Sequence-to-Sequence** (Seq2Seq) based TTS: This approach uses a Seq2Seq model, which consists of an encoder-decoder network, to generate speech from text. The encoder network processes the input text to generate a compact representation, while the decoder network takes this representation and generates a speech signal(Tomás, 2020).
2. **Autoregressive TTS**: This approach is based on auto-regression, where the decoder network generates speech samples one-time step at a time. It uses a recurrent neural network (RNN) to model the sequential dependencies in speech(Tomás, 2020).

Both Seq2Seq and Autoregressive TTS approaches have been used in various TTS systems, such as Tacotron, Tacotron2, and Deep Voice. They have shown good results in terms of the naturalness and fluency of the generated speech. There are several deep learning approaches for text-to-speech synthesis(Kumar et al., 2023), including:

1. **Tacotron**: It is a sequence-to-sequence model with an attention mechanism that converts text into a spectrogram and then waveform.
2. **Tacotron 2**: It is an improvement over Tacotron, which uses a modified architecture and a multi-speaker model to improve the quality of the generated speech.

3. **Deep Voice**: It is a multi-stage generative model that uses a sequence of models to predict various aspects of speech, including phone duration, fundamental frequency, and Mel-spectrogram.
4. **ClariNet:** It is an autoregressive generative model that uses a variant of the WaveNet architecture to predict speech waveform.
5. **FastSpeech:** It is a recent model that uses a feed-forward network architecture to generate high-quality speech in real-time.

These are some of the popular deep-learning approaches for text-to-speech synthesis. The architecture of these models can vary greatly, but most models use a combination of neural networks and attention mechanisms to capture the complex relationships between the text and the audio features.

2.10.2 Basic Architecture of Text-to-Speech System

1. Encoder

The encoder network in text-to-speech (TTS) synthesis using deep learning is responsible for converting the input text into a compact, fixed-dimensional representation. This representation, also known as the "embedding" or "encoding," captures the important semantic and linguistic information contained in the text, such as the meaning of words, their grammatical relationships, and other contextual features(Ajitesh, 2019; Yasuda et al., 2019).

The encoder network typically consists of several layers of artificial neurons, each of which is trained to extract and abstract features from the input text. The first layer typically takes as input the one-hot encoding of the input text, which represents the text as a series of binary vectors, one for each word or character. This input is then transformed through a series of dense layers, activation functions, and attention mechanisms to produce the final encoding(Yasuda et al., 2019).

2. Decoder

The decoder network in TTS synthesis using deep learning is responsible for converting the compact representation of text generated by the encoder network into an audio signal. It typically consists of one or more fully-connected layers, recurrent neural network (RNN) layers, or a combination of both. The decoder network takes the compact representation of the text as input and uses it to generate a sequence of parameters that represent the audio signal(Ajitesh, 2019).

These parameters are then transformed into a waveform representation of the audio signal using a process called synthesis, which can be done using a vocoder.

The decoder network is typically trained using a dataset of audio samples and their corresponding text inputs. During training, the decoder network is updated to minimize the difference between the generated audio signal and the target audio signal, which is obtained by transforming the ground-truth text input into audio using a process called text-to-spectrogram(Yasuda et al., 2019). The output of the decoder network is a sequence of audio parameters, such as Mel-spectrogram coefficients or audio samples that represent the target audio signal. These parameters can be fed into a vocoder to produce the final waveform representation of the audio signal(Shen et al., 2018). The choice of vocoder depends on the specific TTS system, but common options include the WaveNet vocoder and the Griffin-Lim algorithm.

3. Attention Mechanisms

In deep learning, the attention mechanism is used to help the model focus on the most relevant parts of the input when making a prediction. It is based on the concept of directing your focus and paying greater attention to certain factors when processing the data(Brauwers & Frasincar, 2021). The attention mechanism is one component of a network's architecture and is in charge of managing and quantifying the interdependence. The attention mechanism, like other neural network-based methods, tries to mimic the human brain/vision to process data. Human vision does not process the entire image at once; however, it only focuses on specific parts(Valente et al., 2021).

The attention mechanism is a mechanism that was developed to improve the performance of the Encoder-Decoder RNN on machine translation(Jason, 2017).

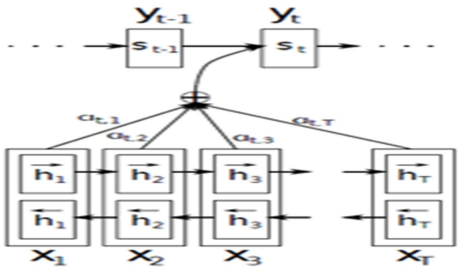


Figure 4: Example of the Attention model(Bahdanau et al., 2014)

All the vectors h1,h2.., etc., used in their work are the concatenation of forward and backward hidden states in the encoder.

$$h_j = \left[\overrightarrow{h}_j^\top ; \overleftarrow{h}_j^\top \right]^\top$$

To put it in simple terms, all the vectors h1, h2, h3…., and hTx are representations of the Tx number of words in the input sentence. In the simple encoder and decoder model, only the last state of the encoder LSTM was used (hTx in this case) as the context vector. The weights are also learned by a feed-forward neural network and which mentioned their mathematical equation below.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j .$$

The context vector ci for the output word yi is generated using the weighted sum of the annotations. The weights αij are computed by a softmax function given by the following equation:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \qquad e_{ij} = a(s_{i-1}, h_j)$$

eij is the output score of a feed-forward neural network described by the function **a** that attempts to capture the alignment between input at **j** and output at **i**. If the encoder produces **Tx** many "annotations" (the hidden state vectors) each having dimension **d,** then the input dimension of the feedforward network is **(Tx, 2d)** (assuming the previous state of the decoder also has **d** dimensions and these two vectors are concatenated)**.** This input is multiplied with a matrix **Wa** of **(2d, 1)** dimensions (of course followed by the addition of the bias term) to get scores **eij (having a dimension (Tx, 1))**. On the top of these **eij** scores, a tan hyperbolic function is applied followed by a softmax to get the normalized alignment scores for output j:

E = I [Tx*2d] * Wa [2d * 1] + B[Tx*1]

α = softmax(tanh(E))

C= IT * α

So, **α** is a (Tx, 1) dimensional vector and its elements are the weights corresponding to each word in the input sentence(Hore & Chatterjee, 2022).

2.11 Long Short-Term Memory (LSTM)

LSTM was Introduced by (Hochreiter & Schmidhuber, 1997), which increased the standard RNN's capacity for remembering by putting a "gate" into the cell, to address the "long-term reliance" problem. LSTM is available in three distinct configurations: LSTM without a forget gate, LSTM with a forget gate and LSTM with a peephole connection.

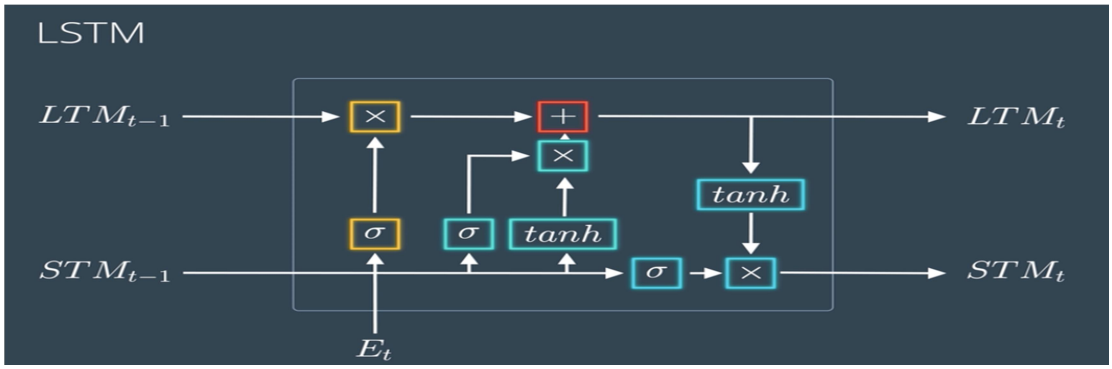


Figure 5: Architecture of LSTM(Singh, 2021)

The above figure demonstrates the architecture of LSTM. The Learn Gate keeps just the knowledge that is necessary for prediction and uses Event (Et) and Prior Short-Term Memory (STMt-1) as input. The Forget Gate determines which knowledge should be maintained and which to forget using input from the Prior Long Term Memory (LTMt-1), The Remember Gate produces output by combining Prior Short Term Memory (STMt-1) and Present Event (Et) and The Usage Gate Consolidate significant data from Prior Long Term Memory and Prior Short Term Memory to construct STM for the following cell and provide output for the current event(Singh, 2021).

Gated Recurrent Unit

Long Short-Term Memory (LSTM) networks were retained by (Cho et al. 2014), recurrent unit (GRU), a form of recurrent neural network (RNN), in 2014. GRU can analyze sequential data, including text, voice, and time-series data, just as LSTM. The fundamental principle of GRU is to selectively update the network's hidden state at each time step via gating techniques. Information flow into and out of the network is managed by the gating mechanisms. The reset gate and the update gate are two of the GRU's two gating systems. The update gate decides how much of the new input should be utilized to update the hidden state, while the reset gate defines

how much of the prior hidden state should be erased. The updated hidden state is used to calculate the GRU's output.

Bidirectional LSTMs

A Bidirectional LSTM, or biLSTM, is a sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backward direction. BiLSTMs effectively increase the amount of information available to the network, improving the context available to the algorithm(Joseph, 2022) (e.g. knowing what words immediately follow and precede a word in a sentence).

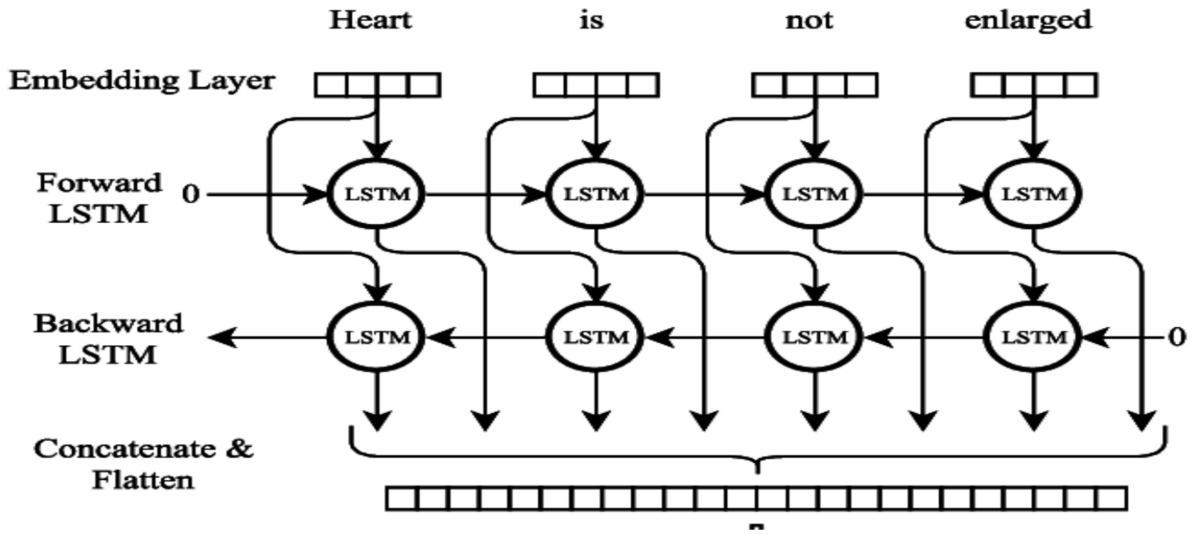


Figure 6: Example of knowing what words immediately follow and precede a word in a sentence(Cornegruta et al., 2016)

Making any neural network has the sequence information in both directions—backward (future to past) or forward—is known as bidirectional long-short term memory (bi-lstm) (past to future). A bidirectional LSTM differs from a conventional LSTM in that our input flows in two ways. We may make input flow in one way, either backward or forward, using the standard LSTM. To maintain both past and future information, bi-directional input may be made to flow in both ways(Yugesh, 2021).

2.12 Afaan Oromoo Language

Afaan Oromoo is a Cushitic language spoken in Ethiopia and is part of the larger Afro-Asiatic language family. It is one of the largest languages in Ethiopia, with over 25 million speakers(Tilahun, 1994). Afaan Oromoo has a complex phonological structure with many

phonemes and unique sounds. A five-vowel system, where the letters A, E, I O, and U stand in for Afaan Oromo vowels, is one of the main characteristics of Afaan Oromo phonology. The use of each vowel is essentially the same throughout all Afaan Oromo dialects. The language uses contrastive or phonemic vowel length. These specific letters are repeated in writing to indicate vowel length: deemuu (go), and nyaadhu (eat). A large number of consonant phonemes are present, along with many glottal and implosive consonants. In addition, Afaan Oromoo has a rich tonal system, which is used to distinguish words with different meanings.

The following rule applies to Afaan Oromo syllabic structures(Yohannes, 2011):

- A word in Afaan Oromo cannot begin with two or more distinct consonants, and the same constraint applies to the beginning of a syllable. Such as "meal," irbaata (i.rbaa.ta).
- Syllables in Afaan Oromo cannot conclude with two or more distinct consonants. for instance, ilma (ilm.a)'son'
- Syllables in Afaan Oromo cannot begin with two or more consonants that sound the same. For instance, tokkummaa (to.kku.mmaa) means "unity."
- Syllables in Afaan Oromo cannot have two or more consonants that sound the same. such as the'smart' haxxee (haxx.ee)
- In the Afaan Oromo structure, two or more vowels of the same kind cannot coexist.

The morphological structure of Afaan Oromoo is also complex and diverse, with a rich system of inflection and derivation. Afaan Oromoo has a rich set of noun and verb forms, which are used to express a wide range of meanings. Nouns in Afaan Oromoo are inflected for case, number, and gender, while verbs are inflected for tense, aspect, and mood. In addition, Afaan Oromoo has a rich system of derivatives, which are used to create new words by adding prefixes, suffixes, and infixes to existing words(Rabbiirra, 2008). These derivatives are used to express a wide range of meanings, including cause, manner, instrument, and purpose, among others.

2.12.1 Language Structure

English and Afaan Oromo have different sentence structures. Subject-object-verb (SOV) language is used by Afaan Oromo. The language that uses the subject, object, and verb in that specific order is known as SOV. A sentence construction known as subject-verb-object(SVO) places the subject first, the verb second, and the object third. For instance, "Mooneeraan bilisa bahe" in Afaan Oromo, The words "Mooneeraa," "bilisa," and "bahe" all refer to different parts

of speech. It possesses an SOV structure as a result. The English translation of the SVO statement, "Mooneeraa has acquired freedom," is "Mooneeraa has got freedom." The way that adjectives are formed in Afaan Oromo and English also differs. Adjectives are often near to the nouns they modify in Afan Oromo, whereas in English they are typically placed before the noun. For instance, namicha gaarii (good guy), followed by gaarii (adj.), (noun) (Tesema & Tamirat, 2017).

2.13 Related Works

Numerous research studies have been undertaken to create a system that is capable of interpreting and translating written text in languages commonly used in developed countries such as English. In their study, A. Karthik and his friends aimed to create a solution for visually impaired individuals. They developed a device, known as Voice Assistance, which was designed to help these individuals to read various messages. The authors highlight the importance of accessible technology for visually impaired people and their efforts to create a system that could make a difference in their daily lives(Karthik et al., 2018). In 2020, Bhardwaj and Sharma made significant advancements in the field of assistive technology for the visually impaired. They created a model for an Image Text Reader, which would enable visually impaired individuals to read the text in their desired language with ease. The system, powered by a Raspberry Pi, is designed to be user-friendly and highly accessible, making it a valuable tool for those with visual impairments(Bhardwaj & Sharma, 2020).

The researchers from Prince Mohammed Bin Fahd University in Saudi Arabia focused on developing a new type of smart glass to help visually impaired students in their education. They designed a device that can capture and recognize text from images, then convert it to speech. The device is powered by a raspberry pi 3 B+ microcontroller and uses a combination of OpenCV software, tesseract OCR tools, and the Efficient and Accurate Scene Text Detector (EAST), which is based on deep learning, to detect text in images. Finally, the identified text is processed using Google's text-to-speech API to convert it into spoken words(Alsaid et al., 2019).

Esra Ali and Tong Boon created a system to aid visually impaired students, utilizing smart glasses equipped with text recognition technology(Hassan & Tang, 2016). The device aims to help users read from hardcopy materials. Another study conducted by Pandya & Goradiya in 2021 aimed to develop audio-assisted electronic glasses for the blind and visually impaired to

read documents. The glasses can distinguish between different bank currencies, text, and human objects, utilizing an OCR model combined with a deep neural network method, such as a Convolutional Neural Network (CNN)(Pandya & C.Goradiya, 2021).

The study aimed to create a multi-functional smart glass to assist visually impaired individuals. The smart glass was designed with two modes: the seeing mode and the reading mode. In the seeing mode, the device was able to recognize 550 different types of objects and their positions (left, right, or center). In the reading mode, the device was able to recognize text and then read it out loud to the user. Additionally, the device was also able to create a copy of the braille script for the text that was recognized. The authors of the paper, Sudharshan, Sowdeshwar, and Jagannath, conducted the study to help visually impaired people with their daily tasks(Sudharshan et al., 2022).

(Mathur et al., 2019)The paper presents an AI-based reading system designed to aid visually impaired individuals. The system employs OCR (Optical Character Recognition) technology and a camera application on smartphones to scan and convert images of written text into machine-encoded text. The generated output is delivered in speech format through a Text-To-Speech module, thus allowing the visually impaired to access information present in written documents. This system offers a convenient and accessible solution for individuals with visual impairments, demonstrating the potential of AI in improving their daily lives.

The reviewed paper presents a solution for visually impaired people to increase their confidence while walking. The authors propose the use of a Smart Walking Stick equipped with ultrasonic sensors to detect obstacles and pits in the environment. The stick is programmed to connect to an Android phone for auditory feedback and to determine the best route using GPS technology. The use of ultrasonic sensors and GPS integration has the potential to reduce accidents and improve mobility for visually impaired individuals(Phadnis et al., 2020).

(Wahab et al., 2021)This paper proposed an affordable mobile application designed for visually impaired individuals. The application was capable of capturing an image of printed material using a mobile camera. The captured image was then converted to text through an Optical Character Recognition (OCR) framework, which utilized image-to-text conversion. Finally, the text was converted to speech format through a Text to Speech (TTS) framework. As a result, visually impaired individuals were able to understand printed material that was not written in

Braille by listening instead of touching it. The application was user-friendly and provided alert sounds to guide users, as well as to inform them of actions taken within the application.

The paper "Raspberry Pi-Based Reader for Blind People" presents a solution for visually impaired individuals to access printed materials. The proposed solution uses a Raspberry Pi and integrates Optical Character Recognition (OCR) and Text-to-Speech (TTS) technologies. The system captures images of typed, handwritten, or printed text using image-sensing devices and converts the images into machine-encoded text using the Tesseract library and Python programming. The text files are processed with OpenCV library and Python programming language to produce audio output in the form of speech. The paper highlights the implementation of this automatic document reader to help visually impaired people access and understand printed materials more easily(Rani et al., 2022).

Another study conducted by (Vaithiyanathan & Muniraj, 2019) proposed the development of an assistive device for visually impaired people, a smart reader that captured an image from a camera, extracted text from the captured image, and converted it to speech as voice output. The image captured by the reader was analyzed using Google Cloud Vision API's Optical Character Recognition (OCR). Image preprocessing methods were employed to remove noise or blur, thus increasing the accuracy of the extracted text. The text-to-speech conversion was performed using software-based TTS technology. The device was implemented using Raspberry Pi interfaced with a night vision camera and a speaker, producing the converted text as speech. The use of Google Cloud Vision OCR with image enhancement methods resulted in high accuracy levels of the extracted text.

The authors of this paper present a solution for individuals with visual impairments or illiteracy to understand Bangla text documents(Rajbongshi et al., 2020). By using a camera-based device, the text within an image is extracted and then converted to speech, allowing the user to listen to the contents. The device was implemented using Raspberry Pi and a camera module, leveraging the Tesseract OCR engine, the Open Source Computer Vision, and the Google Speech Application Program Interface. The results show that this solution is effective in helping individuals who speak Bangla and are unable to read or have limited visual sight. The paper highlights the importance of providing accessible and convenient tools for individuals with visual impairments. By using a combination of cutting-edge technologies, the authors were able

to develop a device that can make a significant impact on the lives of those who face challenges in reading. The device's compact size and ease of use make it an attractive option for individuals who are always on the go, and its compatibility with the Bangla language makes it a valuable resource for the target demographic.

Table 2: Summary of Related Works

No.	Authors	Methods and Tools Used	Gaps
1.	(Alsaid et al., 2019)	Raspberry Pi 3 B+, OpenCV, Tesseract OCR, EAST (Efficient and Accurate Scene Text Detector), Google Text-to-Speech API	This field of study was dependent on language, and most efforts had focused on accessible languages, particularly English.
2.	(Pandya & C.Goradiya, 2021)	OCR model, Convolutional Neural Network (CNN)	
3.	(Mathur et al., 2019)	OCR, Camera application, Text-To-Speech module	
4.	(Wahab et al., 2021)	Optical Character Recognition (OCR) framework, Text-to-Speech (TTS) framework	
5.	(Rani et al., 2022)	Raspberry Pi, OCR, Text-to-Speech (TTS) technologies	
6.	(Ragavi et al., 2016)	Handheld page scanner, Android phone, Image processing, Optical Character Recognition (OCR), Text-to-Speech (TTS) library	
7.	(Vaithiyanathan & Muniraj, 2019)	Raspberry Pi, Google Cloud Vision API, Image preprocessing, Text-to-Speech (TTS) technology	
8.	(Rajbongshi et al., 2020)	Raspberry Pi, Camera module, Tesseract OCR engine, Open Source Computer Vision (OpenCV), Google Speech API	

2.13.1 Speech Synthesis for Afaan Oromoo Language

In 2020, Muhidin Kadir conducted research on developing a text-to-speech synthesizer using statistical parametric speech synthesis. The author emphasized the importance of text preparation, which encompasses tokenization, normalization, phonetic representation of sounds by graphemes, and modeling of the prosodic properties of various speaking styles. To tackle these challenges, various tactics were explored and employed. The author pointed out that existing speech synthesizers, which were based on hidden Markov models (HMM), were not suitable for the Afaan Oromoo language as they did not account for the unique characteristics of the language. Therefore, statistical parametric speech synthesis based on HMM approaches was used in the study and a dataset consisting of 400 sentences and speeches was utilized(Wosho, 2020).

In 2021, Chala Sembeta conducted research on a Text-to-Speech synthesizer for the Afaan Oromoo language using a deep neural network approach(Tashoma, 2021). The purpose of the system was to provide support for individuals with visual impairments and reading disabilities by allowing them to listen to written words across various forms of media. To build the synthesizer, the author gathered a speech dataset consisting of 8076 text and audio pairs from credible sources in the Afaan Oromoo language. The performance of two deep neural network models, Tacotron 2 and Deep Voice 3 was compared through a series of trials. Both objective and subjective evaluations were carried out to determine the effectiveness of the models, with the results showing that the Tacotron 2 model, based on recurrent neural networks, offered the best outcome.

(Beyene, 2022)researched developing a text-to-speech synthesis for the Afaan Oromo language using deep learning. In this study, linguistic features were extracted from the normalized text using the Festival toolbox for Afaan Oromoo TTS. The labeled texts were generated using the parameters of a scheme file and were normalized for phoneme alignment with speech corpora in both the training and testing datasets. The main focus of the research was on the TTS approach using a deep learning model based on a bidirectional long-short-term memory recurrent neural network (BLSTM-RNN) for the Afaan Oromoo language. The RNN model was utilized to extract both a duration model and an auditory model from a given input feature sequence. A speech corpus of a female speaker was used to create a corpus of 1000 texts, with 700 sentences

for training and 300 sentences for testing, along with matching text transcriptions from various dataset domains.

Table 3: Summary of Related Afaan Oromoo Text-to-Speech Synthesis

Authors	Methods	Total Dataset	Mean Opinion Score	Gaps
(Wosho, 2020)	Hidden Markov Model	600 sentences and transcribed audio	Intelligibility: 4.3 Naturalness: 4.1	HMMs face difficulties in capturing the intricate relationships and subtleties in speech, such as prosody, intonation, and timing(Abraham et al., 2022)
(Tashoma, 2021)	CNN based Deep Voice3 and RNN based Tacotron2	8076 sentences And transcribed audio	[Deep Voice 3] Intelligibility: 3.28 Naturalness: 3.02 [Tacotron2] Intelligibility: 4.32 Naturalness: 4.21	Use of traditional Griffin-Lim algorithm for vocoder which does not generate high quality speech compared to flow-based generative models(Masuyama et al., 2019). Griffin-Lim lacks modeling of fine-grained details and does not provide explicit control over voice characteristics such as pitch and prosody and also reduced quality in challenging scenarios.
(Beyene, 2022)	Festival toolkit for Feature Extraction based on BLSTM-RNN	1000 sentences And transcribed audio	Intelligibility: 3.77 Naturalness: 3.76	Small dataset and lacks intelligibility and naturalness. Festival toolkit relies on manually designed linguistic features for feature extraction, which may not capture all the nuances and complexities of the input text.

CHAPTER THREE

3. RESEARCH METHODOLOGY

The chapter provides an overview of the methods and techniques used to accomplish the research objectives. The first section focuses on the data collection and preparation methods utilized in the thesis. These methods play a crucial role in ensuring that the data is suitable for analysis and evaluation.

