Addis Ababa University
College of Natural Sciences

Transformer-based Machine Translation System Model from Geez to Tigrigna

Aberash Berhe Berhanu

A Thesis Submitted to the Department of Computer Science in Partial Fulfilment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia
June 2024

Addis Ababa University

College of Natural Sciences

Aberash Berhe

Advisor: Dr. Yaregal Assabie (PhD)

This is to certify that the thesis prepared by Aberash Berhe, titled: *Transformer-based Machine Translation System Model from Geez to Tigrigna* and submitted in partial fulfilment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

	Name	Signature	Date
Advisor:	Yaregal Assabie (PhD)		20/09/2024
Examiner:	Dida Midekso (PhD		17 Sept'24
Examiner:	Mesfin Kifle (PhD)		17-09-24

Abstract

This thesis presents the first attempt at building a transformer-based neural machine translation system for the language pair of Geez and Tigrigna. Geez and Tigrigna are closely related Semitic languages, with Geez being the liturgical language of the Eritrean and Ethiopian Orthodox churches, and Tigrigna being a widely spoken language in Eritrea and parts of Ethiopia. Due to the lack of publicly available parallel corpora for this language pair, the thesis describes the manual collection and curation of a new Geez-Tigrigna parallel dataset, which consists of 10,362 sentence pairs. This process is detailed as it proved to be a laborious and time-consuming task given the limited availability of translated text between the two languages.

The architecture of the proposed neural machine translation system is based on the transformer model, which has shown state-of-the-art performance on many language pairs. To address the challenges of translating between low-resource languages like Geez and Tigrigna, an alignment-based approach is integrated into the standard transformer architecture. This alignment mechanism aims to better capture the relationships between source and target language elements during the translation process. The word-level alignments between the parallel sentences are done manually.

Experiments are conducted to compare the performance of attention-based recurrent neural network model, a standard transformer model, and the proposed alignment-augmented transformer model. The results show that the standard transformer model achieved a BLEU score of 54%, outperforming the RNN model, which had a BLEU score of 46%. Further improvements were made by integrating the alignment mechanism into the transformer architecture, resulting in an alignment-augmented transformer model that achieved a BLEU score of 63%.

These findings demonstrate the feasibility of building neural machine translation systems for low-resource language pairs like Geez and Tigrigna, and that the proposed alignment-based modifications to the transformer architecture can lead to significant improvements in translation quality compared to the standard transformer model.

Keywords: Machine Translation, Neural Machine Translation, Transformer, Encoder-Decoder model, Alignment

Acknowledgements

First and foremost, I would like to thank God for providing me with the opportunity, strength, and perseverance to undertake and complete this thesis.

Next, I would like to express my deepest gratitude to my advisor, Dr. Yaregal Asabie (Ph.D), for his solid guidance, support, and invaluable insights throughout the duration of this thesis. His expertise and constructive feedback have been instrumental in shaping this thesis.

I extend my sincere appreciation to the University for giving me a scholarship to attend my MSc. study. I would like to acknowledge the contributions of the Geez and Tigrigna language experts, translators, and community members who generously shared their knowledge, resources, and time to assist in the data collection and curation process.

Additionally, I would like to thank my husband, Kibrom Gebrehiwot, for his understanding and support throughout this process. I would also like to extend my heartfelt appreciation to my brother, Mehari Berhe, and my extraordinary sister, Mulu Berhe, for their unwavering support and dedicated belief in me. Their presence and support have been a constant source of strength and have helped me overcome the challenges encountered during this process.

Without the blessings of God, the guidance of my advisor, the support of my friends and family, and the love of my husband, the completion of this thesis would not have been possible. I am truly grateful to all of you.

Dedication

This thesis is dedicated to the Ethiopian Orthodox Church, a respected institution that has played a profound role in preserving and advocating Geez language. For centuries, the Ethiopian Orthodox Church has been the guardian of Geez language, ensuring its continued use as the liturgical language of the faith.