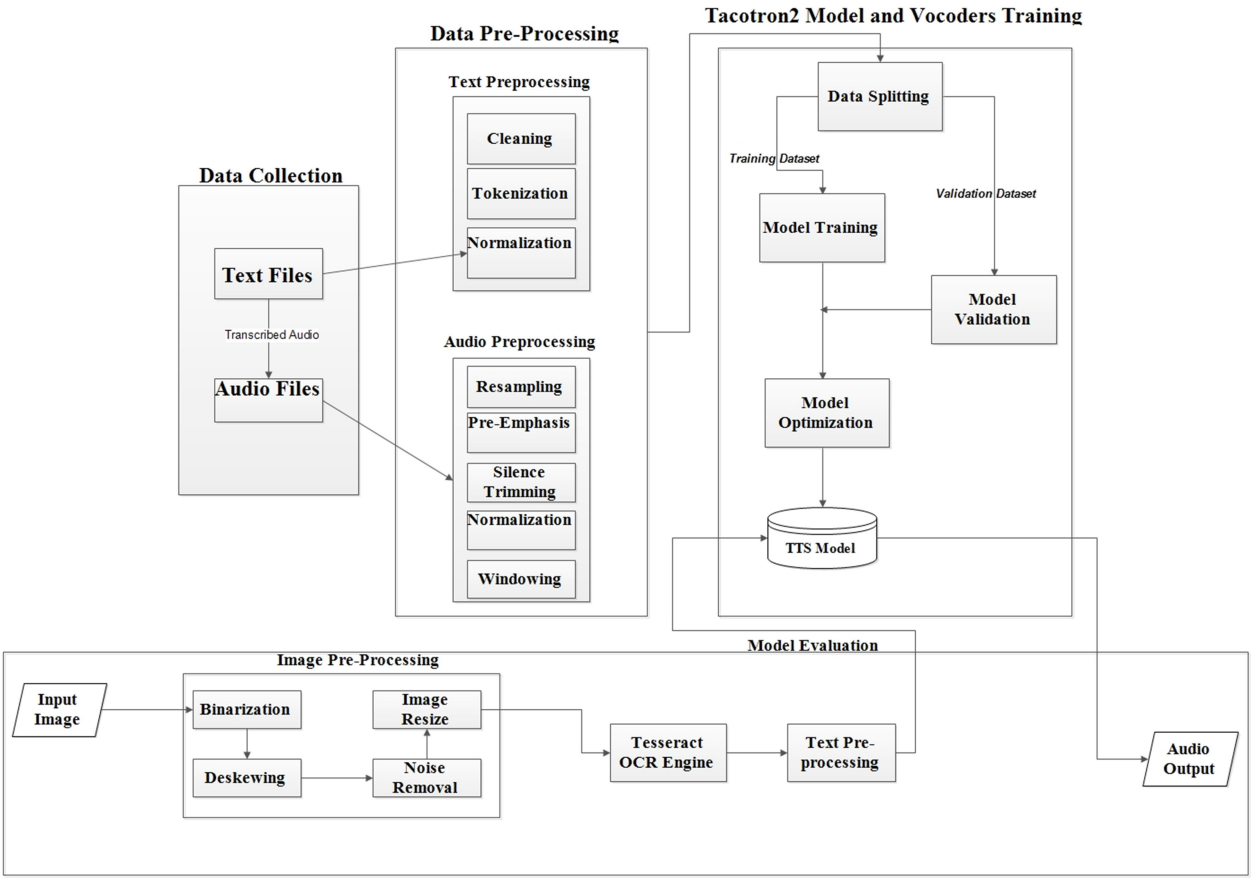


Figure 7: Flow of Proposed Solution Methodology

The second section of the chapter discusses the Tesseract-OCR feature representation and text extraction from an image. Tesseract-OCR is a popular open-source optical character recognition (OCR) engine used to extract text from images. The section explains how the engine works and the features it uses to recognize text. It also discusses the use of other techniques to improve the accuracy of Tesseract OCR, such as image processing. The third section of the chapter describes the speech feature representation. It explains how features are used to represent the audio signal in the proposed model.

The fourth portion of the chapter describes the models' organization, structure, and details. The most well-known TTS models and their operating principles are described in this section. We concentrated on Tacotron 2, a model based on recurrent neural networks (RNNs) that are commonly applied in TTS systems. Training the Tacotron2 model from scratch and also fine-tuning the pre-trained model. The training of WaveGlow and HIFI-GAN vocoders, which are used with Tacotron2 to synthesize speech, is also covered in this section. The last part of the chapter focuses on the evaluation techniques used in this research work.

3.1 Data Collection and Data Sources

The study developed an extensive knowledge of the problems experienced by visually impaired people in reading written material by studying several research papers on the difficulty and challenges faced by visually impaired people. In Addition to Online Available dataset source, we have collected text data for the study from various sources, including news articles from Fana Broadcasting Corporate 326 text sentences and from Oromia Broadcasting Network 264 text sentences, religious books from Holy-Bible 224 text sentences and from Holy-Quran 110 text sentences. These sources were utilized to create a comprehensive dataset of sentences and transcribed audio pairs for the development of the TTS system.

3.1.1 Text Data Collection

To train a TTS system effectively, it is necessary to have a sizable dataset of human speech recorded in natural settings. The LJ Speech Dataset is widely accepted as the benchmark for English language DNN-based synthesizers, and most researchers adhere to its standard guidelines for dataset collection and preparation. To develop a phonetically rich and well-balanced Afaan Oromo speech corpus, a large amount of raw text sentences was gathered from various sources, resulting in a corpus that incorporated a broad range of domains, including a variety of sentence types, structures, and lengths. This approach was expected to enhance the TTS system's quality significantly. The study used a Chala Sembeta dataset that was publicly available online as a benchmark for the development of the Afaan Oromoo Text-to-Speech model. This dataset was created from 8076 Text sentences and transcribed audio pairings from a prior study on Text to Speech Synthesis for Afaan Oromo done at Adama Science and Technology University. We also prepared a custom dataset of 924 Text sentences and transcribed

audio pairs. The benchmark dataset and our custom dataset were combined to create a total dataset size of 9000 sentences and transcribed audio pairs for our investigation.

3.1.3 Audio Recording Process

We used a Samsung A70 smartphone equipped with a speech-recording app to transcribe sentences into an audio data. With a built-in microphone, and numerous recording settings, the Samsung A70 smartphone audio recorder was used to capture high-quality audio. A professional male speaker from Dilla University Department of Afaan Oromoo was invited to read the sentences. The audio data was collected for this study using a Samsung A70 smartphone and then pre-processed to the sampling rate of 22050 Hz, with a bit depth of signed 16 bits using Audacity software. The audio files are saved in WAV format and represented in PCM format as well as pre-processed to mono-channel format.

3.2 Data Pre-Processing

Data preprocessing is a crucial step in transforming raw data, which may contain inconsistencies, incompleteness, and errors, into a desirable format that can be used for various purposes. In deep learning technology, data preprocessing is an essential step to achieve optimal results(Adam, 2020). It involves organizing the data in a way that fits the training model in terms of size and type. Well-organized data are preferable in any field of artificial intelligence as they simplify the learning process. This process involves cleaning the data by removing irrelevant information, transforming it into a supported format, and standardizing it(Adam, 2020). It helps in the iterative training and testing process.

3.2.1 Image Pre-Processing

The popular optical character recognition (OCR) tool Tesseract OCR is used to extract text from images. Pre-processing measures are frequently required for OCR-Tesseract to produce the best results to enhance the image quality and increase the text's legibility(V. S & A, 2015).

The image pre-processing steps for Tesseract OCR typically involve the following:

1. **Binarization**: The image is converted to a binary format, where each pixel is either black or white. This is done to make the text stand out from the background and to remove any noise or artifacts in the image.
2. **Deskewing**: The image is rotated or skewed to align the text horizontally or vertically. This is done to make the text more readable by the OCR engine.

3. **Noise Removal**: Any unwanted noise or artifacts in the image are removed using filters or other image processing techniques. This can improve the accuracy of the OCR engine.
4. **Image resizing**: This step involves resizing the image to a specific size. A good size to work with for OCR is 600 dpi.

By performing these pre-processing steps, OCR-Tesseract can more accurately extract text from images and produce higher-quality results(Mageshwaran, 2021). Other significant image pre-processing methods utilized by Tesseract OCR to extract text from an image include blurring and image Thresholding. Applying a filter to an image to smooth it out is the process of blurring. By reducing background noise, this method can help OCR software recognize text more easily. Image Thresholding is the technique of using a threshold value to transform a grayscale image into a binary image. Text is frequently separated from an image's background using this method.

3.2.2 Text Pre-Processing

Before the dataset is fed into the deep-learning models, several textual and audio preprocessing processes are used to enhance the quality of the audio and text files. Text pre-processing is an essential step for text-to-speech (TTS) synthesis to ensure accurate and natural-sounding voice output(Reichel & Ptzinger, 2006). Some common text pre-processing steps for TTS synthesis include:

Text Cleaning: Removing unnecessary characters, punctuation, and other special symbols from the text data.

Tokenization: Breaking down the text into individual words or phrases, can be further analyzed and processed.

Text Normalization: Converting text to a standard format or language, such as converting abbreviations to their full form, or converting text in a different language to the target language. With the aid of these pre-processing stages, it is possible to make sure that the text input is accurately translated by the TTS system and that the speech output that results is precise, audible, and understandable.

3.2.3 Audio Pre-Processing

To produce high-quality synthesized speech, audio pre-processing is a critical stage in the Text-to-Speech (TTS) synthesis process. The following are a few typical audio pre-processing steps used in TTS:

Resampling: Converting the audio to a desired sampling rate to match the requirements of the TTS system. We used sampling rates of 22.05 kHz.

Pre-emphasis: Applying a high-pass filter to amplify the higher frequency components of the audio signal, and reduce the lower frequency components(Anuj, 2021). This helps to improve the intelligibility of the synthesized speech.

Silence Trimming: Removing any long periods of silence from the audio signal, as they do not contribute to the synthesized speech.

Normalization: Scaling the audio signal to ensure that the amplitude is within a desired range, usually between -1 and 1.

Windowing: Dividing the audio signal into smaller frames, typically 20-30 milliseconds in length, to make it more manageable for the TTS system to process.

Spectral Analysis: Analyzing the frequency content of each audio frame using Fourier transform, and representing the frequency spectrum in a way that is useful for the TTS system.

3.3 Character Embedding

When working with low-resource languages that have little training data and few pre-trained word embedding like word2vec, character embedding is used. Character embedding may perform better in these situations than word-level embedding. This is so that character embedding, which can be trained even with minimal datasets, can capture both the semantic and syntactic information of the text. OOV words can also be handled by character embedding because they can be broken down into individual characters and still have a meaningful representation. In contrast, word2vec embedding needs a lot of text input to learn word representations, and word-level embedding may perform poorly in the absence of this data(Edward, 2019).

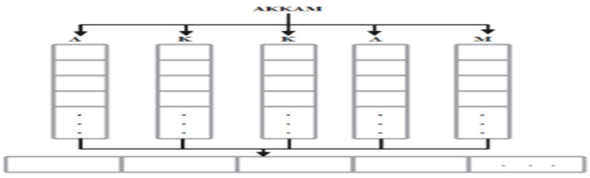


Figure 8: Character Embedding(Tashoma, 2021)

3.4 Image Representation

The term "image representation" refers to the feature representation of an image as a matrix of pixel values. It is a technique for displaying a picture as a two-dimensional array of numerical

values, where each value denotes the intensity of a specific pixel in the image(Himanshi, 2021). Input for computer vision algorithms like object identification, picture classification, and image segmentation is frequently provided through image representation. Images are commonly stored in OCR-Tesseract as a matrix of pixel values, where each pixel refers to a specific spot on the picture and its value denotes the brightness of that spot. In essence, this matrix is a 2D array that can be processed using several image-processing methods to extract characteristics like edges, corners, and blobs that may subsequently be used for character recognition.

Example of a 4x4 grayscale image represented as a matrix of pixel values.

```
[[255, 255, 255, 255],
[128, 128, 128, 128],
[ 64,  64,  64,  64],
[  0,   0,   0,   0]]
```

The image has 4 rows and 4 columns, and each element in the matrix represents the intensity value of a pixel in the image. The values range from 0 (black) to 255 (white), with the middle values representing different shades of gray. OCR-Tesseract generates a text output once the picture has been analyzed and the characters have been identified. A text-to-speech (TTS) system can then take this text output as input, turning it into an audio signal.

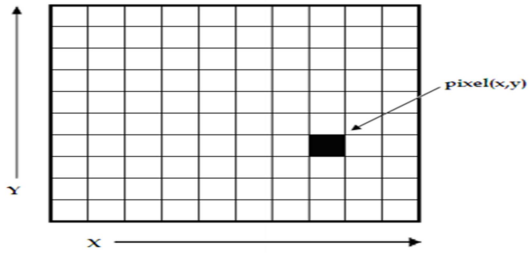


Figure 9: Image Representation as a matrix of pixel example

3.5 Image Model Architecture

Tesseract-OCR uses deep neural network architecture to recognize text from an image. The architecture consists of several layers of convolutional and recurrent neural networks(Harald, 2018). A high-level overview of the OCR-Tesseract neural network architecture is:

Input Layer: The input is an image of text.

Convolutional Layers: These layers extract features from the input image by applying convolutional filters.

Pooling Layers: This layer reduces the size of the feature map obtained from the convolutional layers by down-sampling it.

Recurrent Layers: These layers process the sequence of feature maps obtained from the pooling layers and extract contextual information.

Fully Connected Layers: These layers combine the features obtained from the convolutional and recurrent layers and output a sequence of probability distributions over the possible characters in the input image.

Decoding Layer: These layer uses the probability distributions to generate the final output text.

3.6 Acoustic Feature Representation

Acoustic feature representation refers to the process of converting audio signals into a sequence of features that can be used for further analysis or processing. Acoustic feature representation for Text to Speech involves transforming the text input into a series of audible features that may be utilized to produce speech(Álvarez et al., 2019). A neural network-based model, like Tacotron2 or FastSpeech, is often used for this. It takes a text sequence as input and generates a series of acoustic characteristics, such as Mel-spectrograms or linear spectrograms. To create the final speech waveform, these acoustic data are then sent into a vocoder like WaveGlow or HiFi-GAN. To build a representation of the speech signal that captures the pertinent acoustic information required for synthesis, acoustic features are generated using a mix of text processing, linguistic analysis, and signal processing techniques.

3.6.1 Linear Spectrogram

The energy of an audio signal in each frequency band over time is shown by a linear spectrogram, sometimes called a frequency spectrogram. They are commonly produced by performing the Fourier transform of an audio signal over a brief window, which transforms the signal from the time domain to the frequency domain. As a consequence, a 2D matrix with each row denoting a frequency band and each column denoting a time step is what is known as a spectrogram(Dennis et al., 2011). When examining the spectrum content of an audio stream, linear spectrograms can be helpful, but they are not a precise representation of how the human ear hears sound.

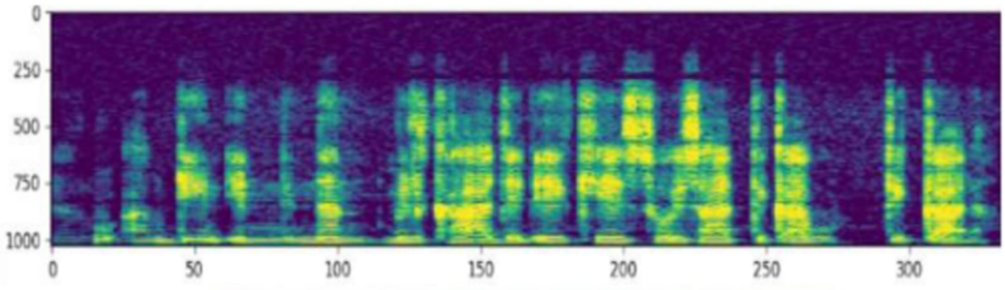


Figure 10: Example of Linear Spectrogram

3.6.2 Mel-Spectrogram

Mel-spectrograms are a particular kind of spectrogram that is frequently employed in voice synthesis and recognition(Roberts, 2020). They are produced by converting the linear spectrogram via a collection of triangle filters that are positioned further apart at higher frequencies and closer together at lower frequencies. A Mel-spectrogram uses the Mel scale to describe the frequency range of a sound wave visually. The Fast Fourier Transform (FFT) of a short-time windowed signal is used to construct it, and the resulting power spectrum is then mapped onto the Mel scale. The resultant Mel-spectrogram is a two-dimensional matrix with time and frequency represented on the x and y axes, respectively. The energy or intensity of each frequency at each respective time is represented by a matching cell in the matrix(Roberts, 2020).

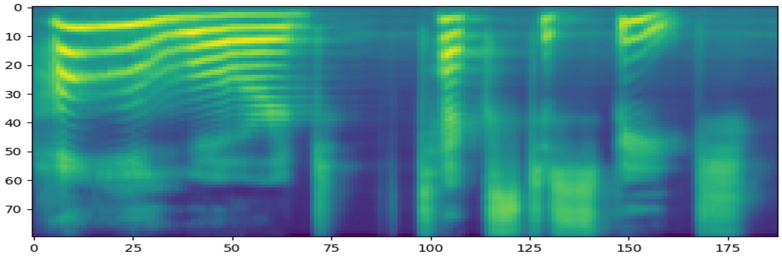


Figure 11: Mel-Spectrogram Example

3.7 Acoustic Model Architecture

Typical acoustic model architecture for TTS based on a neural network consists of three major components: text analysis, acoustic models and vocoders. The neural network architecture integrates text analysis and acoustic modeling together to simplify the existing pipeline to build a neural network-based TTS system and models complex sequential mapping directly from a text sequence (e.g. phonemes, syllables or words) to its acoustic trajectory(W. Wang et al., 2016).

Depending on the kind of neural network and the particular implementation, the acoustic model's unique design may change. For audio modeling in TTS, some frequently used models include:

1. **Convolutional Neural Networks (CNNs):** These are often used for modeling spectral features, such as Mel-spectrograms or linear spectrograms. The input sequence of spectral frames is fed through one or more layers of convolutional filters to extract local features, which are then aggregated over time using pooling layers or recurrent connections(Zhang et al., 2019).
2. **Recurrent Neural Networks (RNNs):** These are often used for modeling temporal dependencies in acoustic features, such as the timing and duration of phonemes. RNNs can be used in a variety of configurations, including standard recurrent networks (such as vanilla RNNs or LSTMs), bi-directional networks (which process the input sequence in both forward and backward directions), and attention-based networks (which dynamically weight the input sequence based on its relevance to the current output)(Marchi et al., 2017).
3. **Transformer Based Models:** These are a more recent development in TTS, and are based on the Transformer architecture originally developed for natural language processing tasks. Transformer-based models can be used to model both spectral and temporal features, and are particularly well-suited for handling long-range dependencies in the input sequence(Ro et al., 2022).

3.8 Flow-Based Model Architecture

Flow-based models and generative models are used as vocoders in text-to-speech systems such as WaveGlow and HIFI-GAN. They use a flow-based model to map a distribution of noise to a distribution of audio features. The key advantage of flow-based models is that they are invertible, meaning that they can be used to synthesize audio from noise or to generate noise from audio features(Prenger et al., 2019).

WaveGlow is a flow-based generative model that uses an invertible 1x1 convolution to generate audio samples. It is typically used with a text-to-speech model like Tacotron2, which generates Mel-spectrograms that are then converted to audio using WaveGlow(Prenger et al., 2019). Similarly, **HIFI-GAN** is another generative adversarial model used for text-to-speech applications, which is used with a text-to-speech model like Tacotron2 and FastSpeech2. HIFI-GAN uses a multi-scale invertible generator that operates on the Mel-spectrogram domain to

generate high-fidelity audio(Aaron, 2021b). In our study, we used WaveGlow with tacotron2 and HIFI-GAN with fine-tuned pre-trained Tacotron2 model to generate audio for the Afaan Oromo language. Vocoders allow for real-time synthesis of speech and HIFI-GAN can generate high-fidelity audio with minimal distortion. It is capable of generating audio that is almost indistinguishable from real human speech. In the end, we compared the output findings from the two models.

3.9 Transfer Learning

The paper "A Concise Review of Transfer Learning" by (Farahani et al., 2020), Provides a concise review of transfer learning. It explains that transfer learning is a technique in machine learning where a model trained on one task is used as the starting point for a model on a second task. In the case of speech synthesis, transfer learning can be used to train a model for a new language or voice using a pre-trained model that has been trained on a different language or voice. It also covers the types of transfer learning, applications of transfer learning, challenges in transfer learning, and future directions in transfer learning. We used a pre-trained Tacotron2 model that was trained on LJSpeech in English and fine-tuned it using our dataset in the Afaan Oromo language. By doing so, we were able to leverage the knowledge gained during the pre-training stage and use it to train a new model that can synthesize speech in the Afaan Oromoo language.

3.10 Model Evaluation Method

In this study, the performance of OCR-Tesseract was assessed using the word error rate, while the Text-to-Speech model was assessed using the Mean Opinion Score (MOS) and Mel Cepstral distortion (MCD).

3.10.1 Word Error Rate (WER)

The study evaluated the effectiveness of Tesseract OCR by analyzing the word error rate, while the performance of the Text-to-Speech model was assessed using the Mean Opinion Score and Mel Cepstral distortion (MCD).

The following procedures were commonly used to calculate WER:

1. The OCR system generates a transcription of the input image.
2. The transcription is compared to the ground truth text (i.e., the actual text in the image).

3. The number of word errors is calculated, which is the number of words that are incorrectly transcribed or missing.

WER is then calculated by dividing the total number of word errors by the total number of words in the ground truth text.

3.10.2 Mean Opinion Score (MOS)

Mean Opinion Score (MOS) is a subjective method used to evaluate the quality of speech generated by Text-to-Speech (TTS) systems. It is a rating given by a human listener to rate the perceived quality of the speech output on a scale of 1 to 5, with 1 being the lowest quality and 5 being the highest quality(Aaron, 2021a).

To evaluate the Afaan Oromo TTS system, human listeners are asked to rate the quality of the speech generated by the system on a scale of 1 to 5 based on the overall impression of the speech, such as naturalness, intelligibility, and similarity to human speech. The listeners are usually native speakers of Afaan Oromo to ensure that the evaluation is done by people who are familiar with the language. The MOS score is then calculated by taking the average rating of all the human listeners. The higher the MOS score, the better the perceived quality of the speech generated by the TTS system.

3.10.3 Mel Cepstral Distortion (MCD)

Mel Cepstral Distortion (MCD) is a measure of how different two sequences of Mel cepstra are, which is widely used to evaluate the performance of speech synthesis models. The MCD metric compares k-th (default k=13) Mel Frequency Cepstral Coefficient (MFCC) vectors derived from the generated speech and ground truth, respectively[2]. However, it's important to note that MCD is a perceptual metric and may not fully capture the quality of synthesized speech as perceived by human listeners(Anumanchipalli et al., 2019).

The following procedure shows how Mel Cepstral Distortion evaluation works:

Target and Synthesized Speech: start with a set of target speech and the corresponding synthesized speech generated by the TTS system to evaluate.

Feature Extraction: Extract the MFCC features from both the target and synthesized speech. MFCCs capture the spectral envelope of the speech signal and are commonly used for representing speech.

[2] https://pypi.org/project/pymcd/

Alignment: Align the target and synthesized MFCC features using dynamic time warping (DTW). Alignment ensures that the corresponding frames of the target and synthesized speech are matched.

Frame-Wise Comparison: Compare the MFCCs frame-by-frame between the aligned target and synthesized speech. Compute the Euclidean distance or another distance measure between the MFCC vectors of each frame; in our case we used Root Mean Square Error (RMSE).

Average Distortion: Calculate the average distance or distortion across all the frames in the aligned target and synthesized speech. This average distance is the MCD.

Interpretation: Lower MCD values indicate better similarity between the target and synthesized speech. A smaller MCD suggests that the TTS system produces synthesized speech that closely resembles the target speech in terms of spectral characteristics.

CHAPTER FOUR

4. PROPOSED SOLUTION AND DESIGN

This chapter outlines the deep neural network-based solution that has been suggested for the Afaan Oromo text reading model. Three parts make up the suggested solution: the OCR-Tesseract implementation phase, which extracts text from images, the Text-to-Speech model training phase, and voice synthesis. This section discusses the specifics of each of the elements that were utilized in the text extraction, model training, and synthesis stages. This section also includes a discussion of the conceptual framework design which shows how text is extracted and then passed to text-to-speech system to generate corresponding audio of text.

4.1 The Proposed Solution

In this study, a plan of action for creating a text-reading model for the Afaan Oromo language was put out. The suggested method was using Tesseract-OCR to extract Afaan Oromo text from an image. A strong optical character recognition engine called Tesseract-OCR can extract text from an image. Because of the structure and quality of the image, image pre-processing is used before feeding it to the Tesseract-OCR engine to accurately detect its text.

The Tacotron2 deep learning model was utilized for speech synthesis after the Afaan Oromo text was retrieved from the image. A cutting-edge neural text-to-speech model called Tacotron2 can produce speech that sounds natural from text input. A predefined Afaan Oromo dataset with text-audio pairings was utilized to train the model since Tacotron2 needs a lot of training data to function successfully. We have trained the Tacotron2 model from scratch on the Afaan Oromo dataset corpus and also using a pre-trained model on the LJSpeech dataset corpus we fine-tuned the model with the Afaan Oromoo dataset. Lastly, the speech was synthesized using the WaveGlow neural vocoder, which we used for the model that was trained from scratch, and the HIFI-GAN neural vocoder, which we used for the model that was fine-tuned. WaveGlow and HIFI-GAN models, a flow-based generative model, were used with Tacotron2's output to produce high-quality speech, and the output of both models was compared using unseen data.