Table of Contents

List of Tables

List of Figures

Acronyms and Abbreviations

AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
BLEU	BiLingual Evaluation Understudy
BPE	Byte Pair Encoding
CBMT	Corpus-based machine translation
CNN	Convolutional Neural Network
EBMT	Example-based Machine Translation
EOS	End-of-sentence
FFN	Feed Forward Neural network
GRU	Gated Recurrent Unit
HMT	Hybrid Machine Translation
LSTM	Long Short-Term Memory
MT	Machine Translation
NLP	Natural Language Processing
NLTK	Natural Language Toolkits
NMT	Neural Machine Translation
OOV	Out of Vocabulary
PB	Phrase Based
POS	Part of Speech
RBMT	Rule Based Machine Translation
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Networks
SMT	Statistical Machine Translation
SOS	Start-of-Sentence
TNN	Transformer Neural Network
ULM	Unigram Language Model

Chapter 1: Introduction

1.1 Background

Language is a fundamental element of human behaviour that plays a crucial role in our lives. It serves as the most effective tool for preserving and transferring information and knowledge across generations. When expressed in a written form, language acts as a reliable means of recording and transmitting valuable insights over an extended period [1]. As the advance in technology increases, the world becomes more connected and the demand for communication among the people increases as a result. The new technology and the globalization stimulate the desire of human beings to understand the languages of each other. So, the idea of translating the languages of individuals among one another became a research idea for several years. Hence, the language barrier among the speakers of different languages is solved. This is by a means of natural language processing.

Natural Language Processing (NLP) is a field of Artificial Intelligence (AI) that enables computers to explore and understand human language, both written and spoken [2]. It was formulated to build software that generates and comprehends natural languages so that a user can have natural conversations with a computer instead of through programming or artificial languages. NLP employs computer algorithms and artificial intelligence to enable computers to recognize and respond to human communication. Technically, the main task of NLP would be to program computers for analysing and processing huge amount of natural language data [3, 4].

The demand for language translation has greatly increased in recent times due to increasing cross-regional communication and the need for information exchange. Most material needs to be translated, including scientific and technical documentation, instruction manuals, legal documents, textbooks, publicity leaflets, newspaper reports etc. Some of this work is challenging and mostly it is tedious and repetitive and requires consistency and accuracy. It is becoming difficult for professional translators to meet the increasing demands of translation. In such a situation the machine translation can be used as a substitute [5].

Machine Translation (MT) is an automatic translation of text from a source language to a target language. The ultimate aim of MT is to reach a level of proficiency where the system can independently produce translations of the finest quality, without requiring human intervention.

Basically, every machine translation system requires automated programs for translation, dictionaries, and grammar to support translation. Since its birth a lot of MT techniques have been developed. The commonly used MT approaches are rule-based, and corpus-based approaches. A third category, which is hybridized of the two approaches is also used for MT tasks.

More recently, Neural Machine Translation (NMT) which models direct mapping between source and target languages with deep neural networks has achieved a big breakthrough in translation performance and become the de facto paradigm of MT [6, 7]. NMT has shown remarkable progress with production systems now being deployed to end-users. It has the ability to learn from each translation task and improve upon each subsequent translation. In addition, NMT can recognize patterns in the source material to determine a context-based interpretation that can predict the likelihood of a sequence of words. It's a breakthrough because it does not have to be supervised by humans. It involves creating large neural networks that allow the system to learn and operate independently. In NMT massive datasets of translated sentences are used to train a model capable of translating between any two languages. The architecture behind NMT is composed of two neural networks used together in tandem to create an encoder-decoder structure. Attention mechanisms have recently been developed to further increase the accuracy of these models.