Image containing Afaan Oromo text were used to test the suggested solution, and the quality of the synthesized speech that was produced was assessed using the Mean Opinion Score (MOS). The evaluation's findings revealed that the suggested method was capable of properly extracting

Afaan Oromo text from images and producing speech that sounded natural. Afaan Oromo speakers, especially those with visual impairments, can benefit from improved accessibility and communication according to this research's major contribution to the development of technology for the Afaan Oromo language. The figure below shows the general architecture of the proposed solution and how we can generate audio speech from image containing Afaan Oromoo text.

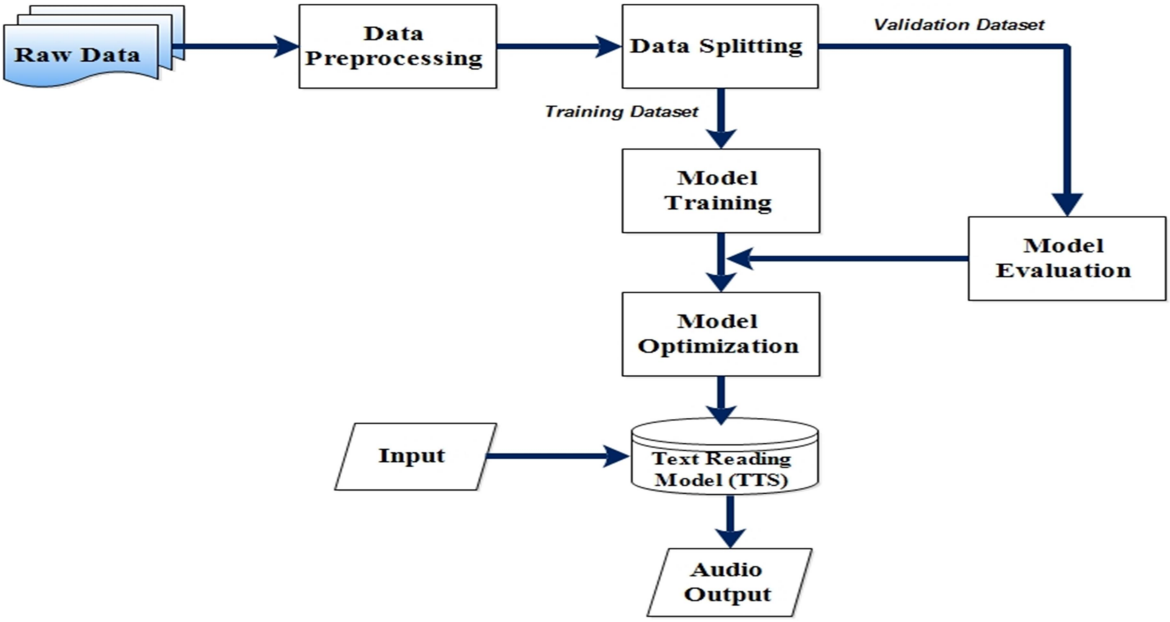


Figure 12: Text-to-speech Model Training Architecture

The above figure shows the architecture of proposed solution how to train text-to-speech synthesis model. The first step is to pre-process the collected raw data as we explained in the previous chapter which is the text and audio dataset. After we pre-processed both text and audio dataset, we split the data into training set which is used to train our model and validation set which is used to validate our model. By using model optimization technique, we minimize the error between training and validation loss of the model. The above architecture is used for Tacotron2 model training which is acoustic feature prediction model and training flow-based generative vocoders WaveGlow model as well as high fidelity generative adversarial network (HIFI-GAN) model. After the model is trained successfully, text input is given to test the model performance. The input text is passed through Tacotron2 model to predict acoustic feature of the input text. The predicted Mel-spectrogram is then given to vocoder to generate the final audio output from the predicted Mel-Spectrogram.

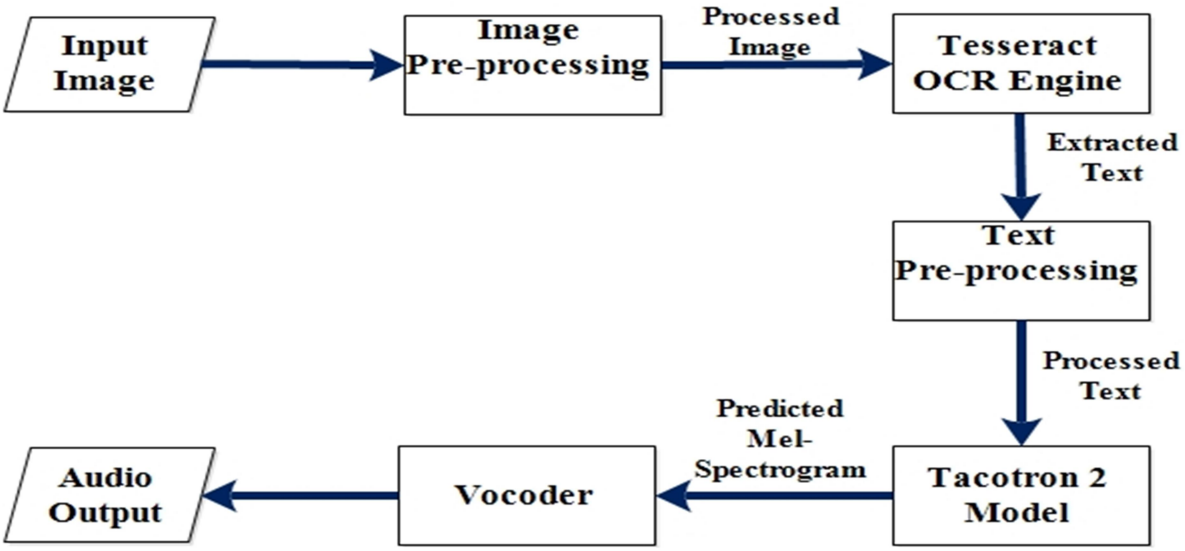


Figure 13: Architecture of Speech Synthesizing

Figure 13 of above shows the architecture of how we can synthesize speech directly from an image containing Afaan Oromoo text. The input image of first pre-processed as explained in previous chapter then passed to Tesseract OCR Engine to extract text from an image. The extracted text then pre-processed and passed to Tacotron2 model to predict acoustic feature of input which is Mel-Spectrogram. The predicted Mel-Spectrogram from Tacotron2 model is forward to vocoder (WaveGlow or HIFI-GAN) to generate the audio from Mel-Spectrogram.

4.2 Tesseract-OCR Implementation

To implement the Tesseract-OCR engine, we used the Python programming language and its pytesseract library, which provides an interface for using the Tesseract-OCR engine in Python. We created a script that reads the image file, preprocesses the image, and extracts the text from the image using the Tesseract-OCR engine. The extracted text is then saved to a text file for further processing. To evaluate the performance of our Tesseract-OCR implementation, we used the Word Error Rate (WER) metrics, which measures the difference between the recognized text and the ground truth text. Tesseract-OCR implementation achieved an accuracy of more than 95% depending on the image quality and low word error rate, as measured by the Accuracy metric.

Image Analysis: Tesseract-OCR is used to analyze image and extract Afaan Oromo text from them as part of our suggested solution for the "Text Reading Model for Afaan Oromo Language". To handle less data, the input picture is first made grayscale as part of the image analysis

process. After that, the image is preprocessed to improve its quality and get rid of any noise or distortions. The Tesseract-OCR engine receives the image after preprocessing to extract the Afaan Oromo text. Open-source software called Tesseract-OCR employs optical character recognition (OCR) to identify text contained inside a picture. It operates by breaking the image into smaller pieces and searching for text in each piece. The extracted text is then saved in a text file after being identified(Smith, 2005). The WER meter and overall Accuracy are trustworthy indicators of how well our implementation performed. The figure below shows the Architecture of the Tesseract-OCR System.

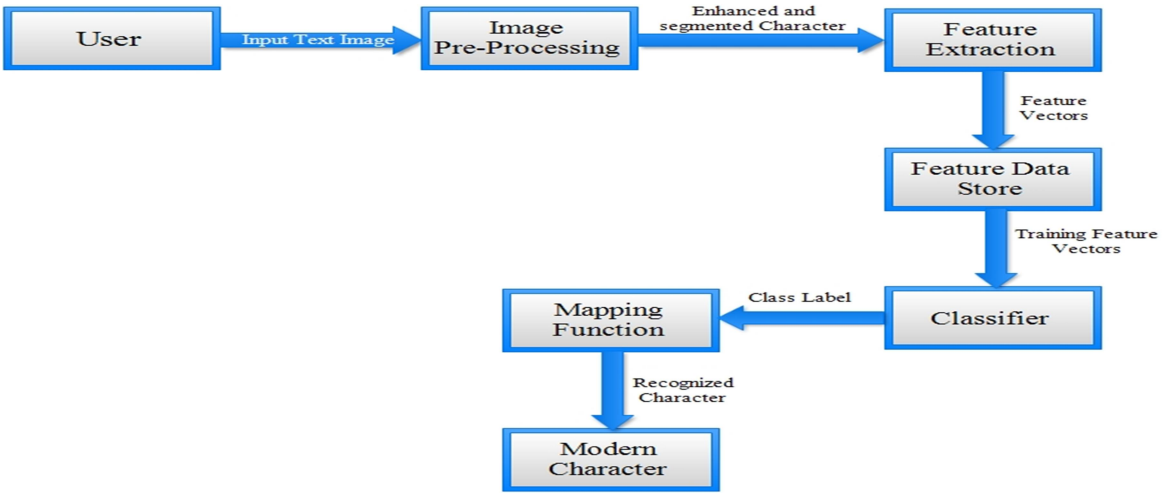


Figure 14: Architecture of Tesseract-OCR System(A. S. & G, 2015)

The figure above show that systematic flow of Tesseract OCR system, starting with preprocessing to enhance image quality, followed by feature extraction to capture important characteristics, classification using an ANN to identify characters, and finally post-processing to present the results in a modern and accessible format.

4.3 TTS Model Training

During the training phase, based on the NVIDIA Github source[3], we used RNN-based Tacotron2 parameter prediction models and trained them from scratch. And also using our own Afaan Oromoo corpus along with the pre-trained model for the English language on the LJSpeech dataset, we fine-tuned it for the Afaan Oromoo language. These models are designed to optimize the processing of the text and audio data being used. However, to effectively train the acoustic model, it is important to analyze the incoming data before feeding it into the model. This analysis

[3] https://github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/SpeechSynthesis/Tacotron2

ensures that the model can accurately capture the degrees of the Afaan Oromo language and produce high-quality speech output. Before training the Tacotron 2 model, it is essential to preprocess the input data for optimal training. This preprocessing includes text analysis, character embedding, and audio analysis. The detailed text and audio analysis are explained in the previous Research methodology section.

Text Analysis: In the text analysis phase, we preprocess the input text by performing various linguistic analyses such as tokenization, text cleaning, and normalization. These analyses help to extract the necessary linguistic features from the input text, which are then fed into the Tacotron 2 model.

Character Embedding: Once the linguistic features have been extracted from the input text, we need to convert them into numerical vectors. To achieve this, we used character embedding techniques, which map each character to a high-dimensional vector space. By doing this, we can preserve the semantic relationships between characters, which are essential for accurate speech synthesis. Character embedding, as opposed to word2vec embedding, was chosen because it can handle uncommon words better and minimizes model complexity by employing a limited number of vectors. It resolved some word embedding. The difference between Character Embedding and Word Embedding is that Character Embedding can build any word as long as those characters are included(Edward, 2018).

Audio Analysis: In the audio analysis phase, we preprocess the input audio data by performing feature extraction, such as spectral analysis, to extract the relevant audio features. These features are then used to train the Tacotron 2 model to synthesize speech that accurately reflects the input audio. By performing text analysis, character embedding, and audio analysis, we can preprocess the input data in a way that maximizes the performance of the Tacotron 2 model during training.

4.4 Acoustic Model

Our proposed solution for acoustic modeling is based on Recurrent Neural Network (RNN). Specifically, we utilized the RNN-based Tacotron 2 model for our Afaan Oromo TTS system. During the training phase, we trained Tacotron 2 parameter prediction models from scratch as well as using a pre-trained model by fine-tuning it.

4.4.1 Recurrent Neural Network Based Model

The RNN-based Tacotron 2 model is a deep learning-based architecture that can generate Mel spectrograms, which represent the speech acoustic features. The model is composed of two main parts: the text encoder and the decoder. The text encoder uses an RNN to encode the input text, while the decoder uses another RNN to generate the Mel spectrogram(Shen et al., 2018).

The Tacotron 2 model predicts Mel-spectrograms from text and is a recurrent sequence-to-sequence model with attention. The encoder converts the entire text into a fixed-size hidden feature representation (blue blocks in the diagram below). The autoregressive decoder (orange blocks), which generates one spectrogram frame at a time, then consumes this feature representation. The flow-based generative WaveGlow replaces the autoregressive WaveNet (green block) in NVIDIA implementation(Shen et al., 2018).

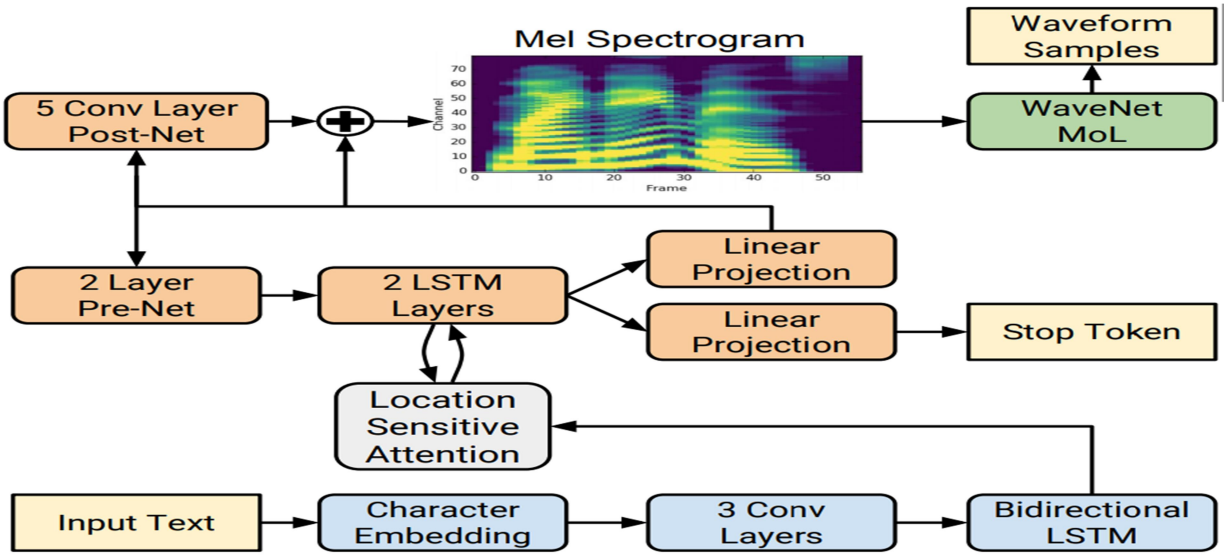


Figure 15: Architecture of Tacotron 2(Shen et al., 2018)

The sequence-to-sequence network consists of several layers:

The **Embedding Layer** converts each character into a 512-dimensional vector that represents its semantic and syntactic features. This helps the model learn the relationship between different characters and words(Shen et al., 2018).

The **Pre-net Layer** applies two fully connected layers with ReLU activation and uses Dropout instead of Zoneout to regularize the LSTM layers to the embedded characters. This helps the model reduce overfitting and learn more robust features(Shen et al., 2018).

The **Encoder Layer** applies a convolutional neural network (CNN) with 3 layers, each with 512 filters and kernel size 5, followed by a bidirectional LSTM with 512 units in each direction to the output of the pre-net layer. The CNN captures local dependencies and context information in the input sequence, while the LSTM captures long-term dependencies and global information(Shen et al., 2018).

The **Attention Layer** uses a location-sensitive attention mechanism to learn alignments between the encoder and decoder outputs. The attention mechanism computes a context vector for each decoder time step by taking a weighted sum of the encoder outputs, where the weights are determined by an alignment score function that considers both the previous alignment and the current decoder state. The location-sensitive attention helps the model avoid repeating or skipping words by using a cumulative alignment vector that tracks how much attention has been paid to each encoder time step(Shen et al., 2018).

The **Decoder Layer** applies two LSTM layers with 1024 units each, followed by a linear projection layer that predicts a Mel spectrogram frame at each time step. The decoder takes as input the previous predicted frame, the context vector from the attention layer, and a speaker embedding vector (if the multi-speaker mode is enabled). The decoder generates a sequence of Mel spectrogram frames that represent the acoustic features of the speech signal(Shen et al., 2018).

The **Post-net Layer** applies a CNN with 5 layers, each with 512 filters and kernel size 5, followed by a linear projection layer that adds a residual to the decoder output. The post-net improves the quality of the Mel spectrogram by refining its details and smoothing its artifacts.(Shen et al., 2018)

Overall, our proposed solution for acoustic modeling using RNN-based Tacotron 2 provides an effective and efficient method for generating high-quality Afaan Oromo speech from text.

4.5 Flow-Based Generative Model

Our proposed solution uses a flow-based generative model as a Vocoder. Specifically, we utilized WaveGlow and HIFI-GAN model with Tacotron 2 model for our Afaan Oromo TTS system

4.5.1 WaveGlow Model

The WaveGlow model is a flow-based generative model that generates audio samples from Gaussian distribution using Mel-spectrogram conditioning (Figure below).

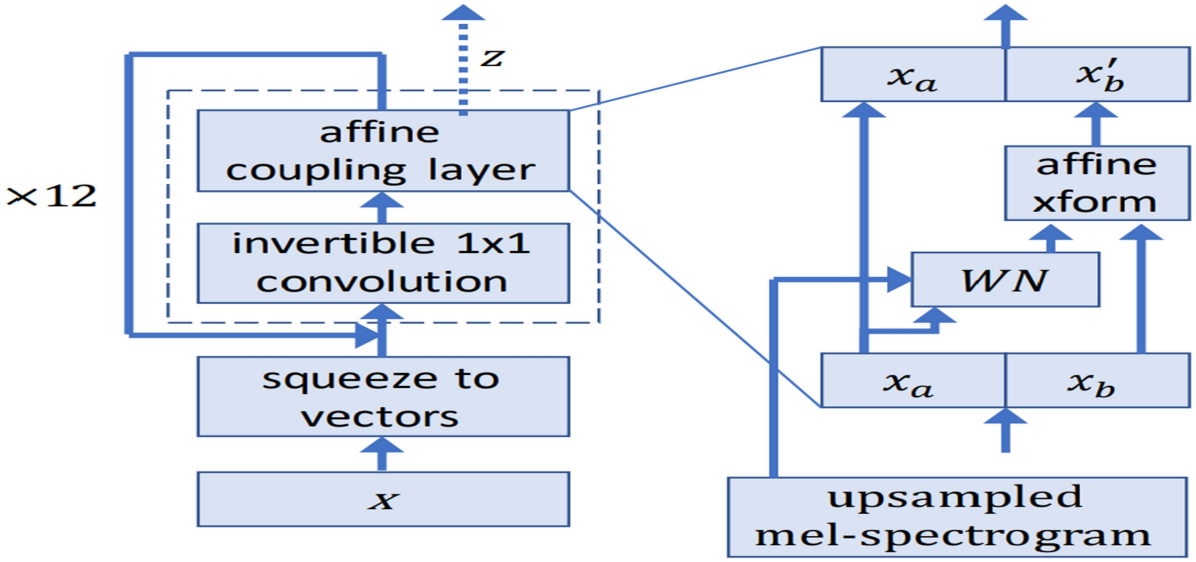


Figure 16: Architecture of WaveGlow (Prenger et al., 2019)

WaveGlow consists of the following layers(Prenger et al., 2019):

The **Squeeze Layer** takes an input audio waveform of shape (B, T), where B is the batch size and T is the number of time steps and reshapes it into (B, C, T'), where C is the number of channels and T' is the reduced number of time steps. For example, if C=4, then T'=T/4. This means that each time step in the output contains 4 samples from the input. The squeeze layer helps to reduce the dimensionality of the input and makes it easier for the model to process.

The **Coupling Layer** splits the input into two parts along the channel dimension: x1 and x2. Then it applies a neural network called WaveNet (WN) to x1 and adds the output to x2. The output of the coupling layer is the concatenation of x1 and x2. The coupling layer learns to transform the input distribution into a simpler one that can be easily sampled. The WaveNet network consists of several layers of dilated convolutions that capture long-range dependencies in the input.

The **Invertible 1x1 Convolution Layer** applies a linear transformation to the input that permutes the order of the channels. The transformation matrix is initialized randomly and learned during training. The invertible 1x1 convolution layer helps to mix the information across

different channels and avoid unwanted correlations. The transformation is invertible, meaning that it can be reversed by applying its inverse matrix.

The **Split Layer** outputs some of the channels as latent variables and passes the rest to the next layer. The latent variables are used to condition the generation process on the Mel spectrogram. The split layer reduces the dimensionality of the input and makes it easier for the model to sample from a simple distribution(Prenger et al., 2019).

During training, the model learns to transform the dataset distribution into spherical Gaussian distribution through a series of flows. One step of a flow consists of an invertible convolution, followed by a modified WaveNet architecture that serves as an affine coupling layer. During inference, the network is inverted and audio samples are generated from the Gaussian distribution. NVIDIA[4] implementation uses 512 residual channels in the coupling layer(Prenger et al., 2019).

4.5.2 HIFI-GAN

HiFi-GAN is a generative adversarial network-based model. Specifically, it is a type of autoregressive model that generates high-fidelity audio waveforms by transforming a noise source using a series of invertible transformations or "flows". Generative adversarial networks (GANs) to produce raw waveforms. The HiFi-GAN model achieves high-fidelity speech synthesis while also being effective. Because speech audio is made up of sinusoidal signals with different periods, the authors show how modeling periodic patterns in audio is essential for improving sample quality(Kong et al., 2020).

The HiFi-GAN model takes in a sequence of audio features, typically Mel-spectrogram or MFCC coefficients, and generates a corresponding waveform. It consists of two main components: a generator and a discriminator(Kong et al., 2020).

The generator is a flow-based model that learns to map a simple, known distribution (e.g., Gaussian noise) to a more complex distribution that captures the underlying structure of the audio data. It consists of multiple invertible blocks, each of which contains several coupling layers that transform the input noise into a more complex signal. These coupling layers learn to

[4] https://github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/SpeechSynthesis/Tacotron2/waveglow

split the input signal into two parts and transform one part while leaving the other unchanged(Kong et al., 2020).

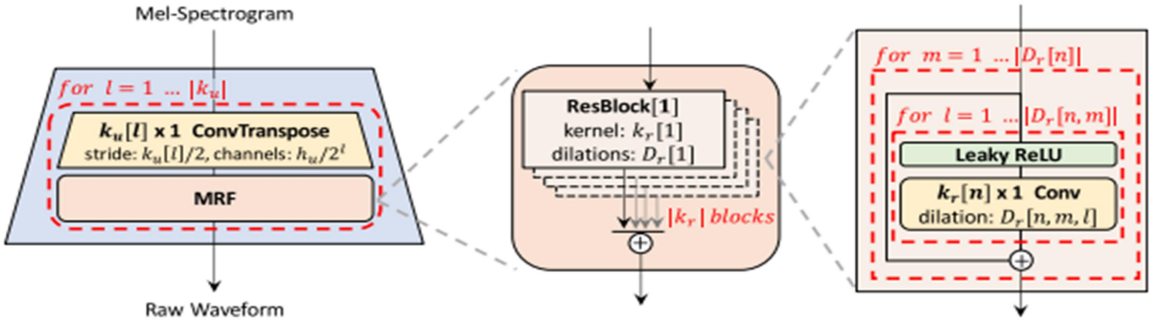


Figure 17: the architecture of generator(Mathieu et al., 2015)

The generator upsamples Mel-spectrograms up to |ku| times to match the temporal resolution of raw waveforms. An MRF module adds features from |kr| residual blocks of different kernel sizes and dilation rates. Lastly, the n-th residual block with kernel size kr[n] and dilation rates Dr[n] in an MRF module is depicted(Kong et al., 2020). The discriminator is a convolutional neural network that takes in an audio waveform and outputs a probability indicating whether the waveform is real or fake. The discriminator is trained to distinguish between real audio waveforms and waveforms generated by the generator(Kong et al., 2020).

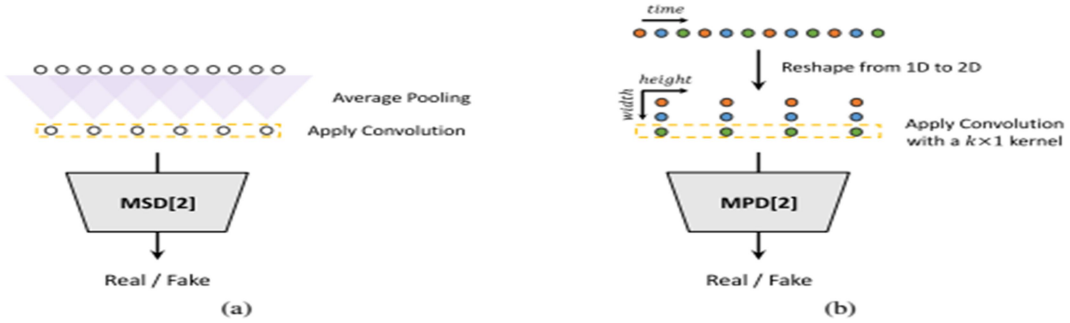


Figure 18: (a) The second sub-discriminator of MSD(Rithesh, 2019). (b) The second sub-discriminator of MPD with period 3(Kong et al., 2020)

The authors propose a multi-period discriminator (MPD) and a multi-scale discriminator (MSD) to achieve high-fidelity speech synthesis. MPD consists of several sub-discriminators that capture different implicit structures by looking at different parts of the input audio. Each sub-discriminator only accepts equally spaced samples of the audio with a period of [2, 3, 5, 7, 11]. The 1D raw audio of length T is first reshaped into 2D data of height T/p and width p, and 2D convolutions are applied to the reshaped data in every convolutional layer of MPD. Each sub-

discriminator is a stack of stridden convolutional layers with leaky rectified linear unit (ReLU) activation. Weight normalization is applied to MPD. The MSD evaluates audio samples at different levels consecutively to capture consecutive patterns and long-term dependencies. The MPD and MSD are used to capture the periodic signals and long-term dependencies of the input audio, respectively, to achieve high-fidelity speech synthesis(Kong et al., 2020).

The HiFi-GAN model is trained using adversarial training, where the generator and discriminator are trained simultaneously. The generator is trained to generate realistic audio waveforms that can fool the discriminator, while the discriminator is trained to correctly distinguish between real and fake waveforms. The HiFi-GAN model has several advantages over traditional vocoders. It can generate high-quality audio waveforms with a wide range of frequency content, including high-frequency harmonics(Kong et al., 2020).

4.6 Speech Synthesis Phase

The Mel spectrogram is a 2D array of numbers that represents the frequency and intensity of sound over time. Each row corresponds to a frequency bin and each column corresponds to a time step(Roberts, 2020). The value at each cell indicates how loud the sound is at that frequency and time. To feed the Mel spectrogram to WaveGlow or HIFI-GAN, we need to reshape it into a 1D vector of numbers. We can do this by flattening the 2D array into a 1D array(Roberts, 2020). For example, if the Mel spectrogram has shape (M, N), where M is the number of frequency bins and N is the number of time steps, we can flatten it into a 1D array of shapes (M*N).

Then, we can concatenate the flattened Mel spectrogram with a random noise vector of the same length. The random noise vector is sampled from a Gaussian distribution with a mean zero and standard deviation of one. This helps to add some variation and diversity to the generated speech.

4.7 Conceptual Design

Conceptual design has the root word “concept,” which describes the idea and intention behind the design. This is contrasted by “execution”, which is the implementation and shape that a design ultimately takes(Levanier, 2020). The purpose of conceptual design is to give visual shape to an idea. It gives a clear idea of what the product can do, and its intended use and considers the user's points of view hence actual results are easier to achieve.

4.7.1 Proposed Conceptual Framework

We performed a lot of image filtering and preprocessing before the actual text recognition was performed. This was to make both our Tesseract-OCR algorithm and TTS work as efficiently and effectively as possible. The figure below provides a flow of the software designed.

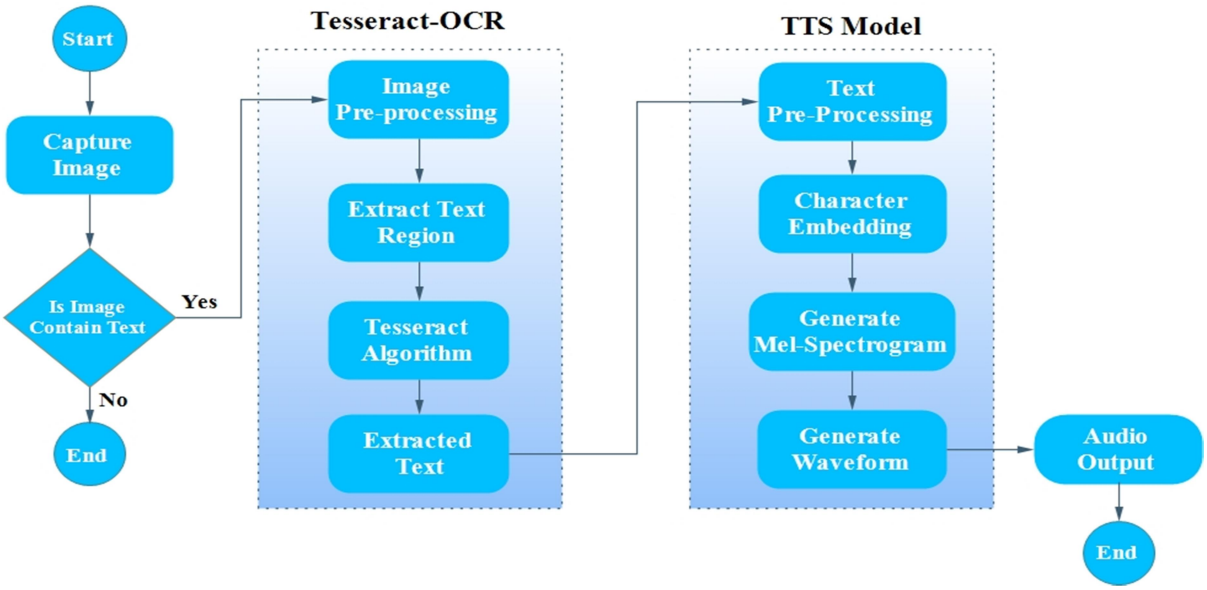


Figure 19: proposed conceptual framework

If the user clicked the button to read text According to our conceptual design of the device, we fed each captured image text block region into a section of code that formatted the region properly for Tesseract-OCR. After filtering and preprocessing, we executed the Tesseract-OCR algorithm using provided text block image and extracted the text from image. Upon determining the text within each region, we performed text preprocessing as the first step in the text-to-speech pipeline. The preprocessed text is then fed into the Tacotron2 model to generate a Mel-spectrogram which is then converted into audio using vocoders to notify the user about the content of the text through audio.

CHAPTER FIVE

5. IMPLEMENTATION OF TESSERACT-OCR AND AFAAN OROMO TEXT TO SPEECH

In this section, we provide a detailed explanation of the implementation process, including the tools used for implementation, data preparation, data preprocessing, feature extraction, and model training. Before training the model, we first prepared, preprocessed, and extracted features from our dataset. Finally, we trained the acoustic feature prediction models and presented the training progress for each model.

5.1 General Purpose Used Tools

5.1.1 Software Tools

Python 3.9: is the latest version of the Python programming language released in 2020. It comes with various new features, improvements, and bug fixes. In our study, we used Python programming language for implementing the data pre-processing, Tesseract-OCR, Tacotron2, WaveGlow, and HiFi-GAN models. Python is a widely used language in the field of deep learning and has a large number of libraries and frameworks available for machine learning tasks.

Anaconda Navigator: Anaconda Navigator was the platform we used in our study to manage our Python environment and packages. To construct and manage virtual environments, install packages, and launch Jupyter notebooks, Anaconda Navigator offers a user-friendly interface. Additionally, it contains several helpful software programs and tools for data science and machine learning, including NumPy, pandas, TensorFlow, and PyTorch. Setting up and administering a Python environment for use in data analysis and machine learning is made simpler by Anaconda Navigator.

Google Colaboratory: Google Colaboratory, a free cloud-based platform for executing and creating machine learning models, was employed in our work. This platform enables collaboration and notebook sharing with others, as well as access to a GPU for training deep learning models. Popular machine learning libraries like PyTorch, TensorFlow, and scikit-learn are also pre-installed, making it simple to use machine learning algorithms without the requirement for local installation.

Matplotlib[3.1.3]: Matplotlib was used for creating visualizations of the training progress of our deep learning models. Specifically, it was used to generate plots of the loss and accuracy values during training, which helped us to monitor the performance of the models and identify any potential issues that needed to be addressed. Matplotlib is a widely-used Python library for creating high-quality 2D plots and graphs, and it provides a range of tools for customizing visualizations and making them more informative.

5.1.2 Hardware Tools

Personal Computer: The personal computer used in our study served as the primary development and experimentation environment for the implementation of the models and algorithms. It was used for coding, data preparation and analysis, model training and evaluation, and result visualization. The computer's hardware specifications and software tools allowed us to efficiently conduct our research tasks while meeting the system requirements of the deep learning frameworks used.

Graphics Processing Unit (GPU) [NVIDIA Tesla T4]**:** In this study, we accelerated the training of deep learning models using a graphics processing unit (GPU). Due to their parallel computing capabilities, GPUs are particularly well suited for deep learning matrix calculations. We were able to dramatically shorten the time required to train our models by using a GPU, which made it easier for us to test out various architectures and hyperparameters.

Power Supply: The power supply for the graphics processing unit (GPU) is an essential component in our study as it provides the necessary power to run the GPU. Since the GPU is a high-performance computing component that requires a significant amount of power, a power supply unit (PSU) with sufficient wattage and efficiency is necessary.

5.2 Data Collection Tools

Microsoft Office[2016]**:** Microsoft Office is a collection of productivity software tools used for a variety of tasks, including word processing, spreadsheet creation, presentation creation, and database management. In our work, we created a speech corpus using Microsoft Excel, a program that is a part of the Microsoft Office suite. To make it simple to handle and analyze the data, we utilized Excel to store the audio path and transcription pairs for each speech sample in a structured way.

Samsung A70 Samsung A70 is a smartphone used for audio recording purposes in our study. The phone is equipped with high-quality microphones that can capture clear and high-fidelity audio recordings

5.3 Data Pre-Processing Tools

Data pre-processing is the collection of procedures and methods used to modify raw data into a more useful and effective form. A variety of tasks, including data cleaning and formatting, the elimination of superfluous or irrelevant data, the handling of missing values, scaling or normalizing the data, and others, may be involved in this.

5.3.1 Image Pre-processing Tools and Libraries

OpenCV[4.5.3]: OpenCV is an open-source computer vision library that we have used for image processing tasks such as cropping, resizing, and color-space conversion.

PIL[8.4.0] (Python Imaging Library): PIL is a library that adds support for opening, manipulating, and saving many different image file formats. We have used it for image resizing and format conversion.

5.3.2 Text Pre-Processing Tools and Libraries

NLTK: NLTK (Natural Language Toolkit): It is a popular library used for natural language processing tasks such as tokenization, stemming, and lemmatization.

5.3.3 Audio Pre-Processing Tools and Libraries

Audacity[3.2.5]: Audacity is a free and open-source digital audio editor that we used to clean and edit our audio recordings. We used it to remove background noise, adjust volume levels, and remove unwanted parts of the recordings.

Librosa[0.10.0]: Librosa is a Python library for analyzing and processing audio signals. We used it to extract features such as the Mel Spectrogram, Mel Frequency Cepstral Coefficients (MFCCs), and pitch from our audio files.

Resampy[0.3.1]: Resampy is a Python library for efficient sample rate conversion of audio signals.

SciPy[1.8.0]: SciPy is a scientific computing library for Python. We used it for various audio signal processing tasks such as filtering, resampling, and segmenting audio files.

NumPy[1.24.2]: NumPy is a Python library for numerical computing. We used it for efficient manipulation of arrays and matrices, which is particularly useful for working with audio data.

5.4 Implementation of Tesseract-OCR

5.4.1 Libraries and Package Used

Tesseract-OCR[4.1.1]: Tesseract OCR engine which is capable of recognizing various image formats and extracting text from them.

Libtesseract-dev[4.1.1]: It is the development files of Tesseract-OCR which contains libraries, headers, and documentation.

Pytesseract[0.3.10]: It is a Python wrapper for Tesseract-OCR that allows the user to easily integrate OCR functionality into their Python scripts.

Pillow[8.4.0]: It is a Python Imaging Library that is used to open, manipulate, and save many different image file formats.

Tesseract[0.1.3]: It is a command-line tool for Tesseract-OCR that allows the user to process images and output recognized text from the command line.

Levenshtein[0.21.0]: It is a Python library that implements the Levenshtein distance algorithm, which is used to measure the difference between two sequences of strings. In the context of OCR, it can be used to correct OCR errors and improve the accuracy of recognized text.

NumPy: NumPy can be used to resize, reshape, or manipulate the pixel values of the image arrays, making them suitable for Tesseract OCR processing.

5.4.2 Image Pre-Processing

We used a variety of image pre-processing methods in our implementation to boost Tesseract-OCR's performance. Binarization, Deskewing, noise removal, and image resizing are some of these methods. By establishing a threshold value to distinguish the text from the backdrop, Binarization was employed to turn the image into a binary representation. If the image was crooked or slanted, Deskewing was used to make it straight. Any unwelcome noise or artifacts in the image that can obstruct the OCR process were removed using noise removal. To standardize the image size and resolution and increase the speed and accuracy of OCR, image scaling was lastly used.

Algorithm Used for Image Pre-processing:

Table 4: Image Pre-Processing Algorithm

Steps	Pre-Processing
Step 1:	Load the image using OpenCV (cv2.imread) and store it
Step 2:	Convert the image to grayscale
Step 3:	Perform Binarization on the grayscale image
Step 4:	Perform deskewing to straighten the text in the image: ➢ Find the coordinates of non-zero (white) pixels in the binary image ➢ Calculate the minimum area rectangle that fits the non-zero coordinates ➢ Extract the angle of rotation from the rectangle ➢ Adjust the angle if necessary to ensure it is within the range of -45 to 45 degrees. ➢ Determine the center of the image. ➢ Generate a rotation matrix with the center, angle, and scale factors. ➢ Apply the rotation to the binary image using cv2.warpAffine
Step 5:	Perform noise removal using a Gaussian filter with a kernel size of 3: ➢ Define the size of the kernel (matrix). ➢ Calculate the values of the Gaussian kernel using the Gaussian distribution ➢ Normalize the values of the kernel ➢ Convolve the image with the kernel by sliding it over the image and multiplying each pixel value by its corresponding weight in the kernel. ➢ Sum up all the weighted pixel values in the kernel and assign this value to the center pixel of the kernel.
Step 6:	Resize the image to the desired size (600 x 800): ➢ Get the current sizes of the image ➢ Calculate the new size based on the desired size ➢ Resize the image with new width and height.
Step 7:	Save the pre-processed image
Step 8:	Return the pre-processed image

5.4.3 Text Extraction Phase

To extract text from an image using Tesseract OCR in our study, we first pre-processed the image using techniques such as Binarization, Deskewing, noise removal, and resizing. We then used the pytesseract library in Python, which is a wrapper for the Tesseract OCR engine, to recognize the text in the pre-processed image. The recognized text was then extracted and returned for further use in our text-to-speech synthesis model.

```
def extract_text_from_image(image_name):
  # Load image
  image = cv2.imread(image_name)
  # pre-process uploaded image
  image = image_preprocess(image)
  recognized_text = pytesseract.image_to_string(image)
  # post-process text by replace newline with space
  # help us to make one line sentence for TTS model
  data = recognized_text.replace('\n', ' ')
  return data
```

Figure 20: Extracting Text from an image using Tesseract-OCR

The figure below shows an image containing Afaan Oromoo Text and extracted text from using the Tesseract-OCR system.

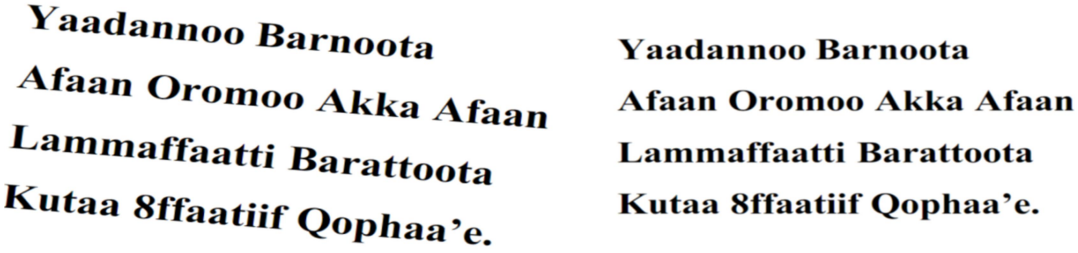


Figure 21: Input Images A) Skewed Image B) Blurred Image

```
extracted = extract_text_from_image('image.jpg')
print(extracted)
```

```
Yaadannoo Barnoota Afaan Oromoo Akka Afaan Lammaffaatti Barattoota  Kutaa 8ffaatiif Qophaa'e.
```

```
extracted = extract_text_from_image('skewed_image.jpg')
print(extracted)
```

```
Yaadannoo Barnoota Afaan Oromoo Akka Afaan Lammaffaatt; Barattoota Kutaa 8ffaatiif Qophaa'e.
```

Figure 22: Tesseract-OCR output of blurred and skewed images

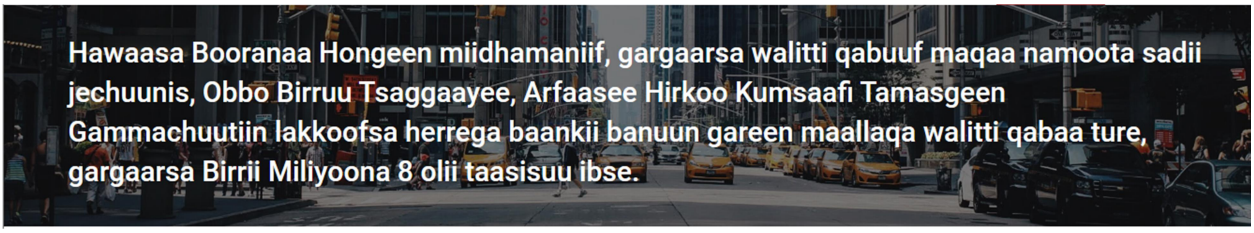
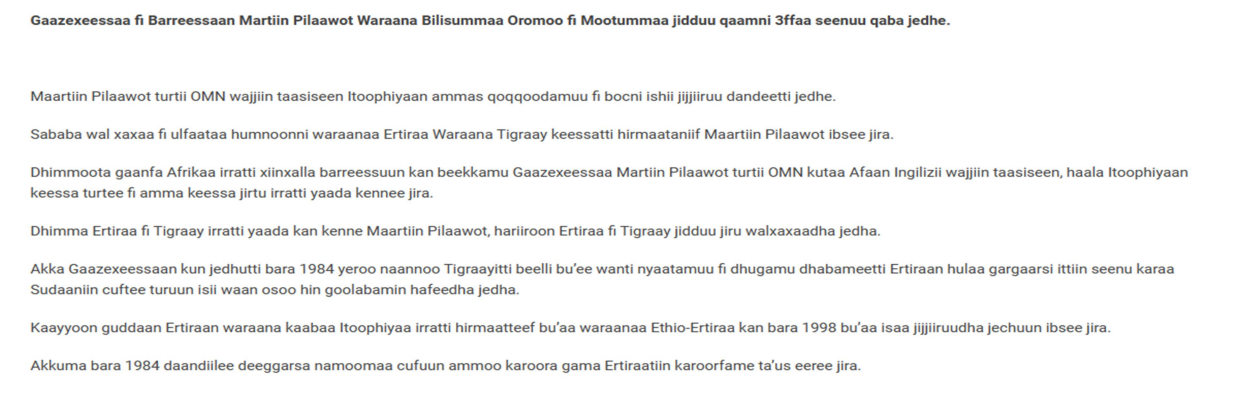


Figure 23: input image containing text with background image

```
extracted = extract_text_from_image('image_5.jpg')
print(extracted)
```

```
Hawaasa Booranaa Hongeen miidhamaniif, gargaarsa walitti qabuuf maqaa namoota sadii
jechuunis, Obbo Birruu Tsaggaayee, Arfaasee Hirkoo Kumsaafi Tamasgeen
Gammachuutiin lakkoofsa herrega. baankii banuun gareen maallaqa walitti qabaa ture,
gargaarsa Birrii Miliyoona 8 olii taasisuu ibse.
```

Figure 24: output image containing text with background image



Gaazexeessaa fi Barreessaan Martiin Pilaawot Waraana Bilisummaa Oromoo fi Mootummaa jidduu qaamni 3ffaa seenuu qaba jedhe.

Maartiin Pilaawot turtii OMN wajjiin taasiseen Itoophiyaan ammas qoqqoodamuu fi bocni ishii jijjiiruu dandeetti jedhe.

Sababa wal xaxaa fi ulfaataa humnoonni waraanaa Ertiraa Waraana Tigraay keessatti hirmaataniif Maartiin Pilaawot ibsee jira.

Dhimmoota gaanfa Afrikaa irratti xiinxalla barreessuun kan beekkamu Gaazexeessaa Martiin Pilaawot turtii OMN kutaa Afaan Ingilizii wajjiin taasiseen, haala Itoophiyaan keessa turtee fi amma keessa jirtu irratti yaada kennee jira.

Dhimma Ertiraa fi Tigraay irratti yaada kan kenne Maartiin Pilaawot, hariiroon Ertiraa fi Tigraay jidduu jiru walxaxaadha jedha.

Akka Gaazexeessaan kun jedhutti bara 1984 yeroo naannoo Tigraayitti beelli bu'ee wanti nyaatamuu fi dhugamu dhabameetti Ertiraan hulaa gargaarsi ittiin seenu karaa Sudaaniin cuftee turuun isii waan osoo hin goolabamin hafeedha jedha.

Kaayyoon guddaan Ertiraan waraana kaabaa Itoophiyaa irratti hirmaatteef bu'aa waraanaa Ethio-Ertiraa kan bara 1998 bu'aa isaa jijjiiruudha jechuun ibsee jira.

Akkuma bara 1984 daandiilee deeggarsa namoomaa cufuun ammoo karoora gama Ertiraatiin karoorfame ta'us eeree jira.

Figure 25: blurred input image containing text

```
extracted = extract_text_from_image('image_1.png')
print(extracted)
```

```
Gaazexeessaa fi Barreessaan Martiin Pilaawot Waraana Bilisummaa Oromoo fi Mootummaa jidduu qaamni 3ffaa seenuu qaba jedhe.

Maartiin Pilaawot turtii OMN wajjiin taasiseen Itoophiyaan ammas qoqqoodamuu fi bocni ishiijijiiruu dandeetti jedhe.
Sababa wal xaxaa fi ulfaataa humnoonni waraanaa Ertiraa Waraana Tigraay keessatti hirmaataniif Maartiin Pilaawot ibsee jira.

Dhimmoota gaanfa Afrika irratti xiinxalla barreessuun kan beekkamu Gaazexeessaa Martiin Pilaawot turtii OMN kutaa Afaan Ingilizii waijiin taasiseen, haala Itoophiyaan
keessa turtee fi amma keessa jirtu irratti yaada kennee jira.

Dhimma Ertiraa fi Tigray irratti yada kan kenne Maartiin Pilaawot, hariiroon Ertiraa fi Tigraay jidduu jiru walxaxaadha jedha.

'Akka Gaazexeessaan kun jedhutti bara 1984 yeroo naannoo Tigraayitti beelli bu'ee wanti nyaatamuu fi dhugamu dhabameetti Ertiraan hulaa gargaarsi ittiin seenu kara
Sudaaniin cuftee turuun isii waan osoo hin goolabamin hafeedha jedha.

Kaayyoon guddaan Ertiraan waraana kaabaa ltoophiyaa irratti hirmaatteef bu'aa waraanaa Ethio-Ertiraa kan bara 1998 bu'aa isa jijiruudha jechuun ibsee jira.

'Akkuma bara 1984 daandiilee deeggarsa namoomaa cufuun ammoo karoora gama Ertiraatiin karoorfame ta'us eeree jira.
```

Figure 26: output of blurred input image containing text

5.5 Training Text-to-Speech Model

5.5.1 TTS Modeling Framework and Libraries

In this study, we trained and implemented the Text-to-Speech (TTS) model using a variety of frameworks and tools. PyTorch and Tensorflow were the primary deep-learning tools we used to build and train the TTS model. Additionally, we accelerated the GPU during training using CUDA, a parallel computing platform. NumPy was utilized for numerical computations, and Matplotlib was used for visualization.

PyTorch: PyTorch is an open-source machine learning framework that is used for developing and training neural network models. It provides a simple and efficient way to build deep learning models and supports both CPU and GPU computation.

TensorFlow: TensorFlow is another open-source machine learning framework that is widely used for developing and training deep learning models. It provides a high-level API for building and training neural networks and supports distributed computing.

CUDA: CUDA is a parallel computing platform and application programming interface (API) developed by NVIDIA for general-purpose computing on GPUs. It allows developers to harness the power of the GPU for deep learning computations, enabling faster model training.

Matplotlib: Matplotlib is a plotting library for Python that provides a wide range of 2D plotting functions. It is used for creating visualizations of data and modeling training progress.

NumPy: NumPy is a Python library used for scientific computing. It provides support for large, multi-dimensional arrays and matrices, as well as a large collection of mathematical functions for manipulating these arrays.

Inflect: inflect is a Python library used for converting numbers to their English word equivalents. It is used in our TTS implementation to convert the numeric values of durations and frequencies to their spoken word equivalents.

Librosa: Librosa is a Python library used for analyzing and processing audio signals. It provides a wide range of functions for performing audio analysis, including feature extraction and time-frequency analysis.

SciPy: SciPy is another Python library used for scientific computing. It provides support for a wide range of mathematical functions and algorithms, including signal processing and optimization.

DLLogger: DLLogger is a logging library developed by NVIDIA for deep learning model training. It provides a simple and efficient way to log training progress and metrics.

Tensorboard: Tensorboard is a visualization tool developed by TensorFlow for monitoring and visualizing deep learning model training. It provides a wide range of visualizations, including loss and accuracy plots, histogram distributions, and model architecture visualizations.

5.5.2 Data Preparation

In the data preparation phase, we uploaded a dataset containing audio paths and text transcriptions, which were stored in a text file with the .txt file extension. We used the "|" delimiter to separate the audio path and text transcription in each line of the file. We loaded the dataset using the PyTorch data loader, which allowed us to efficiently load and preprocess the data.