1.2 Motivation

Ethiopia possesses a rich linguistic variety, with over 80 distinct languages spoken across its diverse regions [8]. These languages traverse multiple language families, including the Semitic, Cushitic, Omotic, and Nilotic groups. Most of the Ethiopian languages are under the Semitic, Cushitic or Omotic groups and some are under the Nilotic group. The Semitic languages are spoken primarily in the northern and central parts of Ethiopia. Geez and Tigrigna are examples of Semitic languages mainly spoken in northern Ethiopia (Tigray) and Eritrea. Geez was the literary language in Ethiopia from the very early period [9] and it is mentioned with the ancient civilization of Ethiopia. Geez has its own unique writing script or alphabet called '*Fidel*', which has also influenced the development of the Tigrigna alphabet. There are lots of ancient and very important Geez written materials located in Ethiopia. The deep cultural and linguistic ties between Geez and Tigrigna present a fascinating opportunity. As closely related languages, these two systems share a wealth of similarities that can be utilized to facilitate mutual understanding and access to the vast abundance of Geez-language literature. This valuable significance encompasses a diverse array of

subjects, from philosophy and religion to medicine, astronomy, and history - all waiting to be unlocked and shared with the Tigrigna-speaking community [10, 11]. It is this rich cultural and linguistic heritage that serves as the driving force behind this thesis. By developing robust machine translation capabilities between Geez and Tigrigna, we aim to bridge the gap between these sister languages, empowering the people of Ethiopia to fully engage with their shared literary and intellectual legacy. Because of this, there is a big initiative in translating those documents written in Geez.

1.3 Statement of the Problem

The field of machine translation has witnessed the application of diverse methodologies in developing systems for a wide array of language pairs. However, the bulk of these efforts have centred around pairing non-Ethiopian languages, often with English, rather than focusing on the unique linguistic landscape within Ethiopia itself. This gap in research attention is particularly evident when it comes to the challenge of translating between the numerous languages spoken across the Horn of Africa nation. While the international community has made significant advances in advancing machine translation technology, the systems readily available for Ethiopian languages remain frustratingly limited. Despite the lack of existing machine translation systems for Ethiopian languages, a growing group of researchers have taken on the challenge. They are determined to develop robust machine translation capabilities to support the rich linguistic diversity of Ethiopia.

Genet Worku [12] has developed an encoder-decoder translation model for Geez-Amharic translation using NMT with an attention mechanism. To handle long-term dependencies in lengthy sentences, an attention mechanism is employed, which focuses on relevant information in the input sentence when generating each word in the target sentence. The model is developed using OpenNMT, and its translation achieves a BLEU score of 15.4%.

Samson Tesfaye [13] worked on a hybrid approach for machine translation from Geez to Amharic language and get a BLEU score result of 18.62%.

Akubazgi GebreMariam [14] has developed a translation system for Amharic to Tigrigna using a hybrid approach. The study established syntactic reordering approach which aligns the structural arrangement order of words in the source sentence to be more similar to the target sentences. The

reordering rules were developed to fulfil for both simple and complex Amharic sentences that have difference in the structural arrangement order of words. Two experiments were conducted, each using a different approach. The first experiment utilized a statistical approach and achieved a BLEU score of 7.02%. The second experiment employed a hybrid approach and yielded a BLEU score of 17.47%.

Adopting neural machine translation approach has the potential for enhancing the performance of translation systems targeting Ethiopian language pairs. Neural machine translation extracts fully automated algorithms inspired by the architecture of the human brain, allowing for more sophisticated and adaptive translation capabilities compared to traditional rule-based or statistical methods. From the researchers' perspective, there has been no prior work specifically focused on developing neural machine translation systems between Geez and Tigrigna languages. This represents a significant gap that currently needs to be addressed in order to facilitate greater accessibility and exchange between the rich literary and cultural traditions associated with these two closely related Ethiopian languages.

1.4 Objectives

General Objective

The general objective of this thesis is to develop a machine translation from Geez to Tigrigna using transformer based neural machine translation approach.

Specific Objectives

The specific objectives of this study are:

- Reviewing related literatures on machine translation and its approaches
- Exploring the language structures of both the source and target languages
- Collecting data and preparing parallel corpus
- Developing the design of the Geez to Tigrigna machine translation
- Implementing the architecture using python code
- Evaluating the performance of the translation

1.5 Methodology

To ensure the successful implementation of the thesis, comprehensive methodological approach will be followed, and the core components and foundational elements of this implementation strategy are as follows:

Literature Review

As the initial step of the research, a comprehensive review of the diverse range of machine translation approaches currently available will be assessed. This extensive literature review will be conducted with the goal of developing a thorough understanding of the various methodologies that have been proposed and implemented in the field of machine translation. During the course of this review, the relative advantages and disadvantages of the existing machine translation approaches will be closely examined. Particular attention will be paid to investigating the linguistic characteristics and behaviours of both Geez and Tigrigna languages, and how these language-specific factors may impact the performance and suitability of different translation techniques. This thorough exploration of the machine translation landscape, coupled with a detailed analysis of the source and target language properties, will provide a strong foundation for subsequently selecting and adapting an appropriate translation approach for the Ethiopian language pair at the centre of this thesis.

Data Collection

To successfully implement this transformer based Geez to Tigrigna neural machine translation, a parallel corpus for the Geez and Tigrigna languages is mandatory. So that data collection from different relevant materials will be one of the essential steps. Therefore, parallel sentences from different religious books mainly the Bible, grammar books [15], other books such as dictionary prepared at Holy Trinity Theological College [16], prayer books, and from some useful contents on the social media is going to be gathered. Once a parallel corpus for the two languages is prepared, the preprocessing tasks can be handled.

System design and prototype development

The overall architecture of the transformer based Geez to Tigrigna neural machine translation will be described using block diagrams. The encoder-decoder transformer framework concatenated with an alignment context will be considered in the development of the model. In order to conduct

experiments, different training hyper parameters are going to be chosen for the model. These parameters encompass the embedding dimension, the number of encoder and decoder layers, the number of attention heads, the hidden dimension of the feed forward network, the dropout rate, the learning rate, the batch size, and the number of epochs. This process primarily aims to train the system. Python programming will be used for coding.

Evaluation

After training the system for the specified epoch number, the quality of the translation is assessed, and the performance of the translation is measured by using BLEU score.

1.6 Scope and Limitations

This work focuses on developing a transformer based neural machine translation model for translating from Geez to Tigrigna. The reverse translation direction, from Tigrigna to Geez, is not included as part of this study. The corpus that was developed and used to train the model is primarily composed of content from religious documents and thus the training data lacks diversity.

1.7 Application of Results

This work can assist Tigrigna speakers in understanding Geez language, thereby aiding historians, researchers, and archaeologists in studying ancient Ethiopia and its civilization. Investigators interested in Geez language and other ancient literatures written in Geez, such as those on medicine, philosophy, astronomy, politics, agriculture, and religious scripts will also find this work valuable. Additionally, the research may serve as a teaching and learning guide. Linguistic professionals can leverage this initial system to conduct further research on Geez language.

1.8 Organization of the Thesis

The organization of the rest of the thesis is structured as follows. Chapter Two will present a literature review which includes the description of both Geez and Tigrigna languages, an overview of MT, an attention mechanism, MT approaches, and evaluation of MT Systems. Chapter Three is all about different related works or tasks done on machine translation and it is structured based on the MT approaches applied, by listing the different approaches as a sub-title and presenting example of papers that applies each method. Chapter Four will describe the architecture of the Geez to Tigrigna neural machine translation based on transformer NMT. The detail of the components of the translation and the algorithms are going to be discussed. Chapter Five deliberates about the experiment and its results; the results of each experiment will then be recorded. Chapter Six will cover the conclusions and recommendations for future works.

Chapter 2: Literature Review

2.1 Introduction

This chapter discusses the behaviours of both the source (Geez) and target (Tigrigna) languages, machine translation, the primary approaches to machine translation (which are rule-based and corpus-based), and the various sub-divisions of machine translation. It also introduces an attention mechanism which is the critical component in many modern neural machine translation models. In addition, the chapter covers the historical development of machine translation, the strengths, and limitations of rule-based versus corpus-based approaches. This review provides the necessary background and context to situate the original research and contributions presented in the subsequent chapters of the thesis.