5.5.3 Loading Text and Audio Dataset

To configure the dataset parameters, we used the argparse library, and we defined the dataset parameters as a separate argument group. We specified the training and validation filelists using the --training-files and --validation-files arguments, respectively. We also provided the option to load Mel-spectrograms from disk instead of computing them on the fly using the --load-mel-from-disk flag. Finally, we defined the type of text cleaners for input text using the --text-cleaners argument.

```
# dataset parameters
dataset = parser.add_argument_group('dataset parameters')
dataset.add_argument('--load-mel-from-disk', action='store_true',
                     help='Loads mel spectrograms from disk instead of computing them on the fly')
dataset.add_argument('--training-files',
                     default='filelists/ljs_audio_text_train_filelist.txt',
                     type=str, help='Path to training filelist')
dataset.add_argument('--validation-files',
                     default='filelists/ljs_audio_text_val_filelist.txt',
                     type=str, help='Path to validation filelist')
dataset.add_argument('--text-cleaners', nargs='*',
                     default=['AO_cleaners'], type=str,
                     help='Type of text cleaners for input text')
```

Figure 27: defining dataset parameters

For the actual data loading and processing, we used the TextMelLoader class, which is a subclass of the PyTorch Dataset class. This class loads the audio and text pairs, normalizes the text and converts them to sequences of one-hot vectors, and computes Mel-spectrograms from audio files. We passed the dataset path, audio paths and text, and dataset parameters as arguments to the constructor of the TextMelLoader class.

Algorithm: To get Mel-Spectrogram Text Pair

Input: audio path (path to the audio file), text (input text)

Process: 1. Compute Mel-spectrogram from the audio file

2. Normalize and convert the input text to one-hot vectors

3. Calculate the length of the normalized text

Output: Return a tuple containing normalized text, Mel-spectrogram, and text length

Algorithm: To get Mel-Spectrogram

Input: audio path (path to the audio file)

Process: If Mel-spectrogram exists on disk for the given audio path:

Load the Mel-spectrogram from the disk

Else:

1. Load the audio waveform from the audio path

2. Normalize the audio waveform

3. Convert the normalized waveform to a Mel-spectrogram using Short-Time Fourier Transform (STFT)

Output: Return the Mel-spectrogram

Algorithm: to get Text

Input: text (input text)

Process: 1. Normalize the input text

2. Convert the normalized text into a sequence of one-hot vectors

Output: Return the sequence of one-hot vectors

5.5.4 Text Pre-Processing

One of the most important steps in training a Text-to-Speech (TTS) model is text data preparation. To make sure that the model can handle a variety of input text types, it entails cleaning and normalizing the text data. In our implementation, we utilized the use of several text-

cleaning techniques, including tokenization, text normalization, and expanding abbreviations. In addition, we applied word-to-number, word-to-ordinal, and word-to-currency conversion methods.

```
def AO_cleaners(text):
  '''Pipeline for Afaan Oromoo text, including number and abbreviation expansion.'''
  text = convert_to_ascii(text)
  text = lowercase(text)
  text = expand_numbers(text)
  text = expand_abbreviations(text)
  text = collapse_whitespace(text)
  return text
```

Figure 28: Afaan Oromoo Text Cleaners

Expanding Abbreviations: To make sure the text data is acceptable for the TTS model, we expanded abbreviations like "Obb." to "Obbo," "Dr." to "Doktar," and "m/b" to "mana barumsaa".

```
# List of (regular expression, replacement) pairs for abbreviations:
_abbreviations = [(re.compile('\\b%s\\.' % x[0], re.IGNORECASE), x[1]) for x in [
  ('add.', 'aaddee'),
  ('obb.', 'obbo'),
  ('dr.', 'dooktara'),
  ('prof.', 'profeesara'),
  ('k.k.f', 'kan kana fakkaatan'),
  ('f.k.n', 'fakeenyaaf'),
  ('hub.', 'hubadhu'),
  ('A.L.I', 'akka lakkoofsa itiyoophiyaa'),
  ('A.L.A', 'akka lakkoofsa awurooppaa'),
  ('m/murti', 'mana murtii'),
  ('w/d', 'waaree dura'),
  ('w/b', 'waaree booda'),
  ('w.k.f', 'waan kana fakkaatan'),
  ('bul.', 'bulchiinsa'),
  ('ykn', 'yookaan'),
  ('ABO', 'adda bilisumma oromoo'),
  ('KFO', 'kongiransii federalistii oromoo'),
  ('FDG', 'fincila diddaa garbummaa'),
  ('WBO', 'warraana bilisumma oromoo'),
  ('I/G', 'itti gaafatamaa'),
  ('DH.K.D', 'dhaloota kiriistoosin dura'),
  ('DH.K.B', 'dhaloota kiriistoosin booda'),
  ('hogg.', 'hooganaa'),
  ('h/bulaa', 'horsiisee bulaa'),
  ('q/bulaa', 'qonnaan bulaa'),
  ('Qar.', 'qarshii'),
  ('m/ministeeraa', 'muummee ministeeraa'),
  ('m/b', 'mana barumsaa'),
]]


def expand_abbreviations(text):
  for regex, replacement in _abbreviations:
    text = re.sub(regex, replacement, text)
  return text
```

Figure 29: implementation of expanding abbreviations

Text Normalization: To make sure the input text is in a uniform format, we applied a variety of text normalization algorithms. This entails changing all text to lowercase, expanding numbers, and substituting spaces for any non-alphabetic letters.

```
def expand_numbers(text):
  return normalize_numbers(text)

def lowercase(text):
  return text.lower()

def collapse_whitespace(text):
  return re.sub(_whitespace_re, ' ', text)
```

Figure 30: implementing text normalization

Numbers-to-Word: We translated numerical values from the input text into their corresponding word forms. For instance, "123" became "dhibba tokkoofi digdamii sadi." This technique allows the TTS model to handle numerical input text more effectively.

```
def replace_numbers_with_words(string):
    words = []
    for word in string.split():
        if any(char.isdigit() for char in word):
            word = ''.join(filter(lambda x: x.isdigit() or x == '.', word))
            if '.' in word:
                integer_part, decimal_part = word.split('.')
                integer_part_words = number_to_words(int(integer_part))
                decimal_part_words = number_to_words(int(decimal_part))
                words.append(integer_part_words + "-tuqaa-" + decimal_part_words)
            else:
                words.append(number_to_words(int(word)))
        else:
            words.append(word)
    return " ".join(words)
```

Figure 31: Expanding number to words

Currency-to-Word: We converted the input text's currency symbols (like $) to their corresponding word representations (like dollars). By using this method, the TTS model is better able to handle currency input text.

```
def _expand_dollars(m):
  match = m.group(1)
  parts = match.split('.')
  if len(parts) > 2:
    return match + ' fi doolaaraa'  # Unexpected format
  dollars = int(parts[0]) if parts[0] else 0
  cents = int(parts[1]) if len(parts) > 1 and parts[1] else 0
  if dollars and cents:
    dollar_unit = 'doolaaraa' if dollars == 1 else 'fi doolaaraa'
    cent_unit = 'saantima' if cents == 1 else 'fi saantima'
    return '%s %s, %s %s' % (dollars, dollar_unit, cents, cent_unit)
  elif dollars:
    dollar_unit = 'doolaaraa' if dollars == 1 else 'fi doolaaraa'
    return '%s %s' % (dollars, dollar_unit)
  elif cents:
    cent_unit = 'saantima' if cents == 1 else 'fi saantima'
    return '%s %s' % (cents, cent_unit)
  else:
    return 'doolaaraa zeeroo'
```

Figure 32: expanding currency to word

Tokenization: We used character embedding to convert the input text into a sequence of vectors. To achieve this, we first tokenized the input text by splitting it into individual characters. This allowed us to create a mapping between each character and a corresponding vector, which the TTS model can easily process.

5.5.5 Audio Processing

These parameters are used to configure the audio processing functions used. The audio parameters declared are max-wav-value, sampling rate, filter length, hop-length, win-length, mel-min, and mel-fmax.

```
# audio parameters
    audio = parser.add_argument_group('audio parameters')
    audio.add_argument('--max-wav-value', default=32768.0, type=float,
                       help='Maximum audiowave value')
    audio.add_argument('--sampling-rate', default=22050, type=int,
                       help='Sampling rate')
    audio.add_argument('--filter-length', default=1024, type=int,
                       help='Filter length')
    audio.add_argument('--hop-length', default=256, type=int,
                       help='Hop (stride) length')
    audio.add_argument('--win-length', default=1024, type=int,
                       help='Window length')
    audio.add_argument('--mel-fmin', default=0.0, type=float,
                       help='Minimum mel frequency')
    audio.add_argument('--mel-fmax', default=8000.0, type=float,
                       help='Maximum mel frequency')
```

Figure 33: audio processing parameters

Audio Feature Extraction and Normalization: Using PyTorch and audio processing methods, the get_mel function creates a Mel-spectrogram of an audio file. First, it determines whether or not the Mel-spectrogram needs to be loaded from the disk. The function loads the audio file and determines whether it needs to be generated by examining the sampling rate. To add a batch dimension, the audio is next normalized and unsqueezed. A Mel-spectrogram is produced by passing the generated tensor through STFT using the stft.mel_spectrogram function. Before returning the Mel-spectrogram as a PyTorch tensor, the function first uses the squeeze function to eliminate the batch dimension.

```
def get_mel(self, filename):
        if not self.load_mel_from_disk:
            audio, sampling_rate = load_wav_to_torch(filename)
            if sampling_rate != self.stft.sampling_rate:
                raise ValueError("{} {} SR doesn't match target {} SR".format(
                    sampling_rate, self.stft.sampling_rate))
            audio_norm = audio / self.max_wav_value
            audio_norm = audio_norm.unsqueeze(0)
            audio_norm = torch.autograd.Variable(audio_norm, requires_grad=False)
            melspec = self.stft.mel_spectrogram(audio_norm)
            melspec = torch.squeeze(melspec, 0)
        else:
            melspec = torch.load(filename)
            assert melspec.size(0) == self.stft.n_mel_channels, (
                'Mel dimension mismatch: given {}, expected {}'.format(
                    melspec.size(0), self.stft.n_mel_channels))

        return melspec
```

Figure 34: audio normalization and Mel-Spectrogram generation

5.6 Implementation of the TTS Model

In the implementation of the Tacotron2 model, two types of training were conducted. The first type involved training the model from scratch using a custom dataset and data cleaners. This type of training is useful when there is no pre-existing model available that can be fine-tuned for a specific task. During this training, the model was trained on the entire dataset to learn the mapping between the input text and the corresponding speech output.

The second type of training was fine-tuning a pre-trained model on the English language for Afaan Oromoo. Fine-tuning is a technique that involves taking a pre-trained model and adapting it to a new task or domain by training it on a new dataset. In this case, the pre-trained model was already trained in the English language, and it was fine-tuned using the Afaan Oromoo dataset. Fine-tuning a pre-trained model can be an effective way to reduce the training time and improve the performance of the model on a new task.

5.6.1 Training Tacotron2 from Scratch

We trained a Tacotron2 model from scratch for Afaan Oromoo text-to-speech (TTS) synthesis. The training process involved 500 epochs, with each epoch consisting of 1566 iterations. We monitored the training and validation loss throughout the training process to assess the convergence and performance of the model. The validation loss, on the other hand, evaluates the model's performance on a separate validation set, indicating how well the model generalizes to unseen data. As the training progressed, we observed a gradual decrease in both the training and

validation loss. This decrease signifies that the model was learning and improving its ability to generate accurate and natural-sounding speech. The convergence of the loss values indicates that the model was effectively capturing the patterns and characteristics of Afaan Oromoo speech data.

After training for the specified number of epochs, we decided to stop the training when the training loss reached 0.2543270669421917 and the validation loss was 0.343282864005751. These loss values indicated that the model had achieved a satisfactory level of performance and further training may result in diminishing returns or overfitting to the training data. It is worth noting that the training loss and validation loss values alone do not provide a complete assessment of the model's quality. Additional evaluation metrics, such as subjective evaluations, can be employed to further evaluate the generated speech's perceptual quality and similarity to natural speech.

Figure 35: Tensorboard Scalar Training loss of Tacotron2

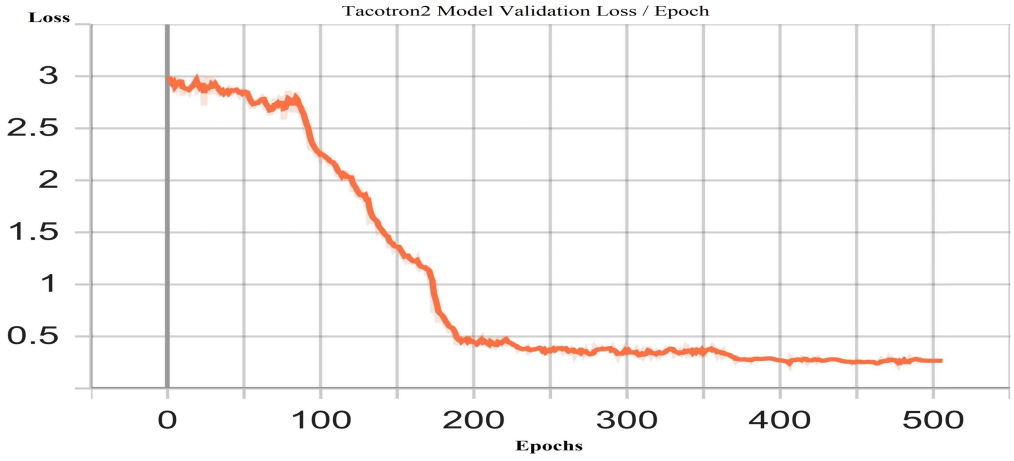


Figure 36: Validation Loss of Tacotron2 Model Training

5.6.2 Training of WaveGlow Model

We conducted training on WaveGlow, a neural vocoder, using our own Afaan Oromoo dataset. The training process involved training the model from scratch for a total of 497 epochs or 778,302 iterations. During the initial stages of training, the training loss was recorded as -2.4532. It's worth noting that the negative sign in the loss indicates that it is a log-likelihood loss, commonly used in probabilistic models. The log-likelihood loss is commonly used in probabilistic models to measure the discrepancy between the predicted distribution and the target distribution(Prenger et al., 2019). As the training progressed, the loss gradually decreased, indicating improved model performance.

To evaluate the model's performance, we calculated the validation loss at the end of each epoch. The validation loss is an important metric that indicates how well the model generalizes to unseen data. Throughout the training process, we monitored the scalar training and validation loss using Tensorboard. This visualization provided insights into the model's convergence and performance. To ensure model checkpointing and track progress, we saved the model every 10 epochs. However, at epoch 490, we noticed a significant increase in both the training and validation loss. This increase could indicate overfitting or convergence to a suboptimal solution. As a result, we decided to halt the training process and ignore the model saved after epoch 490.

At the point of stopping, the validation loss was recorded as -8.330729133012344, while the training loss was -9.478466033935547. It's important to note that the negative sign in the loss values is consistent with the log-likelihood loss used in WaveGlow. The training process for WaveGlow using our Afaan Oromoo dataset demonstrated significant progress, as evidenced by the decreasing loss. However, further investigation with more datasets and fine-tuning may be required to improve the model's performance and prevent overfitting. These findings provide valuable insights into the development and utilization of speech synthesis systems for Afaan Oromoo.

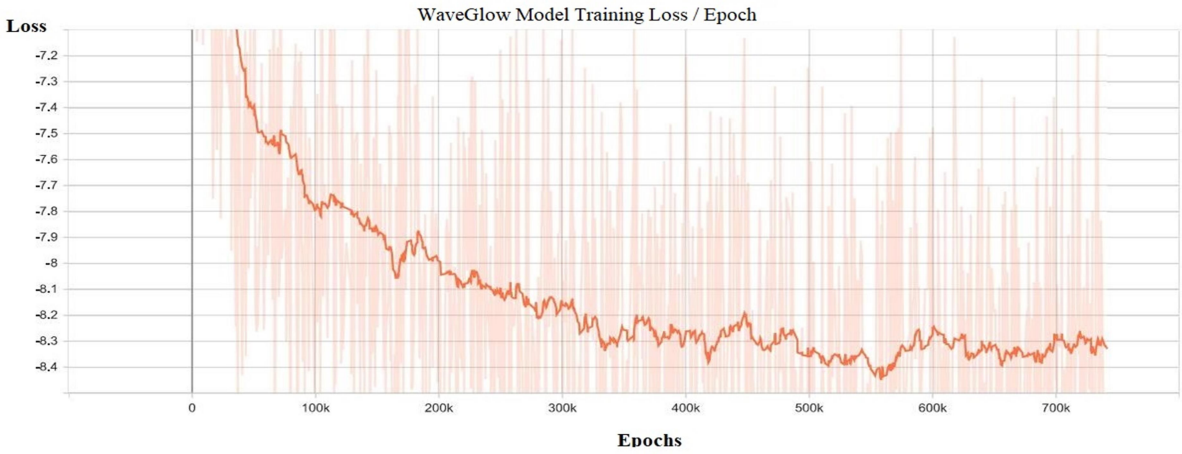


Figure 37: Tensorboard Scalar Training loss of WaveGlow Model

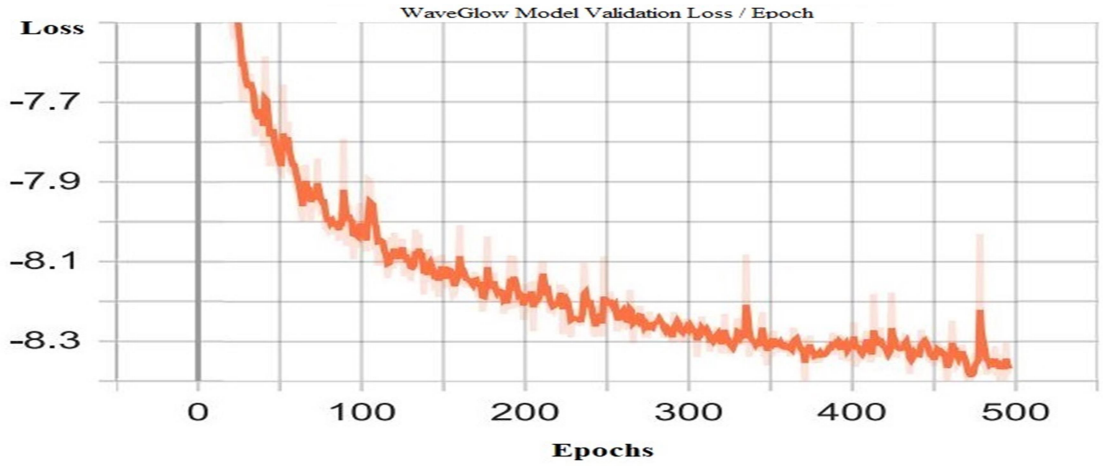


Figure 38: Tensorboard Scalar Validation Loss of WaveGlow Model

5.6.3 Testing Inference

the inference results of the Tacotron2 model trained from scratch for Afaan Oromoo TTS synthesis, combined with the WaveGlow vocoder, demonstrated the successful generation of high-quality and natural-sounding speech. The subjective evaluations affirmed the effectiveness of the models, indicating their potential for various applications requiring Afaan Oromoo speech synthesis.

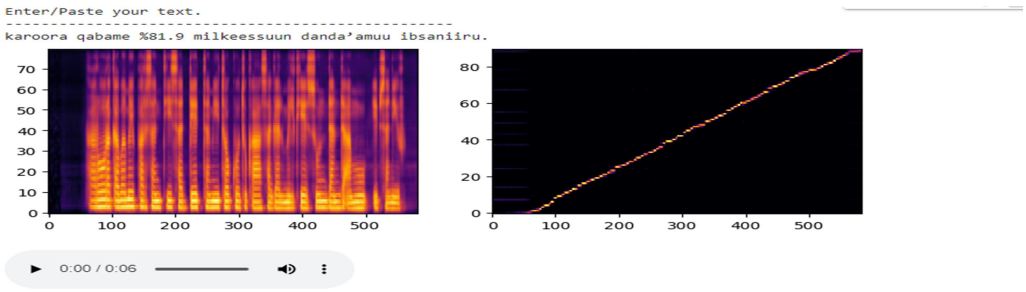


Figure 39: Tacotron2 and WaveGlow Inference 1

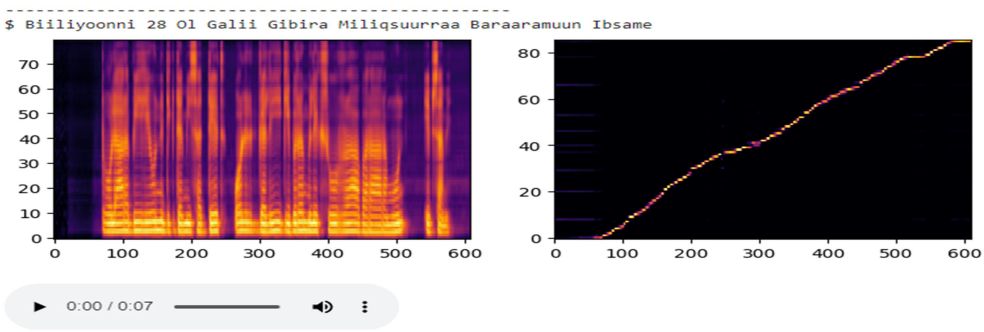


Figure 40: Tacotron2 and WaveGlow Inference 2

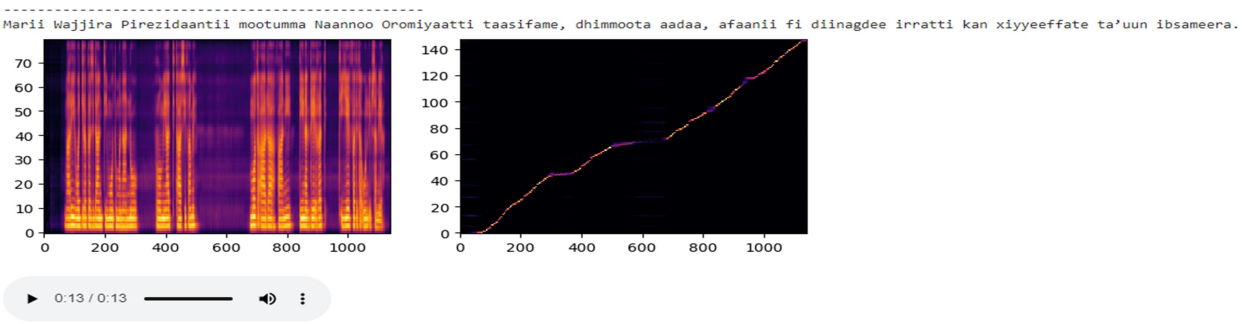


Figure 41: Tacotron2 and WaveGlow Inference 3

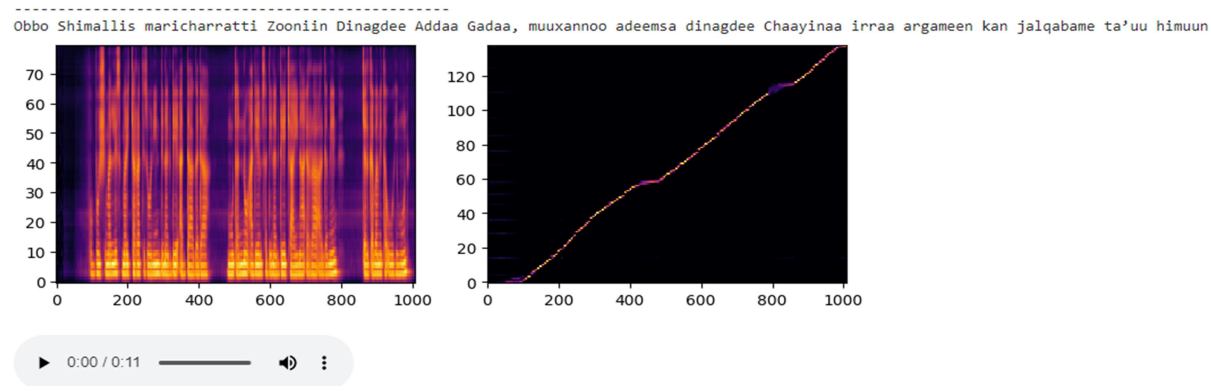


Figure 42: Tacotron2 and WaveGlow Inference 4

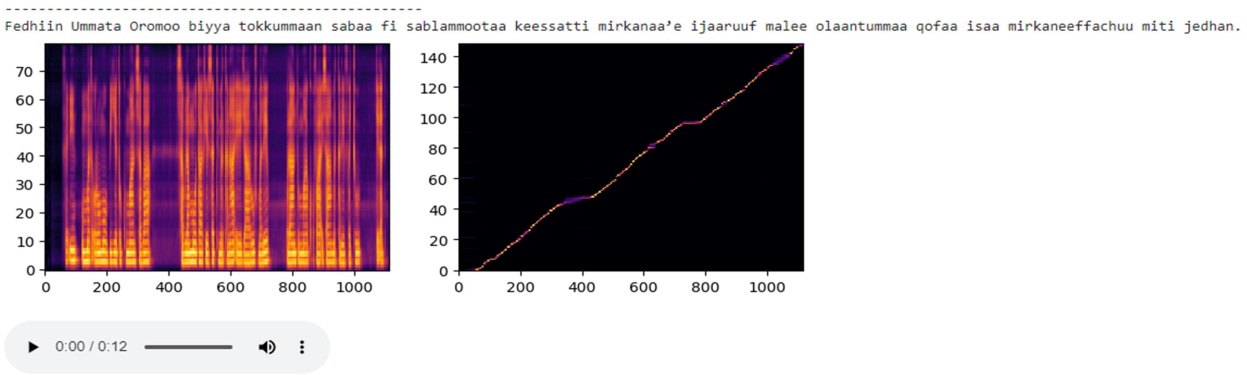


Figure 43: Tacotron2 and WaveGlow Inference 5

5.6.4 Fine Tuning Pre-Trained Tacotron2 Model

In the fine-tuning part, a pre-trained Tacotron2 model on the English language is loaded and trained using Afaan Oromoo dataset and our text cleaners. The batch size used for training is 30, and the pronunciation ARPAbet from the previous model is omitted since it does not work with the Afaan Oromoo language. The learning rate used is 3e-4, and the model is trained for 3000 iterations. The training is performed on Google Colab for 6 hours. The alignment of the model started changing after the 5th epoch, and at the 50th epoch, the validation loss became 0.259536. The model started generating almost natural sound by the 80th epoch (2642 iterations), with the validation loss reduced to 0.233419. The training was stopped at 90 epochs when the result was almost near to natural sound.