2.2 Geez Language

2.2.1 Overview of Geez Language

Ethiopia is a country of ancient civilization and thousands of years of literary history with its own alphabet, numerals, calendar, writings and so on. Stone inscriptions, book of the Old Testament, the earliest New Testament, royal chronicles, and various religious manuscripts are found written in Ethiopic language [17]. The country is one of the classical countries that have their own alphabet and writing system. Using this indigenous alphabet, Ethiopians have developed their own writing tradition and produced many works of literature. Accordingly, Ethiopia is a country with plenty of classical magnificently enlightened manuscripts and literature. In Geez language, many classical works were recorded including secular writings, several indigenous original manuscripts have been written. These include historical, theological, philosophical, astronomical, medical, and political works [18, 19].

In the Ethiopian Orthodox Tewahido Church, Geez is still the language of liturgy and other religious, social, and philosophical services such as *qəne*. It remained the literacy language long after it ceased to be a spoken vernacular after the decline of the Aksumite Empire. Geez was replaced by Amharic as a written media outside the church by the second half of the 19th century, during the reign of Tewodros II [20].

Geez adopted its scripts from Sabean (the language of the Aksumite Kingdom) and it is the only language in Ethiopia which has its own alphabets. The other Ethiopian languages such as Tigrigna and Amharic adopted these alphabets fully from Geez. In Geez there are 26 basic characters with 7 columns each. The total number of the characters is 182. Ethiopic language has its own unique numbering system which is completely different from the worldwide used Arabic numbers. The way used to read is also unique [1, 21].

The list of Geez alphabets includes ሀ(*hä*), ለ(*lä*), ሐ(*ḥä*), መ(*mä*), ሠ(*ṣä*), ረ(*rä*), ሰ(*sä*), ቀ(*qä*), በ(*bä*), ተ(*tä*),ኀ(*ḫä*), ነ(*nä*), አ(*aä*), ከ(*kä*), ወ(*wä*), ዐ(*ạä*), ዘ(*zä*), የ(*yä),* ደ(*dä*), ገ(*gä),* ጠ(*ṭä*), ጰ(*ṗä*), ጸ(*tsä*), ፀ(*tṣä*)፣ ፈ(*fä*)and ፐ(*pä*) [22]. Each of the columns of the alphabet are labelled as first order (*gəzə*), second order (*kaʔəbə*), third order (*saləsə*), fourth order (*rabəʔə*), fifth order (*ḥaməsə*), sixth order (*sadəsə),* and seventh order (*sabəʔə*) of alphabets. The list of these alphabet is given in Appendix A. In Geez language there are letters that have similar sounds. Even though they have similar sounds, the letters are different in shape orthographically. These include *hä*(ሀ,ሐ,ኀ), *se*(ሠ, ሰ), *a*(አ, ዐ) and *tsä*(ጸ,ፀ). These letters have unique importance in Geez script. That means they make words having different meanings. The following list shows some examples of Geez words pronounced the same with different writing and different meaning. መሀረ(*mäharä* 'he taught') and መሐረ(*mäḥarä* 'he forgave'), ኀለየ(*ḫaläyä* 'he thanks') and ሐለየ(*ḥaläyä* 'he thought'), ሠረቀ(*ṣäräqä* 'went out') and ሰረቀ(*säräqä* 'he stole'), አመት(*ʔamätə* 'servant') and ዓመት(*ʔạmätə* 'year'), ሰዐለ(*säʔạlä* 'he draws')and ሰአለ(*säʔəalä* 'he begged'). There are also words have different meanings when they are read smoothly or stressed. For instance, the word *mäkanə* is to mean a place when it is read smoothly, and it has the meaning of barren when it is read the other way. This nature of the language challenges the machine translation task.