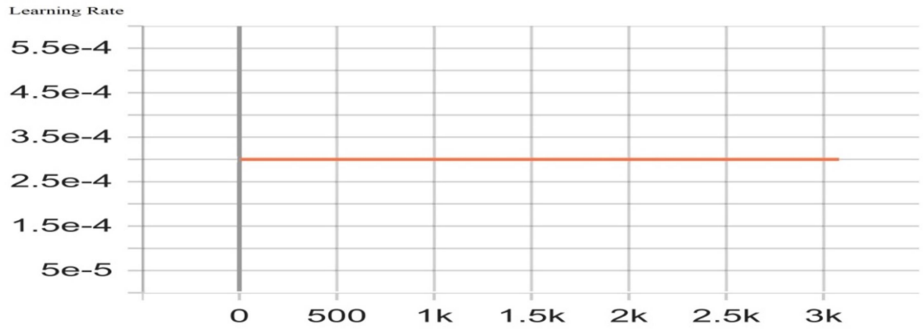


Figure 44: Tensorboard Scalar Graph of Fined-tuned Model Learning Rate

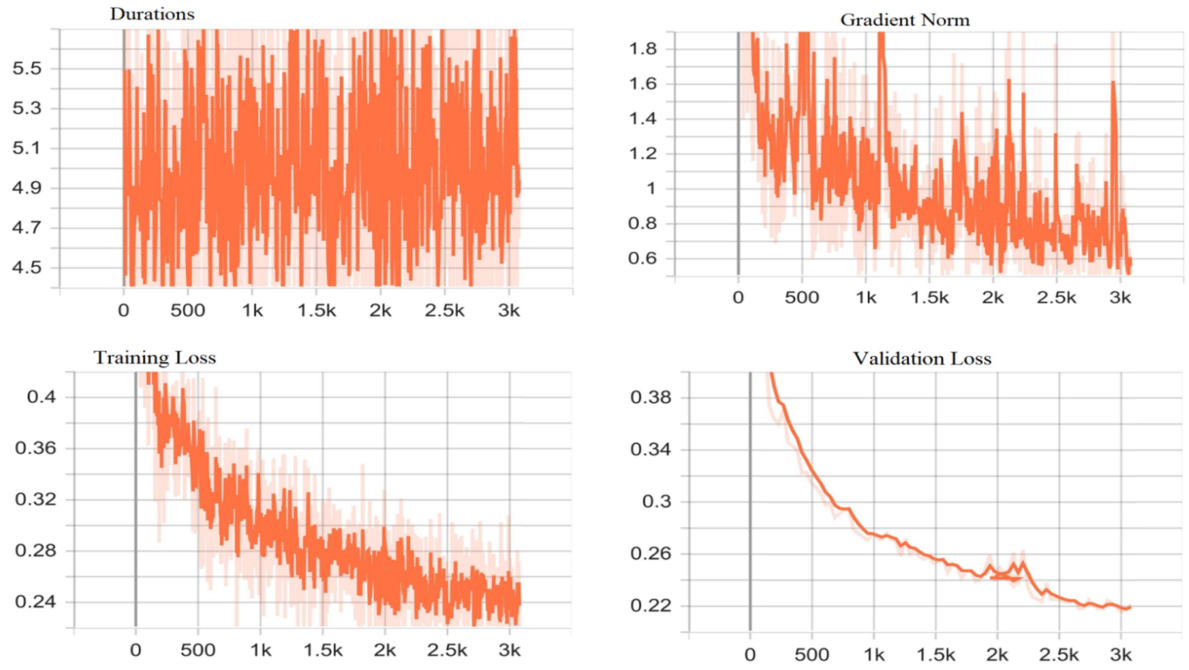


Figure 45: Tensorboard Scalar graph of the Fine-tuned Model

Evaluation metrics play a crucial role in assessing the performance of a training model. In our project, we utilized Tensorboard to visualize various metrics throughout the training process, which lasted approximately 6 hours and spanned around 90 epochs on Google Colaboratory. The gradient norm, indicative of gradient stability, remained consistent, signifying the absence of significant gradient explosions or vanishing gradients.

5.6.5 Fine-tuning HIFI-GAN Model

In our study, we fine-tuned the HIFI-GAN model vocoder to generate high-quality speech for the Afaan Oromoo language. The fine-tuning process involved using the same dataset that was used to train our Tacotron2 model. We specifically fine-tuned the Universal HIFI-GAN model, which was originally trained on English language data.

To do this, we first loaded the pre-trained Universal HIFI-GAN model, which was learned on English language data, together with our fine-tuned Tacotron2 model, which was trained on Afaan Oromoo. These models were combined to modify the HIFI-GAN vocoder to better fit the unique qualities of Afaan Oromoo. The fine-tuning process lasted for a total of 10500 iterations. Before initiating the training, we generated ground-truth alignment spectrograms. These spectrograms served as a reference for HiFi-GAN to learn and mimic the speech characteristics produced by our Tacotron2 model.

We kept an eye on the Mel-spectrogram error during training to gauge how well the fine-tuning was working. 0.510 was recorded as the Mel-spectrogram error at the start of the training. The model's performance significantly increased throughout the training process, decreasing the Mel-spectrogram error. The model got more accurate at producing Mel-spectrograms that closely mirrored the target speech at 6k iterations, as seen in generator Mel-spectrogram error reaching a value of 0.339, we ignore the model saved after this iterations and we take the model saved at 6k iteration for evaluation. This process enabled us to generate high-fidelity and near-natural speech samples for the Afaan Oromoo language, paving the way for improved text-to-speech synthesis in this language.

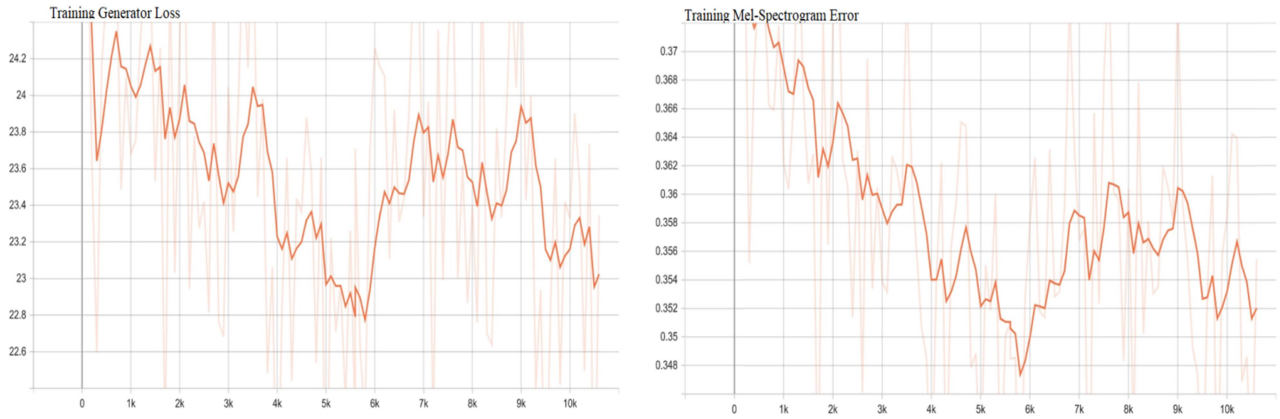


Figure 46: HIFI-GAN Training Generator Loss and Mel-Spectrogram Error

HIFI-GAN Training Generator Loss: The Generator Loss calculates the difference between the ground truth speech from the training dataset and the synthesized speech produced by the HIFI-GAN model. It demonstrates how well the generator network can produce speech that sounds remarkably similar to the target speech. Greater alignment between the generated speech and the real world means lower generator loss, which suggests better speech synthesis with higher quality and greater accuracy.

Mel-Spectrogram Error: This metric evaluates the variation between the target Mel-spectrogram, which is obtained from the ground truth speech, and the generated Mel-spectrogram, which represents a speech in the frequency domain. It measures the distortion or departure from the planned target in the mel-spectrogram that was created. More accurate and realistic-sounding synthesized speech is produced when the model generates Mel-spectrograms that closely resemble the target spectrograms, which is indicated by a reduced Mel-spectrogram error.

5.6.6 Testing Inference

It can be difficult to provide any quantitative evaluation of a Tacotron2 model's output, making performance evaluation tough. A Tacotron2 model's output can often be assessed by listening to the generated speech and evaluating how closely it matches the input text. One common approach to evaluating the quality of generated speech is to use subjective listening tests, where a group of human evaluators listens to the generated speech and provides ratings based on various criteria. These criteria may include measures of speech naturalness, intelligibility, and overall

quality. The figure below shows the input text provided to the model and the resulting Mel-spectrogram and audio generated using the HiFi-GAN as a vocoder.

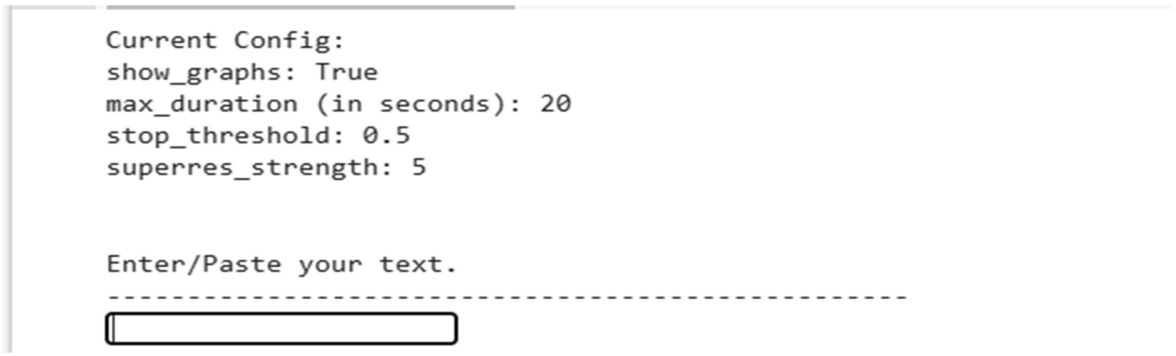


Figure 47: Request Inference Input Text Form

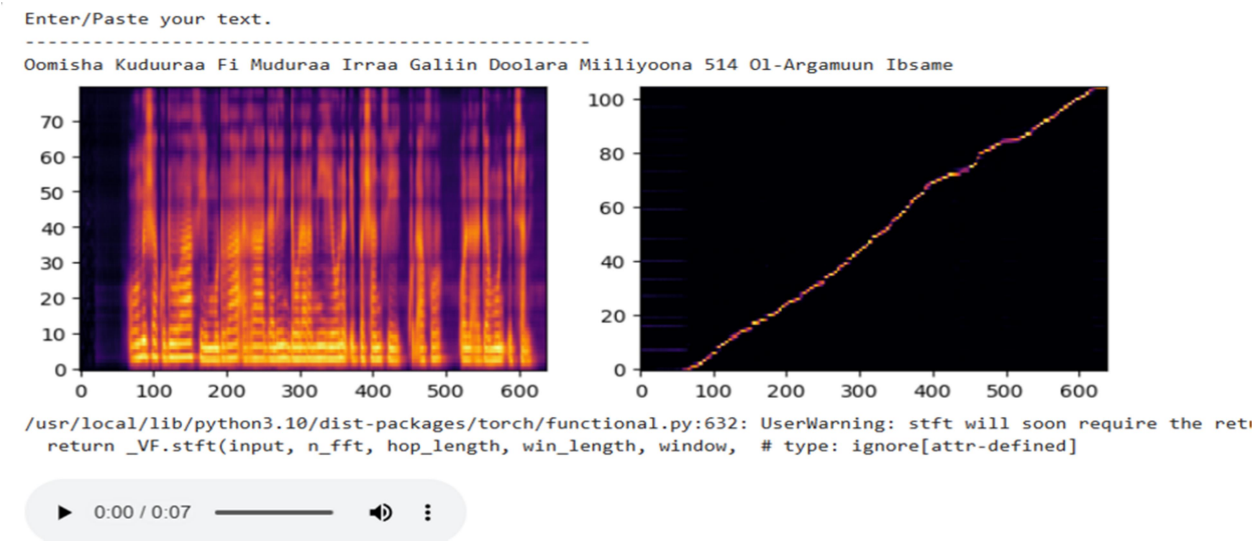


Figure 48: Test Input Text and output Result 1

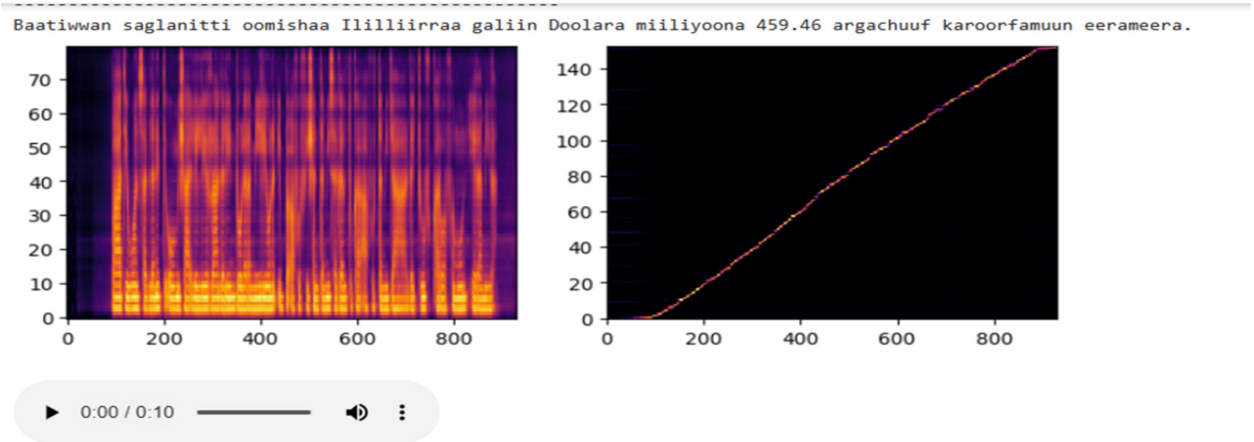


Figure 49: Test Input Text and Output Result 2

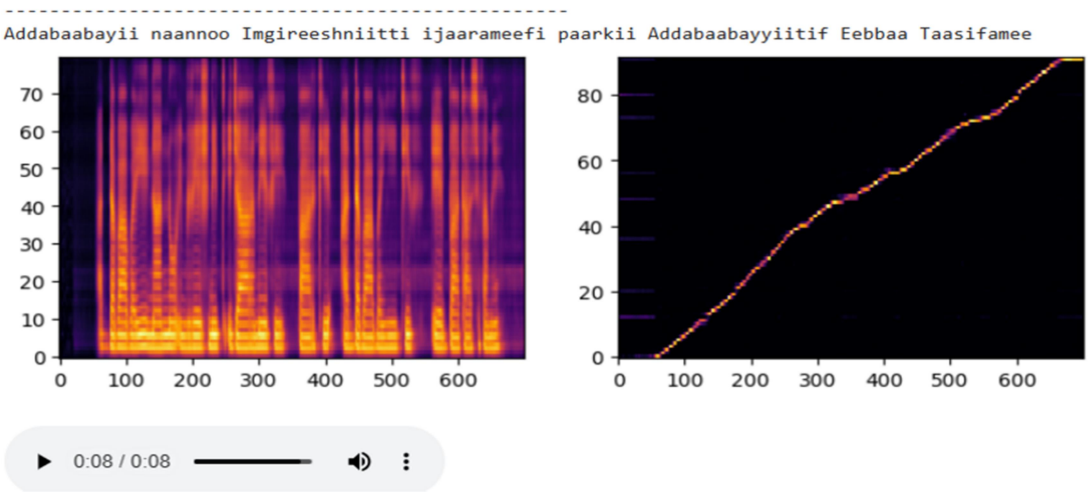


Figure 50: Test Input Text and Output Result 3

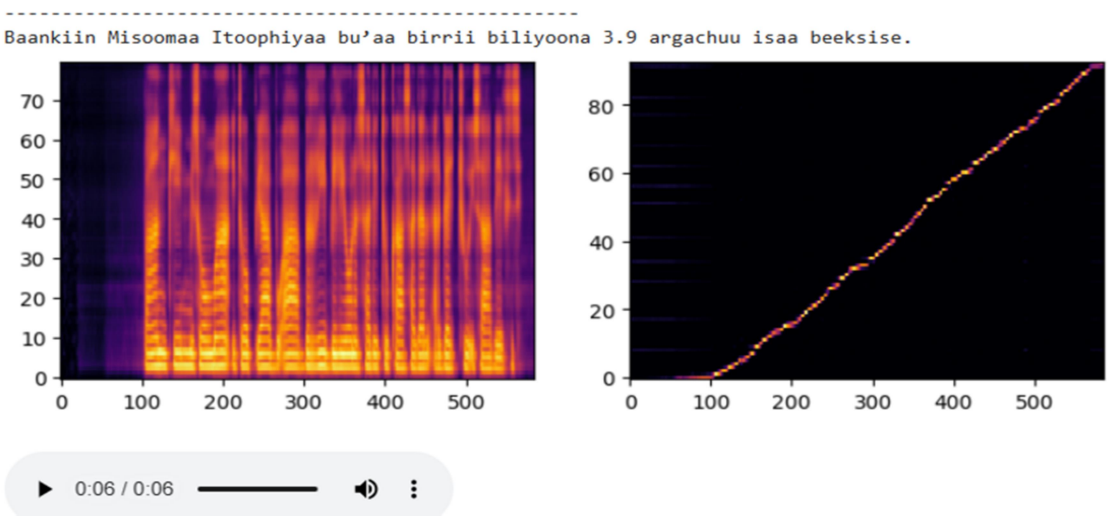


Figure 51: Test Input Text and Output Result 4

5.7 Speech Synthesis

In the speech synthesizing phase of our study, we employed a multi-step process to convert input images into synthesized speech. The overall workflow involved several key components, including text extraction using Tesseract OCR, inputting the extracted text into a trained Tacotron2 model for Mel-spectrogram prediction, and using a vocoder to generate speech from the predicted Mel-spectrograms. Both the Tacotron2 model and the vocoder were evaluated, including the model trained from scratch and the fine-tuned model.

We utilized Tesseract OCR to extract text from the input images; the extracted text was fed into a Tacotron2 model. We trained our Tacotron2 model from scratch on our own dataset and data

cleaners, ensuring that it learned the patterns and nuances of the target language. The Tacotron2 model predicted Mel-spectrograms, which are representations of the speech in the frequency domain over time. These Mel-spectrograms capture the acoustic characteristics of the speech, such as the pitch, intonation, and pronunciation.

To convert the predicted Mel-spectrograms into audible speech, we employed two different vocoders: WaveGlow and HIFI-GAN. The WaveGlow vocoder was trained from scratch using our data, while the HIFI-GAN vocoder was fine-tuned on our dataset using a pre-trained HIFI-GAN model. The vocoders take the Mel-spectrograms as input and generate high-quality audio output. In the evaluation of the Tacotron2 model and vocoders, we employed Mean Opinion Score (MOS) to assess the quality and naturalness of the synthesized speech. This evaluation process involved presenting the synthesized speech to a group of evaluators who listened to the speech samples and provided their ratings based on criteria such as clarity, naturalness, and overall quality.

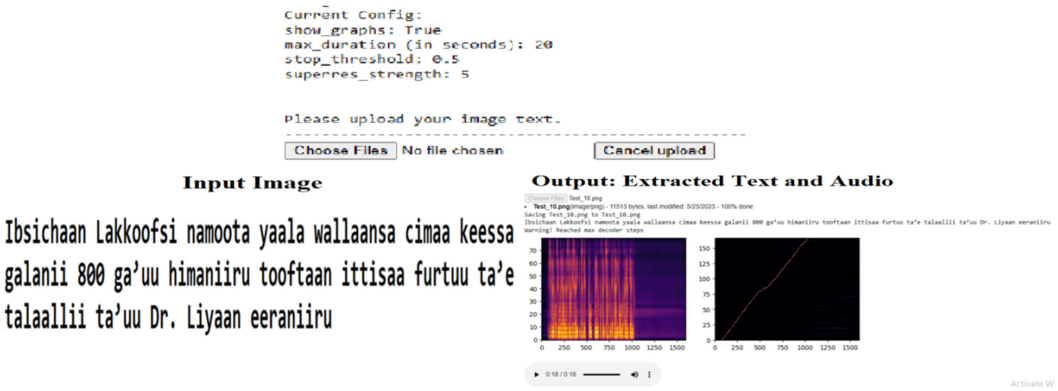


Figure 52: Test Input Image and Output Result 1

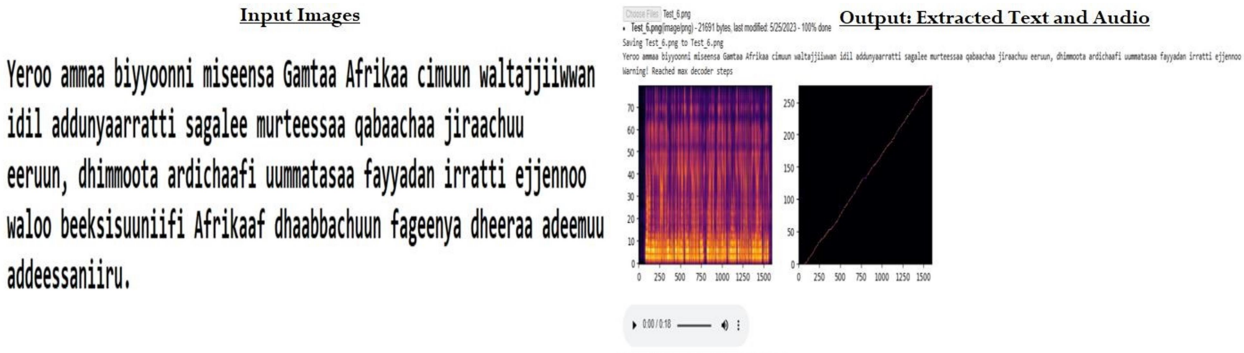


Figure 53: Test Input Image and Output Result 2

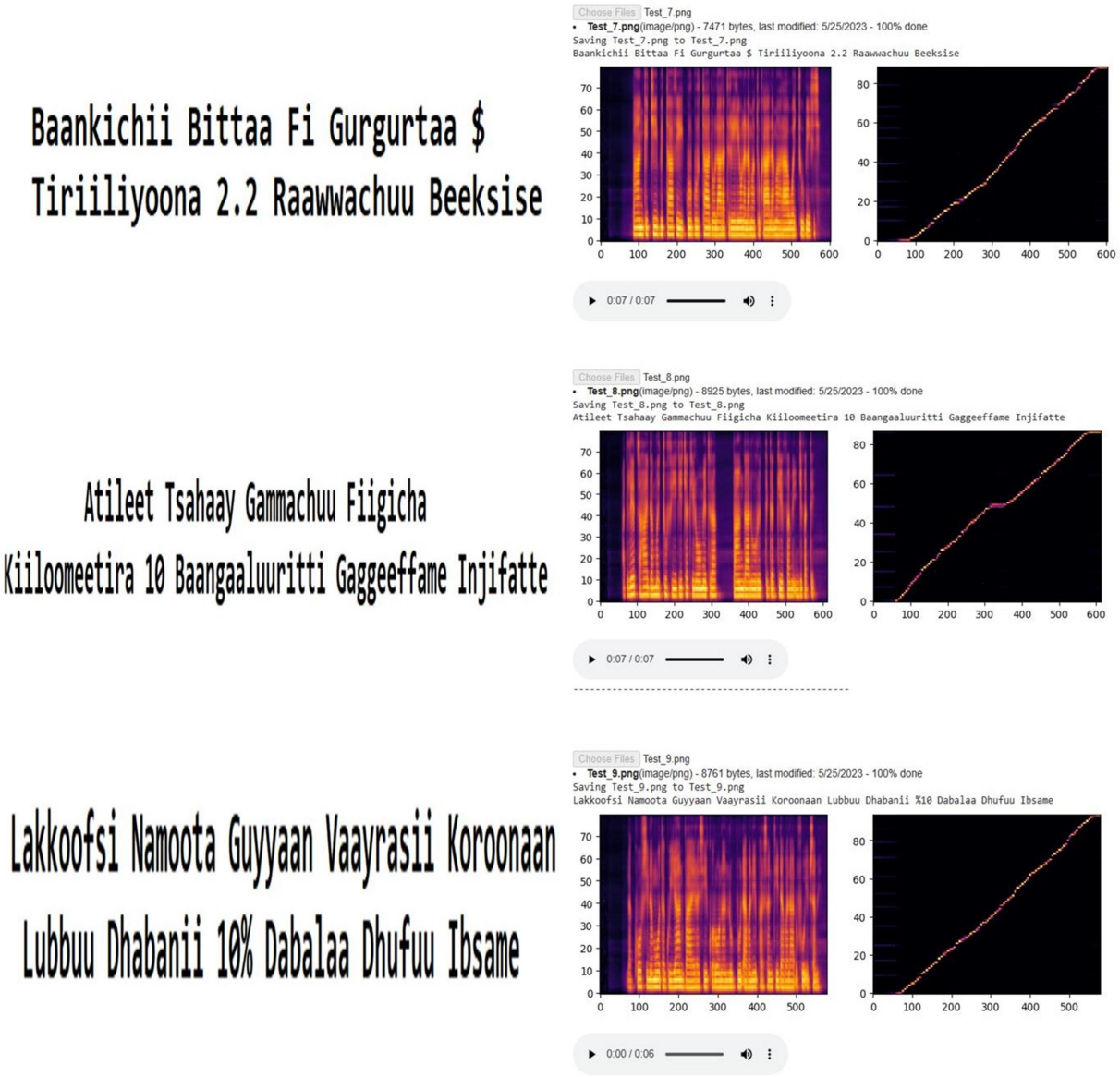


Figure 54: Test Input Images and Output Result 3

CHAPTER SIX

6. RESULT EVALUATION AND DISCUSSION

The evaluation of our speech synthesizing phase involved assessing the performance of Tesseract-OCR for text extraction from input images, as well as the quality of the synthesized speech. In terms of Tesseract-OCR, we evaluated its performance using the Word Error Rate (WER) metric, which measures the accuracy of the extracted text compared to the ground truth. The evaluation results demonstrated the effectiveness of Tesseract-OCR in accurately recognizing and extracting text from the input images, with a relatively low WER. Moving on to the quality of the synthesized speech, subjective Mean Opinion Score (MOS) ratings and Objective Mel Cepstral Distortion (MCD) were obtained to evaluate the speech output. The results revealed remarkable progress in achieving natural-sounding speech, with the synthesized speech being perceived as almost near to natural sound and of high quality by human evaluators. Although occasional word omissions were observed in some instances, it was not a consistent issue and did not occur in every synthesized speech sample.

This represents a significant improvement compared to previous text-to-speech synthesis approaches for Afaan Oromoo, where the output often sounded less natural and had more pronounced errors. The Tacotron2 model's effectiveness, the fine-tuning procedure, and the high-end vocoders used in our pipeline are all responsible for the improved performance of our speech synthesis system. The outcomes show excellent potential for additional developments in voice synthesis for Afaan Oromoo, which will improve the overall user experience in a variety of speech-generating applications.

6.1 Result Evaluation

In our study, we chose Tesseract OCR as the optical character recognition (OCR) system for several reasons. Tesseract-OCR is an open-source OCR engine that has gained popularity and widespread adoption in the research and development community.

Tesseract-OCR is an open-source Nature which means that its source code is freely available and can be modified and customized according to specific needs. it also language supports a wide range of languages, this language support was crucial for our study and its ability to handle complex scripts and character sets, including Latin and non-Latin languages. Tesseract-OCR has

shown reliable and robust performance in various OCR tasks. Tesseract OCR can be easily integrated into different software applications and workflows. It provides APIs and libraries for seamless integration with programming languages such as Python, making it convenient for researchers and developers to incorporate OCR capabilities into their projects(Smith, 2005).

Considering these factors, we found Tesseract OCR to be a suitable choice for our study. Its open-source nature, language support, robust performance, community support, and integration flexibility aligned well with our requirements for accurately recognizing Afaan Oromoo text from images.

Why Tacotron2 Model?

- ✓ Tacotron2 is a widely used text-to-speech (TTS) model known for its impressive natural-sounding speech synthesis capabilities.
- ✓ Tacotron2 is its end-to-end architecture, which directly generates Mel-spectrograms from input text without relying on intermediate linguistic features (Shen et al., 2018).
- ✓ Tacotron2 utilizes an attention mechanism to align the generated Mel-spectrogram frames with the input text, allowing for more accurate and expressive synthesis(Shen et al., 2018).

Other Models

- ✓ The Hidden Markov Model (HMM) operates on the principle that the current state only relies on the preceding state. This implies that HMMs are built on the assumption that there is statistical independence between the hidden states (which represent linguistic units or phonemes) and the observed acoustic features (such as spectral characteristics of speech). However, HMMs face difficulties in capturing the intricate relationships and subtleties in speech, such as prosody, intonation, and timing(Abraham et al., 2022).
- ✓ The absence of autoregressive generation in FastSpeech can lead to limitations in capturing fine-grained prosody and timing details. Tacotron2, being autoregressive, can naturally model duration information and better capture the temporal dynamics and nuances of speech, including intonation and rhythm. Modeling, In this case, FastSpeech can lead to less natural-sounding speech in certain cases(Gopalakrishnan et al., 2022). According to (Gopalakrishnan et al., 2022) A survey was conducted to assess the performance of different models when trained on small datasets. The findings of the

survey revealed that the Tacotron2 text-to-speech (TTS) model consistently produced the most realistic and natural-sounding speech among the models analyzed. This implies that Tacotron2 outperformed other models in terms of synthesizing speech that closely resembles human speech.

WaveGlow stands out for its ability to generate high-quality, natural-sounding speech with minimal artifacts and distortions. Unlike traditional vocoders like Griffin-Lim, WaveGlow is based on a flow-based generative model, which allows for more accurate modeling of the complex distribution of audio signals. WaveGlow produces highly intelligible speech with good prosody, capturing subtle details and nuances in the audio(Prenger et al., 2019).

HiFi-GAN is an advanced vocoder model that excels in generating high-fidelity speech waveforms. HiFi-GAN is trained using a multi-resolution spectrogram architecture, which allows it to capture fine-grained details in the Mel-spectrograms and produce high-quality audio output(Kong et al., 2020). It has been praised for its ability to handle challenging acoustic conditions and produce natural-sounding speech even in low-resource settings. Compared to other vocoders like MelGAN, WaveNet, or WaveGlow, HiFi-GAN generally achieves better audio quality and a more accurate representation of the target speech signal(Kong et al., 2020). However, HiFi-GAN may require more computational resources and training data to achieve optimal results compared to simpler vocoder models that why we choose Fine-tuning the pre-trained model.

6.1.1 Word Error Rate Evaluation

To evaluate the accuracy of the Tesseract OCR system, we utilized the word error rate metric. WER measures the discrepancy between the recognized text output by the OCR system and the ground truth text(Kenneth, 2021). We carefully curated a representative dataset of Afaan Oromoo texts including various fonts, sizes, blur, and skewed images to test the OCR system's performance. The accuracy of Tesseract OCR can vary depending on several factors, such as the quality of the input images, the clarity of the text, the language being recognized, and the specific use case. According to the Tesseract OCR Engine Wiki on GitHub, the accuracy of Tesseract OCR can be improved by using specific preprocessing techniques.

6.1.2 Subjective Evaluation

Table 5: Afaan Oromoo Text-to-Speech System Comparison

No	Title	Authors	Total Dataset	Model Used	Integrate OCR System	MOS Result
1.	Text to Speech Synthesizer for Afaan Oromoo using Statistical Parametric Speech Synthesis	Muhidin Kadir Wosho (2020)	400 text and audio pairs	Statistical parametric speech synthesis based on HMM	No	Intelligibility 4.3 and Naturalness 4.1
2.	Text to Speech Synthesizer for Afaan Oromo Using Hidden Markov Model	Muhidin Kadir Wosho (2021)	600 text and audio pairs	hidden Markov models (HMM)	No	Intelligibility 4.3 and Naturalness 4.1
3.	Text to Speech Synthesizer For Afaan Oromo using Deep neural network	Chala Sembeta Tashoma (2021)	8076 text and audio pairs	Deep Voice3 and Tacotron 2 model with Griffim-Lim Vocoder	No	Intelligibility 3.28 and naturalness 3.02 for [Deep Voice 3], Intelligibility 4.32 and naturalness 4.21 for [Tacotron2]
4.	Text To Speech Synthesis for Afaan Oromoo Language Using Deep Learning Approach	Soressa Beyene (2022)	1000 text and audio pairs	Festival toolkit for Feature Extraction based on BLSTM-RNN	No	Intelligibility 3.77 and naturalness 3.76

In our study, to assess the quality of synthesized speech using our Text-Reading model, we employed the mean opinion score (MOS).

MOS is a subjective evaluation method where human listeners rate the quality of the synthesized speech based on a predefined scale. We gathered a group of native Afaan Oromoo Speakers from the Dilla University, Meda-Welabu University and Oda-Bultum University Department of Afaan Oromoo who listened to audio samples generated by the Text-reading model and provided ratings on naturalness and intelligibility. The average MOS score reflects the overall perceived quality of the synthesized speech.

Table 6: Mean Opinion Score Evaluation Result our Experiments

No	Experiments	Integrate OCR System	Intelligibility	naturalness
1	Tacotron2 With WaveGlow	Yes	4.11	3.87
2	Tacotron2 with HIFI-GAN	Yes	4.39	4.35

6.1.3 Objective Evaluation

In our study, we employed the widely-used objective evaluation method known as Mel Cepstral Distortion (MCD) tests to assess the quality of the synthesized speech generated by our Afaan Oromoo Text Reading model. The MCD test is a popular metric used to measure the similarity between the synthesized speech and the original reference speech. To perform the MCD test, we extracted the mel-spectrograms from both the synthesized speech and the reference speech. Then, we computed the distance between the mel cepstral coefficients of the two spectrograms, using root mean square error (RMSE) then percentage of the dynamic range. A lower MCD score indicates a higher similarity between the synthesized speech and the reference speech, indicating better quality and accuracy.

Table 7: Mel Cepstral Distortion (MCD) conducted evaluation result

Model	Vocoder	S1%	S2%	S3%	S4%	S5%	S6%	Average percentage
Tacotron2	WaveGlow	5.67%	4.19%	5.47%	5.74%	5.17%	5.51%	**5.29%**
Fine-tuned Tacotron2	Fine-Tuned HIFI-GAN	3.09%	3.39%	2.83%	4.87%	2.64%	2.14%	**3.16%**
Authors			**Model Used**				**Total Average**	
Soressa Beyene (2022)			BLSTM-based on RNN				**3.89**	
			merlin wave				**3.71**	

The table above presents the results of the Mel Cepstral Distortion (MCD) test conducted on two variations of the Tacotron2 model: one trained from scratch with the WaveGlow vocoder, and

the other fine-tuned with the HIFI-GAN vocoder. The MCD test measures the percentage of the dynamic range, calculated using root mean square error (RMSE), between the ground truth samples and the synthesized speech then comparing with other study result. However, it's important to note that MCD is a perceptual metric and may not fully capture the quality of synthesized speech as perceived by human listeners(Anumanchipalli et al., 2019).

6.2 Discussion of the Result

This study marks a pioneering effort in the development of an Afaan Oromoo Text Reading model, leveraging the integration of Tesseract-OCR with a deep neural network approach. Through extensive experimentation and the utilization of the benchmarked dataset corpus, supplemented by additional text and audio pair datasets, we achieved remarkable results surpassing previous TTS systems in the field. Our comprehensive literature review guided us in selecting the necessary software and hardware components to realize our envisioned reading device.

In our initial experiment, we sought to identify a suitable OCR system capable of accurately extracting Afaan Oromoo text from images. After rigorous testing of various OCR systems, we determined that Tesseract-OCR, coupled with appropriate image pre-processing techniques, outperformed other alternatives. With an accuracy exceeding 95% and a minimum word error rate, Tesseract-OCR proved its effectiveness in extracting text from images.

Subsequently, we embarked on training Tacotron2 and WaveGlow models from scratch, employing our Afaan Oromoo dataset. Tacotron2, a recurrent neural network (RNN)-based model, was trained using Mel-spectrogram representations of audio signals over 500 epochs with a batch size of 8, each consisting of 1566 iterations for 15 days. Based on the observed training and validation loss, we decided to conclude training when the training loss reached 0.253041690178744 and the validation loss stood at 0.34068194017557. These loss values indicated a satisfactory level of model performance, suggesting that further training would yield diminishing returns or risk overfitting. Our training journey also encompassed the WaveGlow model, which served as the vocoder for the Tacotron2 model. Training the WaveGlow model involved 497 epochs, with periodic model check-pointing every 10 epochs for 10 days. Below table shows training and validation loss of reviewed Text-to-Speech Synthesis and ours.

Table 8: Training and Validation loss of Models Training

Authors	Model	Training Loss	Validation Loss	Total Iterations / Epochs
(Tashoma, 2021)	Tacotron2	0.889	1.15	100k iterations
(Beyene, 2022)	Festival toolkit for Feature Extraction based on BLSTM-RNN	0.15	0.25	25 epochs
Our Training	**Model Name**	**Training Loss**	**Validation Loss**	**Iterations(epochs)**
	Tacotron2 From Scratch	0.2543270669	0.343282864	500 epochs
	WaveGlow From Scratch	-9.4784660339	-8. 33072913	490 epochs
	Fine-tuning Tacotron2	0.218435	0. 233419	3000 iterations
	Fine-tuning HIFI-GAN	21.8 (Generator Training loss)	0.339 (Generator Mel-Spectrogram error)	6000 iterations

Our third experiment revolved around fine-tuning pre-trained models, specifically Tacotron2 and HIFI-GAN, using our datasets and customized hyperparameters. Through this process, we achieved impressive results, nearing natural sound and receiving positive subjective evaluations with significantly fewer training iterations and time requirements. Notably, the Tacotron2 model reached optimal performance after 3,000 iterations, while the HIFI-GAN model exhibited the best results of up to 6,000 iterations. Limited by time and resources, we were unable to train the HIFI-GAN model from scratch, which typically necessitates a minimum of 3 million iterations for optimal results.

Subjective evaluation tests were conducted for both the second and third experiments, assessing performance by collecting unseen text data in image format. The Tacotron2 model paired with the WaveGlow vocoder, trained from scratch, achieved impressive MOS scores of 4.11 and 3.87

for intelligibility and naturalness, respectively. These scores suggest that further dataset augmentation and training are required to enhance the naturalness of the synthesized speech. On the other hand, the fine-tuned pre-trained Tacotron2 and HIFI-GAN models achieved MOS scores of 4.39 and 4.35 for intelligibility and naturalness, respectively. This highlights the viability of fine-tuning pre-trained models for low-resourced languages like Afaan Oromoo, enabling successful speech synthesis with reduced data requirements and training time.

The objective evaluation results of our study reveal the performance of two variations of the Tacotron2 model: one trained from scratch with the WaveGlow vocoder and the other fine-tuned with the HIFI-GAN vocoder. The average percentage of the dynamic range, calculated using root mean square error (RMSE), for the Tacotron2 trained from scratch with WaveGlow is 5.29%. This indicates a moderate level of deviation between the ground truth samples and the synthesized speech. However, the fine-tuned Tacotron2 with the fine-tuned HIFI-GAN achieves a lower average percentage of the dynamic range at 3.16%, indicating a closer alignment between the ground truth samples and the synthesized speech. These results suggest that the fine-tuning process with the HIFI-GAN vocoder improves the quality and fidelity of the synthesized speech compared to the Tacotron2 model trained from scratch with WaveGlow.

How We Answered Our Research Questions:

RQ1: In our study, we improved the quality of speech synthesis by leveraging advanced model Tacotron2, using generative vocoders WaveGlow and HIFI-GAN, and by increasing dataset size. These models allowed us to generate high-quality and natural-sounding speech with improved control over voice characteristics such as pitch and prosody.

RQ2: To develop a model that can efficiently read Afaan Oromoo text, we conducted a series of experiments involving the training of different components of the speech synthesis system.

Firstly, we trained the Tacotron2 model from scratch using a large dataset of Afaan Oromoo text and corresponding speech pairs. This involved optimizing the model's architecture and parameters to effectively capture the linguistic patterns and characteristics of the language. Additionally, we fine-tuned the Tacotron2 model using a same dataset of Afaan Oromoo text and speech pairs. This fine-tuning process allowed the model to adapt and specialize in reading Afaan Oromoo text more efficiently, resulting in improved accuracy and naturalness.

Furthermore, we trained the WaveGlow vocoder from scratch, which is responsible for generating high-quality speech from the Tacotron2 model Mel-spectrograms. Lastly, we fine-

tuned the HIFI-GAN vocoder, using Afaan Oromoo speech data. This fine-tuning process enhanced the vocoder ability to generate high-quality and natural-sounding speech, further improving the efficiency of the overall speech synthesis.

RQ3: To evaluate the extent to which the newly developed model improves the quality of speech synthesis, we conducted extensive evaluations using objective and subjective metrics.

Subjective evaluation was conducted using the Mean Opinion Score (MOS) metric, where human evaluators rated the intelligibility and naturalness of the synthesized speech. In Table 7 and Table 8 of our study, we presented the MOS scores obtained for our model, as well as the scores for other models for comparison. The MOS score for fine-tuned Tacotron2 combined with fine-tuned HIF-GAN vocoder were consistently higher, indicating improved quality and naturalness compared to the other models. Objective evaluation was performed using the Mel-Cepstral Distortion (MCD) metric, which measures the similarity between the synthesized speech and the reference speech. In Table 9 of our study, we compared the MCD scores of our model to other existing models. The results showed that our model fine-tuned Tacotron2 combined with fine-tuned HIF-GAN vocoder achieved significantly lower MCD scores, indicating better alignment and similarity with the reference speech.

The Major Contributions of this Study are as Follows:

- Development of Afaan Oromoo Text Reading Model integrated with Tesseract OCR System.
- Improved Intelligibility and Naturalness of Afaan Oromoo Text to Speech Synthesis.
- Increased Afaan Oromoo Dataset corpus for Text to speech synthesis.
- Contribution to assistive technology, potentially influencing future research and advancements in this domain.
- Potential applications beyond visually impaired individuals: The outcomes of our study have potential applications for individuals with reading difficulties, content accessibility, audio content generation, and social and cultural preservation.

7. CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

This study presents a novel approach to developing the Afaan Oromoo Text Reading model by integrating the Tesseract-OCR system with an RNN-based speech synthesizer using a deep neural network approach. By applying advanced image pre-processing techniques, such as Binarization, Thresholding, deskewing, and resizing, the accuracy and efficiency of text extraction from images were significantly improved. In addition to the Chala Sembeta dataset corpus, we curated a comprehensive dataset of 9000 text and audio pairs from diverse sources, including social media news and the Holy Quran, to train our models.

The proposed solution successfully addressed the challenge of Afaan Oromoo text reading by utilizing the Tesseract-OCR system with image pre-processing and speech synthesis. We developed a Text-to-Speech model using a deep neural network approach. The training process involved training the Tacotron2 model from scratch and fine-tuning pre-trained models using our dataset. We also trained WaveGlow and HIFI-GAN vocoders to generate high-quality audio from the predicted Mel-Spectrograms.

During the speech synthesizing phase, we evaluated the performance of both the Tacotron2 model with WaveGlow vocoder and the Tacotron2 model with HIFI-GAN vocoder by conducting subjective evaluation tests. The results showed that the Tacotron2 model trained from scratch with WaveGlow vocoder achieved a score of 4.11 and 3.87 for both intelligibility and naturalness. However, further improvements are necessary to achieve more natural sound. On the other hand, the fine-tuned pre-trained Tacotron2 and HIFI-GAN models achieved a score of 4.39 and 4.35 for both intelligibility and naturalness, indicating the effectiveness of fine-tuning pre-trained models for low-resource languages like Afaan Oromoo.

In conclusion, our study makes significant contributions by successfully developing an Afaan Oromoo Text Reading model and proposing an efficient solution for text extraction and speech synthesis. The findings suggest that further improvements can be achieved by increasing the dataset size, exploring hyperparameters tuning, and conducting more extensive training. This study sets the stage for future research in Afaan Oromoo natural language processing and offers

promising avenues for the development of highly accurate and natural-sounding TTS systems in the language.

7.2 Recommendations

7.2.1 Future Work

Dataset Expansion: To improve the performance and generalization capabilities of the developed TTS system, future researchers should focus on expanding the dataset. This can be achieved by collecting more diverse and representative text and audio pairs from various sources such as books, newspapers, online articles, and conversations. By including a wide range of linguistic variations, accents, and speech patterns, the TTS system will be better equipped to handle different scenarios and deliver more accurate and natural-sounding output.

Expanding language support: Adding Most Spoken language in Ethiopia to support individuals with impairment.

Incorporate Speech Recognition Techniques: Integrating speech recognition techniques into the TTS system can enhance its functionality and usability for visually impaired individuals. By enabling speech input, users would have the option to interact with the system through voice commands, making it more intuitive and user-friendly. This would involve implementing state-of-the-art speech recognition algorithms and techniques to accurately transcribe spoken words into text, which can then be processed and synthesized by the TTS system.

Adding Navigation and Sign Recognition: To further assist visually impaired individuals in their daily activities, future work could focus on incorporating navigation capabilities into the TTS system. This could involve integrating GPS technology and street mapping data to provide real-time audio-based navigation instructions, helping users navigate safely and independently through unfamiliar environments. Additionally, incorporating sign recognition technology would enable the system to identify and read aloud signs, labels, and important visual cues, providing crucial information to visually impaired individuals while they are out and about.

Facial Recognition for Personalization: To enhance the user experience and foster social connectivity, future researchers can explore the integration of facial recognition technology into the TTS system. This would enable the system to recognize and identify familiar faces, such as family members or close friends, and associate them with specific audio profiles. By

personalizing the reading experience, the TTS system can deliver tailored content, including personalized messages, greetings, and other relevant information based on the detected faces.

Integration with Real-World Applications: Future researchers should explore the integration of the developed Afaan Oromoo Text Reading model with real-world applications and assistive technologies. This could involve collaborations with software developers, accessibility organizations, and assistive device manufacturers to integrate the Text Reading model into existing platforms or develop new applications specifically tailored to the needs of visually impaired individuals.

In conclusion, future research in the field of an Afaan Oromoo Text Reading model should consider expanding the dataset, incorporating speech recognition techniques, adding navigation and sign recognition capabilities, incorporating facial recognition for personalization, integrating with real-world applications, and exploring post-processing techniques. These advancements will contribute to the development of a more versatile, user-friendly, and effective Text Reading model that can provide to the unique needs of visually impaired individuals in the Afaan Oromoo.

***

SPECIAL ACKNOWLEDGEMENT

This research was funded by Adama Science and Technology University with grant number **ASTU/SM-R/815/23** Adama, Ethiopia

REFERENCES

Aaron, B. (2021a). *Text To Speech — Foundational Knowledge*. https://towardsdatascience.com/text-to-speech-foundational-knowledge-part-2-4db2a3657335

Aaron, B. (2021b). *Text To Speech — Lifelike Speech Synthesis*. https://towardsdatascience.com/text-to-speech-lifelike-speech-synthesis-demo-part-1-f991ffe9e41e

Abhishek, A. (2020). *Image Text Recognition Using Deep Learning and Deploying the model in Cloud*. https://medium.com/analytics-vidhya/image-text-recognition-738a368368f5

Abraham, K., Gassiat, E., & Naulet, Z. (2022). Fundamental limits for learning hidden Markov model parameters. *IEEE Transactions on Information Theory*.

Adam, E. E. B. (2020). Deep learning based NLP techniques in text to speech synthesis for communication recognition. *Journal of Soft Computing Paradigm (JSCP)*, *2*(04), 209–215.

Aishwarya.27. (2023). *Recurrent Neural Network*. https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/

Ajitesh, K. (2019). *Neural Network Architecture for Text-to-Speech Synthesis*. https://vitalflux.com/neural-network-architecture-tts/#:~:text=Decoder network%3A The decoder network %28autoregressive recurrent neural,fed into Vocoder which creates time-domain waveforms %28Speech%29.

Alam, M., Samad, M. D., Vidyaratne, L., Glandon, A., & Iftekharuddin, K. M. (2020). Survey on Deep Neural Networks in Speech and Vision Systems. *Neurocomputing*, *417*, 302–321. https://doi.org/10.1016/j.neucom.2020.07.053

Alkaddo, A., & Albaqal, D. (2022). Implementation of OCR using Convolutional Neural Network (CNN): A Survey. *Journal of Education and Science*, *31*, 27–41. https://doi.org/10.33899/edusj.2022.133711.1236

Alsaid, H., Alkhatib, L., Aloraidh, A., Alhaidar, S., & Bashar, A. (2019). Deep Learning Assisted Smart Glasses as Educational Aid for Visually Challenged Students. *2019 2nd International Conference on New Trends in Computing Sciences, ICTCS 2019 - Proceedings*. https://doi.org/10.1109/ICTCS.2019.8923044

Álvarez, D., Pascual, S., & Bonafonte, A. (2019). *Problem-Agnostic Speech Embeddings for Multi-Speaker Text-to-Speech with SampleRNN*. 35–39. https://doi.org/10.21437/ssw.2019-7

Anuj, S. (2021). *Introduction to Audio Analysis and Processing*. https://blog.paperspace.com/introduction-to-audio-analysis-and-synthesis/

Anumanchipalli, G. K., Chartier, J., & Chang, E. F. (2019). Speech synthesis from neural decoding of spoken sentences. *Nature*, *568*(7753), 493–498. https://doi.org/10.1038/s41586-019-1119-1

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *ArXiv Preprint ArXiv:1409.0473*.

Beyene, S. (2022). Text To Speech Synthesis for Afaan Oromoo Language Using Deep Learning Approach. *New Media and Mass Communication*, *101*, 15–33. https://doi.org/10.7176/nmmc/101-02

Bezabih, L., Abebe, T. W., & Fite, R. O. (2017). Prevalence and factors associated with childhood visual impairment in Ethiopia. *Clinical Ophthalmology*, *11*, 1941–1948. https://doi.org/10.2147/OPTH.S135011

Bhardwaj, P., & Sharma, R. K. (2020). *Image Text Reader for Visually Impaired in Desired Language*.

Bieniecki, W., Grabowski, S., & Rozenberg, W. (2007). Image preprocessing for improving OCR accuracy. *Proceeding of the 3rd International Conference of Young Scientists "Perspective Technologies and Methods in MEMS Design", MEMSTECH 2007*, 75–80. https://doi.org/10.1109/MEMSTECH.2007.4283429

Brailleworks.com. (2023). *Audio Documents for People Who are Blind, Visually Impaired or with a Reading Disability*. https://brailleworks.com/audio/

Brauwers, G., & Frasincar, F. (2021). A general survey on attention mechanisms in deep learning. *IEEE Transactions on Knowledge and Data Engineering*.

Britz A. (2017). *Recurrent Neural Networks Tutorial , Part 1-4*. 1–15.