2.2.2 Word Class in Geez Language

Nouns

Nouns are words that are used to identify or address an object. All objects having a definite and indefinite volume are called by a name. Nouns typically represent a person, place, thing, or idea. They are often used as the subject or object of a sentence. Examples include እግዚአብሔር (*ʔəgəziabəherə* 'God'), ሰብእ (*säbəʔə* 'Human'), እንስሳ (*ʔənəsəsa* 'animal'), መሬት (*märetə* 'earth'), ነፋስ (*näfasə* 'wind'), እሳት (*ʔəsatə,* 'fire'), ማይ (*mayə* 'water'), አዳም (*adamə* 'adam'), ኢየሩሳሌም (*ʔəyärusalem 'Jerusalem'*), ፍቅር (*fəqər 'love'*) etc [15].

Like other semitic languages, most Ethiopic words can be boiled down to three, or sometimes two or four consonantal roots. All Ethiopic words have gender, however, unlike other semitic languages, Geez gender rules are lax. Aside for nouns for human beings Geez nouns have variable genders. E.g a noun like ቤት (*bätə* 'house') will appear in some texts as masculine and in others as feminine. It completely depends on the author's preference. Geez words also do not have a definite or indefinite article [23]. For example, the word ቤት (*bätə* 'a house, the house') and ብእሲ (*bəʔəsi* 'a man, the man').

Geez plural nouns:

- Many male human nouns have the አን (*ʔanə*) ending. The plural form of መዘምር (*mäzämərə* 'singer') is መዘምራን (*mäzäməranə* 'singers').
- Nouns for both male and female have the አት (*ʔatə)* ending. An example for this can be ነቢያት (*näbəyatə* 'prophets') for ነቢይ (*näbəye* 'prophet') and ንግሥታት (*nəgəṣətatə* 'queens') for ንግሥት (*nəgəṣətə* 'queen').
- Biconsonantal words typically have the አው (*ʔäw*) ending. Example is አበው (*ʔabäw* 'fathers') for አብ (*ʔabə* 'father').
- All others have broken plurals, this means, there is an internal vowel change in the word. There is no real way to predict broken plurals, but after a while they will become intuitive. To consider some examples, the plurals for ሀገር (*hagärə* 'country') and ቤት (*betə* 'house') are አህጉር (*ʔahəgurə* 'countries') and አብያት (*ʔabəyatə* 'houses') respectively.
- There are also some completely irregular plurals as in the plurals of ብእሲ (*bəʔəsi* 'man') and ብእሲት (*bəʔəsitə* 'woman') whose plurals are ሰብእ (*säbəʔə)* or እደው(*ʔədäwə)* 'men') and አንስት (*ʔanəsətə* 'women') correspondingly.

Adjectives

Most adjectives in Geez, as in other semitic languages are verbal participles. A smaller number are denominal. They appear in a few different patterns. Adjectives always reflect the number, gender, and case of the nouns they modify. Adjectives usually appear after the nouns they modify [23]. The pattern most adjectives follow is shown in Table 2.1 though there are many irregular forms.

Table 2.1 Patterns of Most Adjectives of Geez Language

	Male	Female
Singular	-	-ት(*tə*)
Plural	-ኣን(*ʔanə*)	-ኣት(*ʔatə*)

Examples to illustrate this could be: ኣበድ ወልድ (*ʔabədə wäləd*ə ‘a lazy boy’), ራትዕ ነብይ (*ratəʔə näbəyə* righteous prophet), ክብርት ንግሥት (*kəbərətə nəgəṣətə* ‘mighty queen’), ቅዱሳን ኣበው (*qədusanə ʔabäwə* ‘saint fathers’).

Personal Pronouns

Geez has ten distinct pronouns as illustrated in [17] that act as linking verbs. These pronouns are listed in Table 2.2.