Cornegruta, S., Bakewell, R., Withey, S., & Montana, G. (2016). Modelling radiological language with bidirectional long short-term memory networks. *ArXiv Preprint ArXiv:1609.08409*.

Dennis, J., Dat, T., & Li, H. (2011). Spectrogram Image Feature for Sound Event Classification in Mismatched Conditions. *Signal Processing Letters, IEEE*, *18*, 130–133. https://doi.org/10.1109/LSP.2010.2100380

DiPietro, R., & Hager, G. D. (2020). Chapter 21 - Deep learning: RNNs and LSTM. In S. K. Zhou, D. Rueckert, & G. B. T.-H. of M. I. C. and C. A. I. Fichtinger (Eds.), *The Elsevier and MICCAI Society Book Series* (pp. 503–519). Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-12-816176-0.00026-0

Edward, M. (2018). *Besides Word Embedding, why you need to know Character Embedding?*

Edward, M. (2019). *Besides Word Embedding, why you need to know Character Embedding?*

Eric, U., Sally, B., Patrick, F., & Laujan. (2023). *What is text to speech?* https://learn.microsoft.com/en-us/azure/cognitive-services/speech-service/text-to-speech

Farahani, A., Pourshojae, B., Rasheed, K., & Arabnia, H. R. (2020). A Concise Review of Transfer Learning. *Proceedings - 2020 International Conference on Computational Science and Computational Intelligence, CSCI 2020*, 344–351. https://doi.org/10.1109/CSCI51800.2020.00065

Ganesan, J., Azar, A. T., Alsenan, S., Kamal, N. A., Qureshi, B., & Hassanien, A. E. (2022). Deep Learning Reader for Visually Impaired. In *Electronics* (Vol. 11, Issue 20). https://doi.org/10.3390/electronics11203335

Gebrehiwot, Y. G. (2015). *Towards More Inclusive University Curricula the Learning Experiences of Visually Impaired Students in Higher Education Institutions of Ethiopia*. *June*, 310.

Gopalakrishnan, T., Imam, S. A., & Aggarwal, A. (2022). Fine Tuning and Comparing Tacotron 2, Deep Voice 3, and FastSpeech 2 TTS Models in a Low Resource Environment. *2022 IEEE International Conference on Data Science and Information System (ICDSIS)*, 1–6. https://doi.org/10.1109/ICDSIS55133.2022.9915932

Gourav, S. (2021). *Introduction to Artificial Neural Networks*. https://www.analyticsvidhya.com/blog/2021/09/introduction-to-artificial-neural-networks/

Harald, S. (2018). *Build a Handwritten Text Recognition System using TensorFlow*. https://towardsdatascience.com/build-a-handwritten-text-recognition-system-using-tensorflow-2326a3487cd5

Hassan, E. A., & Tang, T. B. (2016). Smart Glasses for the Visually Impaired People. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9759*, V–VI. https://doi.org/10.1007/978-3-319-41267-2

He, X., & Deng, L. (2017). Deep Learning for Image-to-Text Generation: A Technical Overview. *IEEE Signal Processing Magazine*, *34*(6), 109–116. https://doi.org/10.1109/MSP.2017.2741510

Himanshi, S. (2021). *How Do Computers Store Images?* https://www.analyticsvidhya.com/blog/2021/03/grayscale-and-rgb-format-for-storing-images/

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hore, P., & Chatterjee, S. (2022). *A Comprehensive Guide to Attention Mechanism in Deep Learning for Everyone*. https://www.analyticsvidhya.com/blog/2019/11/comprehensive-guide-attention-mechanism-deep-learning/

Iryna, S. (2021). *OCR with Deep Learning: The Curious Machine Learning Case*. https://labelyourdata.com/articles/ocr-with-deep-learning

Jason, B. (2017). *How Does Attention Work in Encoder-Decoder Recurrent Neural Networks*. https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks/

Joseph. (2022). *What You Need to Know About Bidirectional LSTMs with Attention in Py*. https://reason.town/bidirectional-lstm-with-attention-pytorch/

Karita, S., Chen, N., Hayashi, T., Hori, T., Inaguma, H., Jiang, Z., Someki, M., Soplin, N. E. Y., Yamamoto, R., Wang, X., & others. (2019). A comparative study on transformer vs rnn in speech applications. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 449–456.

Karthik, A., v. Kaarthick, R., & Prabakaran, S. (2018). *Voice Assistance for visually impaired people*.

Kate Beck. (2018). *Challenges That Blind People Face*. https://healthfully.com/challenges-that-blind-people-face-4369128.html

Kenneth, L. (2021). *OCR Evaluation Metrics*.

Kong, J., Kim, J., & Bae, J. (2020). HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, *2020-Decem*(NeurIPS).

Kumar, Y., Koul, A., & Singh, C. (2023). A deep learning approaches in text-to-speech system: a systematic review and recent research perspective. *Multimedia Tools and Applications*, *82*(10), 15171–15197. https://doi.org/10.1007/s11042-022-13943-4

Lara-alvarez, C., & Bayro-corrochano, E. (2014). *Automated Banknote Identification Method for the Visually Impaired*. 572–579.

Legese, N., Abdissa, B., Begna, Z., & Lemma, D. (2023). *Magnitude of Visual Impairment and associated factors among primary School Children of Ambo*. 1–19.

Leng, Y., Tan, X., Zhao, S., Soong, F., Li, X.-Y., & Qin, T. (2021). MBNet: MOS prediction for synthesized speech with mean-bias network. *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 391–395.

Levanier, J. (2020). *What is conceptual design?* https://99designs.com/blog/tips/conceptual-design/

Liu, Y., Wei, F., Li, S., Ji, H., Zhou, M., & Wang, H. (2015). A dependency-based neural network for relation classification. *ArXiv Preprint ArXiv:1507.04646*.

Mageshwaran, R. (2021). *Survey on Image Preprocessing Techniques to Improve OCR Accuracy*. https://medium.com/technovators/survey-on-image-preprocessing-techniques-to-improve-ocr-accuracy-616ddb931b76

Marchi, E., Vesperini, F., Squartini, S., & Schuller, B. (2017). Deep Recurrent Neural Network-Based Autoencoders for Acoustic Novelty Detection. *Computational Intelligence and Neuroscience*, *2017*, 4694860. https://doi.org/10.1155/2017/4694860

Masuyama, Y., Yatabe, K., Koizumi, Y., Oikawa, Y., & Harada, N. (2019). Deep griffin--lim iteration. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 61–65.

Mathieu, M., Couprie, C., & LeCun, Y. (2015). Deep multi-scale video prediction beyond mean square error. *ArXiv Preprint ArXiv:1511.05440*.

Mathur, A., Pathare, A., Sharma, P., & Oak, S. (2019). AI based Reading System for Blind using OCR. *Proceedings of the 3rd International Conference on Electronics and Communication and Aerospace Technology, ICECA 2019*, 39–42. https://doi.org/10.1109/ICECA.2019.8822226

Mitiku, B. (2020). *Braille reading and writing skills of students at Sebeta primary school for the blind*.

Mulugeta, E., & Mekuriaw, D. (2017). Academic challenges of visually challenged female

students in Addis Ababa University Ethiopia. *Global Journal of Human-Social Science (G), 17*(4), 1–7.

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *ArXiv Preprint ArXiv:1511.08458*.

Omede, A. A. (2015). The challenges of educating the visually impaired and quality assurance in tertiary institutions of learning in Nigeria. *International Journal of Educational Administration and Policy Studies*, *7*(7), 129–133. https://doi.org/10.5897/ijeaps2015.0407

Pancholi, P., & Patel, S. (2021). *Literature Survey for Applications of Artificial Neural Networks* (pp. 1–17). https://doi.org/10.4018/978-1-7998-4042-8.ch001

Pandya, K., & C.Goradiya, D. B. (2021). *AUDIO ASSISTED ELECTRONIC GLASSES FOR BLIND using deep learning*.

Patel, C., Patel, A., & Patel, D. (2012). Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study. *International Journal of Computer Applications*, *55*(10), 50–56. https://doi.org/10.5120/8794-2784

Pawar, N., Shaikh, Z., Shinde, P., & Warke, Y. P. (2018). Image to Text Conversion Using Tesseract. *International Research Journal of Engineering and Technology*, *516*, 516–519. www.irjet.net

Phadnis, A., Nimbalkar, N., Pagar, S., & Charumathi, K. S. (2020). *Aid for Blind People using IoT*. 314–320.

Ping, W., Peng, K., Gibiansky, A., Arik, S. O., Kannan, A., Narang, S., Raiman, J., & Miller, J. (2017). Deep voice 3: Scaling text-to-speech with convolutional sequence learning. *ArXiv Preprint ArXiv:1710.07654*.

Prenger, R., Valle, R., & Catanzaro, B. (2019). Waveglow: A flow-based generative network for speech synthesis. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3617–3621.

Rabbiirra, G. (2008). *Furtuu: seerluga afaan Oromoo, (Oromoo grammar), Kutta 9 fi 10*. Dhaabbata Maxxansiisaa Kurraaz Intarnaashinaal. https://books.google.de/books?id=zefwjgEACAAJ

Ragavi, K., Radja, P., & Chithra, S. (2016). Portable text to speech converter for the visually impaired. *Advances in Intelligent Systems and Computing*, *397*, 751–758. https://doi.org/10.1007/978-81-322-2671-0_71

Rajaram, S., Gupta, P., Andrassy, B., & Runkler, T. (2018). *Neural Architectures for Relation Extraction Within and Across Sentence Boundaries in Natural Language Text*. https://doi.org/10.13140/RG.2.2.36649.70245

Rajbongshi, A., Islam, M. I., Rahman, M. M., Majumder, A., Islam, M. E., & Biswas, A. A. (2020). Bangla optical character recognition and text-to-speech conversion using raspberry Pi. *International Journal of Advanced Computer Science and Applications*, *11*(6), 274–278. https://doi.org/10.14569/IJACSA.2020.0110636

Ramiah, S., Liong, T. Y., & Jayabalan, M. (2015). Detecting text based image with optical character recognition for English translation and speech using Android. *2015 IEEE Student Conference on Research and Development, SCOReD 2015*, 272–277. https://doi.org/10.1109/SCORED.2015.7449339

Rani, G. J., Kumar, T. P., Rama, G. S., Sidhardha, C., & Saritha, B. (2022). Raspberry Pi Based Reader for Blind People. *AIP Conference Proceedings*, *2418*, 5–8. https://doi.org/10.1063/5.0082133

Reichel, U., & Ptzinger, H. (2006). *Text Preprocessing for Speech Synthesis*.

Rithesh, K. (2019). *descriptinc/melgan-neurips*. https://github.com/descriptinc/%0Amelgan-neurips, 2019.

Ro, J. H., Stahlberg, F., Wu, K., & Kumar, S. (2022). Transformer-based Models of Text Normalization for Speech Applications. *ArXiv Preprint ArXiv:2202.00153*.

Roberts, L. (2020). *Understanding the Mel Spectrogram*. https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

Rokach, A., Berman, D., & Rose, A. (2021). Loneliness of the Blind and the Visually Impaired. *Frontiers in Psychology*, *12*. https://doi.org/10.3389/fpsyg.2021.641711

S., A., & G, H. (2015). Recognition of Historical Records Using Gabor and Zonal Features. *Signal & Image Processing : An International Journal*, *6*, 57–69. https://doi.org/10.5121/sipij.2015.6405

S, V., & A, S. (2015). Performance Comparison of OCR Tools. *International Journal of UbiComp*, *6*(3), 19–30. https://doi.org/10.5121/iju.2015.6303

Scotland, S. (2023). *Accessible formats Braille, Technology*. https://sightscotland.org.uk/articles/information-and-advice/accessible-formats

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R., Saurous, R. A., Agiomvrgiannakis, Y., & Wu, Y. (2018). Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2018-April*, 4779–4783. https://doi.org/10.1109/ICASSP.2018.8461368

Singh, G. (2021). *Understanding Architecture of LSTM*. https://www.analyticsvidhya.com/blog/2021/01/understanding-architecture-of-lstm

Skerry-Ryan, R. J., Battenberg, E., Xiao, Y., Wang, Y., Stanton, D., Shor, J., Weiss, R. J., Clark, R., & Saurous, R. A. (2018). Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. *35th International Conference on Machine Learning, ICML 2018*, *11*, 7471–7480.

Smith, R. (2005). *An Overview of the Tesseract OCR Engine*.

Sudharshan, G., Sowdeshwar, S., & Jagannath, M. (2022). Smart Glass with Multi-Functionalities for Assisting Visually Impaired People. *Journal of Physics: Conference Series*, *2318*(1). https://doi.org/10.1088/1742-6596/2318/1/012001

Sun, G., Zhang, Y., Weiss, R. J., Cao, Y., Zen, H., & Wu, Y. (2020). Fully-Hierarchical Fine-Grained Prosody Modeling for Interpretable Speech Synthesis. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2020-May*, 6264–6268. https://doi.org/10.1109/ICASSP40776.2020.9053520

Swain, D., & Sahoo, S. S. (2021). *Empowering the Visually Impaired Learners with Text-to-Speech-Based Solution BT - Machine Learning and Information Processing* (D. Swain, P. K. Pattnaik, & T. Athawale (eds.); pp. 565–573). Springer Singapore.

Tan, X., Qin, T., Soong, F., & Liu, T.-Y. (2021). A survey on neural speech synthesis. *ArXiv Preprint ArXiv:2106.15561*.

Tashoma, C. S. (2021). *Text to Speech Synthesizer for Afaan Oromo Using Deep Neural network*. http://etd.astu.edu.et/handle/123456789/1870

Tesema, W., & Tamirat, D. (2017). *Investigating Afan Oromo Language Structure and Developing Effective File Editing Tool as Plug-in into Ms Word to Support Text Entry and Input Methods*. *May*. www.pubicon.in

Tilahun, G. (1994). *Afaan Oromoo - the Oromo language and the latin alphabet*. https://www.africa.upenn.edu/Hornet/Afaan_Oromo_19777.html

Together Civil Society, O. (2020). *Country Background: Living With Visual Impairment In Ethiopia*. https://www.together-et.com/info-zone/country-background-living-with-visual-impairment-in-ethiopia/

Tolla, T. T. (2022). *Visually Impaired Children in some sites of Ethiopia.pdf*. https://journals.bdu.edu.et/index.php/bje/article/view/21/47

Tomás, O. (2020). *Differences between Autoregressive, Autoencoding and Sequence-to-Sequence Models in Machine Learning*. https://github.com/christianversloot/machine-learning-articles/blob/main/differences-between-autoregressive-autoencoding-and-sequence-to-sequence-models-in-machine-learning.md

Vae, Q. F., Prior, A. P., Sun, G., Zhang, Y., Weiss, R. J., Cao, Y., Zen, H., Rosenberg, A., Ramabhadran, B., & Wu, Y. (2020). *GENERATING DIVERSE AND NATURAL TEXT-TO-SPEECH SAMPLES USING A QUANTIZED FINE-GRAINED VAE AND AUTOREGRESSIVE PROSODY PRIOR*.

Vaithiyanathan, D., & Muniraj, M. (2019). Cloud based text extraction using google cloud vison for visually impaired applications. *Proceedings of the 11th International Conference on Advanced Computing, ICoAC 2019*, 90–96. https://doi.org/10.1109/ICoAC48765.2019.246822

Valente, J., Tekinerdogan, B., Voort, M. van der, & Saman, G. (2021). *Effect of Attention Mechanism in Deep Learning-Based Remote Sensing Image Processing*. https://doi.org/https://doi.org/10.3390/rs13152965

Wahab, M. N. A., Mohamed, A. S. A., Sukor, A. S. A., & Teng, O. C. (2021). Text Reader for Visually Impaired Person. *Journal of Physics: Conference Series*, *1755*(1). https://doi.org/10.1088/1742-6596/1755/1/012055

Wang, W., Xu, S., & Xu, B. (2016). First step towards end-to-end parametric TTS synthesis: Generating spectral parameters with neural attention. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, *08-12-Sept*(2013), 2243–2247. https://doi.org/10.21437/Interspeech.2016-134

Wang, Y., Stanton, D., Zhang, Y., Skerry-Ryan, R. J., Battenberg, E., Shor, J., Xiao, Y., Ren, F., Jia, Y., & Saurous, R. A. (2018). Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. *35th International Conference on Machine Learning, ICML 2018*, *12*, 8229–8238.

World Health Organization. (2019). *World report on vision*. https://www.who.int/publications/i/item/9789241516570

Wosho, M. K. (2020). *Text to Speech Synthesizer for Afaan Oromoo using Statistical Parametric Speech Synthesis*.

Yasuda, Y., Wang, X., & Yamagishi, J. (2019). Initial investigation of an encoder-decoder end-to-end TTS framework using marginalization of monotonic hard latent alignments. *ArXiv Preprint ArXiv:1908.11535*.

Yihun, S. G., & Belay, M. A. (2020). The challenges and opportunities of visually impaired students in inclusive education: The case of Bedlu. *Journal of Pedagogical Research*, *4*(2), 112–124. https://doi.org/10.33902/jpr.2020060437

Yohannes, M. G. (2011). *KAMISEE OROMO PHONOLOGY*. *May*, 1–105.

Yugesh, V. (2021). *Complete Guide To Bidirectional LSTM*. https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/

Zhang, B., Leitner, J., & Thornton, S. (2019). *Audio Recognition using Mel Spectrograms and Convolution Neural Networks*. http://noiselab.ucsd.edu/ECE228_2019/Reports/Report38.pdf

APPENDICES

Appendices A: Questionnaire for Mean Opinion Score Evaluation

Evaluator Information:

Name: _____ Occupation: _____

Native language: _____ Familiarity with Afaan Oromoo language: _____

Instructions:

Please listen to the provided speech samples generated by our Afaan Oromoo Text Reading model and evaluate them based on their Intelligibility and naturalness. Rate each sample on a scale from 1 to 5, where 1 represents the lowest score and 5 represents the highest score.

Speech Sample Evaluation:

Sample 1:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 2:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 3:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 4:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 5:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 6:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 7:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 8:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 9:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

Sample 10:

MOS Score: (1-5) Intelligibility: _____, Naturalness: _____

General Questions:

How would you rate the overall Intelligibility of the synthesized speech samples?

Excellent ☐ Good ☐ Average ☐ Poor ☐ Very Poor ☐

How would you rate the overall naturalness of the synthesized speech samples?

Extremely Natural ☐ Natural ☐ Somewhat Natural ☐

Artificial ☐ Very Artificial ☐

Did you notice any word omissions or inconsistencies in the synthesized speech samples?

Yes ☐ No ☐

If you answered "Yes" to the previous question, please provide details regarding the word omissions or inconsistencies you observed.

_____

How does the synthesized speech compare to previous text-to-speech synthesis approaches for Afaan Oromoo that you may be familiar with?

Significantly better ☐ slightly better ☐

SimilarSlightly worse ☐ significantly worse ☐

Do you have any additional comments or feedback regarding the synthesized speech samples?

_____

Thank you for participating in the evaluation of our TTS model! Your feedback is valuable and will help us further improve the system.

Appendices B: MOS Evaluation Samples List

Sample 1

Oomisha Kuduuraa Fi Muduraa Irraa Galiin Doolara Miiliyoona 514 Ol-Argamuun Ibsame

Sample 2

Finfinnee, Caamsaa 6, 2015 (FBC).
Naannoo Hararitti "Nageenyaaf nan fiiga" mataa duree jedhuun fiigichii daandirraa gaggeefameera.

Sample 3

Baatiwwan saglanitti oomishaa Ililliirraa galiin Doolara miiliyoona 459.46 argachuuf karoorfamuun eerameera.

Sample 4

Baankiin Misoomaa Itoophiyaa bu'aa birrii biliyoona 3.9 argachuu isaa beeksise.

Sample 5

Inistiraakter Daani'eel Gabramaariyaam Leenjisaa Yeroo Waaliyaawwanii Ta'uun Muudaman

Sample 6

Afrikaa Bahaatti Miidhamtoota Deeggaruuf Kan Oolu € Biiliyoonni 2.4 Argamuun Ibsame

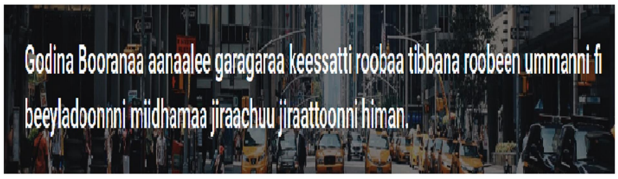
Sample 7

Yeroo ammaa biyyoonni miseensa Gamtaa Afrikaa cimuun waltajjiiwwan idil addunyaarratti sagalee murteessaa qabaachaa jiraachuu eeruun, dhimmoota ardichaafi uummatasaa fayyadan irratti ejjennoo waloo beeksisuuniifi Afrikaaf dhaabbachuun fageenya dheeraa adeemuu addeessaniiru.

Sample 8

MM Abiy Ahimad Guyyaa Afrikaa Sababeeffachuun Erga Dabarsan, Ergaasaaniin, hoggantoonni Afrikaa tokkummaa abbootii duraaniin eegalame daran akka cimsan waamicha dhiyeessaniiru.

Sample 9



Sample 10

Ibsichaan Lakkoofsi namoota yaala wallaansa cimaa keessa galanii 800 ga'uu himaniiru tooftaan ittisaa furtuu ta'e talaallii ta'uu Dr. Liyaan eeraniiru

Appendices C: Tacotron2 with HIFI-GAN Evaluators Sample Result

Appendices C.1: Intelligibility Evaluation Result

Samples	Eval_1	Eval_2	Eval_3	Eval_4	Eval_5	Eval_6	Eval_7	Eval_8	Eval_9	Eval_10
Sample1	4	4	5	5	5	5	4			
Sample2	4	4	4	4	4	4	4			
Sample3	4	5	4	5	5	5	5			
Sample4	4	5	5	5	4	5	5			
Sample5	4	5	5	5	4	4	5			
Sample6	4	4	3	5	4	5	4			
Sample7	3	4	3	3	4	3	4			
Sample8	4	4	5	4	5	4	4			
Sample9	5	5	5	5	5	5	5			
Sample10	5	4	5	4	5	4	4			
Average Intelligibility = 4.39										

Appendices C.2: Naturalness Evaluation Result

Samples	Eval_1	Eval_2	Eval_3	Eval_4	Eval_5	Eval_6	Eval_7	Eval_8	Eval_9	Eval_10
Sample1	5	5	4	4	5	4	5			
Sample2	4	4	4	4	4	4	4			
Sample3	5	5	4	4	4	4	5			
Sample4	5	5	4	5	5	5	4			
Sample5	5	5	4	4	5	4	5			
Sample6	5	5	2	5	4	5	4			
Sample7	4	3	3	4	3	4	3			
Sample8	5	5	5	4	4	4	4			
Sample9	5	5	5	5	4	5	5			
Sample10	5	5	3	4	4	4	5			
Average Naturalness = 4.35										

Appendices D: Tacotron2 with WaveGlow Evaluators Sample Result

Appendices D.1: Intelligibility Evaluation Result

Samples	Eval_1	Eval_2	Eval_3	Eval_4	Eval_5	Eval_6	Eval_7	Eval_8	Eval_9	Eval_10
Sample1	4	4	5	5	4	4	4			
Sample2	4	3	4	4	3	3	4			
Sample3	4	5	4	5	4	5	4			
Sample4	4	5	4	5	4	5	5			
Sample5	4	5	5	3	4	5	4			
Sample6	4	4	3	4	4	4	4			
Sample7	3	3	3	3	4	3	3			
Sample8	4	4	4	4	5	4	4			
Sample9	5	4	5	5	4	4	5			
Sample10	5	4	4	4	5	4	4			
Average Intelligibility = 4.11										

Appendices D.2: Naturalness Evaluation Result

Samples	Eval_1	Eval_2	Eval_3	Eval_4	Eval_5	Eval_6	Eval_7	Eval_8	Eval_9	Eval_10
Sample1	4	4	4	4	3	4	4			
Sample2	4	3	4	4	4	4	4			
Sample3	5	5	4	3	4	3	5			
Sample4	3	5	4	3	4	3	3			
Sample5	4	5	4	4	5	5	4			
Sample6	4	4	2	4	4	3	4			
Sample7	4	4	3	4	3	4	4			
Sample8	3	4	4	4	4	4	3			
Sample9	4	4	4	4	4	4	4			
Sample10	4	5	3	4	4	4	4			
Average Naturalness = 3.87										

Appendices E: How to Calculate MCD Objective Evaluation

Appendices E.1: Function to Calculated MCD

```
import librosa
import numpy as np

def calculate_mcd_PL(source_path, target_path, sr):
    # Load source and target waveforms
    source_waveform, _ = librosa.load(source_path, sr=sr)
    target_waveform, _ = librosa.load(target_path, sr=sr)

    # Extract Mel Cepstral Coefficients (MCCs) from source and target
    source_mcc = librosa.feature.mfcc(y=source_waveform, sr=sr, n_mfcc=40)
    target_mcc = librosa.feature.mfcc(y=target_waveform, sr=sr, n_mfcc=40)

    # Align the source and target MCCs by resizing
    source_mcc_aligned = np.resize(source_mcc, target_mcc.shape)

    # Calculate the root mean square error (RMSE) between source and target MCCs
    rmse = np.sqrt(np.mean((source_mcc_aligned - target_mcc) ** 2))

    # Calculate MCD as a percentage of the dynamic range
    percentage_mcd = (rmse / np.ptp(target_mcc)) * 100

    return percentage_mcd
```

Appendices E.2: How to use MCD Function

```
source_path = "Generated_audio.wav"
target_path = "Ground_truth_audio.wav"
sr = 22050  # Sample rate of the audio

percentage_mcd = calculate_mcd_PL(source_path, target_path, sr)
print("Percentage MCD: {:.2f}%".format(percentage_mcd))
```