Table 2.2 List of Personal Pronouns in Geez Language

Category	Singular	Plural
First person	ኣነ (*ʔanä* ‘I’)	ንሕነ (*nəḥənä* ‘We’)
Second person	ኣንተ (*ʔanətä* ‘You’) (Male)	ኣንትሙ (*ʔanətimu* ‘You’)
	ኣንቲ (*ʔanəti* ‘You’) (Female)	ኣንትን (*ʔanətənə* ‘You’)
Third person	ውእቱ (*wəʔətu* ‘He’) and (‘It’)	እሙንቱ (*ʔəmunətu* ‘They’) (Male)
	ይእቲ (*yəʔəti* ‘She’) and (‘It’)	እማንቱ (*ʔəmanətu* ‘They’) (Female)

Linking Verb

The independent forms of personal pronouns act as linking verbs when they appear after a noun as in the cases of ነብይ ኣነ (*näbəyə ʔane* ‘I am a prophet’), ነብያት ንሕነ (*näbəyat nəḥənä* ‘we are prophets’), ካህን ኣንተ (*kahənə ʔantä* ‘you are a priest’), ካህናት ኣንትሙ (*kahənatə ʔanətimu* ‘you are priests’), እም ኣንቲ (*ʔəmə ʔanəti* ‘you are a mother’), እማት ኣንትን (*ʔəmat ʔanətənə* ‘you are mothers’), ኣብ ውእቱ (*ʔabə wəʔətu* ‘he is a father’), ኣበው እሙንቱ (*ʔabäwə ʔanətimu* ‘they are fathers’), ብእሲት ይእቲ (*bəʔəsitə yəʔət* ‘she is a woman’), ኣንስት እማንቱ (*ʔanəsətə ʔəmanətu* ‘they are women’).

- Personal pronouns act as linking verbs when they appear after a noun.

  ነቢይ አነ (*näbəyə ʔane* ‘I am a prophet’), ነቢይ አንተ (*näbəyə ʔantä* ‘you are a prophet’)

- The pronouns can appear twice for emphasis, this is at both ends of a sentence.

  አንተ ነቢይ አንተ (*ʔantä näbəyə ʔantä* ‘you are surely a prophet’)

  ውእቱ በጊዕ ውእቱ (*wəʔətu bägəʔə wəʔətu* ‘it is definitely a sheep’)

- The third person masculine pronoun ውእቱ (*wəʔətu*) appear as the sole linking verb irrespective of person, number, or gender of the actual subject in certain texts.

  አነ ነቢይ ውእቱ (*ʔane näbəyə wəʔətu* ‘I am a prophet’)

  አንተ ነቢይ ውእቱ (*ʔantä näbəyə wəʔətu* ‘you are a prophet’)

Demonstrative Pronouns

Demonstrative pronouns, like the nouns, are marked for gender and number. There are two pairs of pronouns for ‘this’ and ‘these’. The list of the demonstrative pronouns is given in Table 2.3. Examples: ዝንቱ ብእሲ ውእቱ ነቢይ (z*ənətu bəʔəsi näbəyə wəʔətu* ‘this man is a/the prophet’), ዛቲ ብእሲት ይእቲ እም (*zati bəʔəsit yəʔəti ʔəmə* ‘this woman is a/the mother’).

Table 2.3 Demonstrative Pronouns in Geez Language

Masculine	Feminine
ዝንቱ (z*ənətu* ‘this’)	ዛቲ (*zati* ‘this’)
ዝ (z*ə* ‘this’)	ዛ (*za* ‘this’)
እሉ (*ʔəlu* ‘these’)	እላ (*ʔəla* ‘these’)
እሎንቱ (*ʔəlonətu* ‘these’)	እላንቱ (*ʔəlanətu* ‘these’)
ውእቱ (*wəʔətu* ‘that’)	ይእቲ (*yəʔəti* ‘that’)
እሙንቱ (*ʔəmunətu* ‘those’)	እማንቱ (*ʔəmanətu* ‘those’)

Verb

Verbs, as the primary part of speech that convey actions, states, or events. Typically, they consist of a core element and additional affixes that denote tense, mood, aspect, and agreement with both the subject and object [15]. Geez verbs are primarily triliteral, meaning they consist of three root letters. However, there is also a significant number of quadrilateral verbs with four letters. A small portion of verbs have either two or five root letters. These verbs can be classified into three broad categories or stems, which can be thought of as vowel templates with fixed root-letters. Unlike in other Semitic languages, these stems are not derived from each other, and most roots only appear in one stem [23]. Examples are: ሞተ (*mote* 'to die'), ሐየወ (*ḥaye*wä 'to live'), ሐየለ (*ḥayelä* 'to be strong'), ከፈለ (*käfälä* 'to divide'), ፈነወ (*fänäwä* 'to send').

Adverb

Adverbs are words that give additional meaning to verbs. The job of adverbs is to tell the verb's place, time, degree, and so on by giving information how, why, where … etc. about the verb. Mostly Geez adverbs come after the verb they modify. The words ዮም (*yomə* 'today'), እሙነ (*ʔəmunä* 'surely'), ዝየ (*zəye* 'here'), ፍጡነ (*fəṭunä* 'fastly') are examples of adverbs. When we see the sentence "እሙነ ይመጽእ ንጉሥነ" (*ʔəmunä yəmätsəʔə nəguṣənä* 'our king will come surely'), the word እሙነ (surely) is emphasizing the verb ይመጽእ (will come).

Preposition:

Geez prepositions stand as independent words with three very important exceptions:

- በ (*bä* 'in/into/by'), example: በቤት (*bäbetə* 'in the house')
- ለ (*lä* 'for/to'), example: ለቤት (*läbetə* 'to the house')
- እም (*ʔəmə* 'from/out of'), example: እምቤት (*ʔəməbetə* 'from the house')

Table 2.4 lists some of the important prepositions in Geez language.

Table 2.4 List of Important Prepositions in Geez

List of Preposition	Meaning	List of Preposition	Meaning
ኀበ (*ḫabä*)	with, near, by	ምስለ(*məsəlä*)	in the company of
ውስተ(*wusəte*)	within, inside	እንተ(*ʔənətä*)	by way of, through

ዲበ(*dibä*)	on, over	በእንተ(*bäʔənətä*)	about, concerning
ላዕለ(*laʔəlä*)	upon, concerning	ድኅረ(*dəḫərä*)	behind, after
ታሕተ(*taḥətä*)	Under	ቅድመ(*qədəmä*)	in front of, before
ከመ(*kämä*)	like, similar to	እስከ(*ʔəsəkä*)	until, up to, as far as

The preposition ባ(*ba* 'by, with, in') as depicted in Table 2.5 has several important functions. The negative particle with this preposition is አል(*ʔalə*), thus forming አልቦሙ (*ʔaləbomu* 'not in them'), አልብነ (*ʔaləbənä* 'not in us') etc. The most important function of this preposition is with the 3rd person masculine and singular suffix which indicates existence (there is…, there are…).

Table 2.5 Function of the Preposition 'ba' in Geez

Category	Singular		Plural	
	Prefix	Meaning	Prefix	Meaning
1st person (masc. and femn.)	ብየ(*bəyä*)	I have	ብነ(*bənä*)	We have
2nd person masc.	ብከ(*bəkä*)	You have	ብክሙ(*bəkəmu*)	You have
2nd person femn.	ብኪ(*bəki*)	You have	ብክን(*bəkənə*)	You have
3rd person masc.	ቦ(*bo*) or ቦቱ(*botu*)	He has	ቦሙ(*bomu*)	They have
3rd person femn.	ባ(*ba*)or ባቲ(*bati*)	She has	ቦን(*bonə*) or ቦቶን(*botonə*)	They have

2.2.3 Geez Grammar

The grammatical structure of Geez reflects its origins as a Semitic language, as well as the unique cultural and religious context in which it evolved as the liturgical language of the Ethiopian orthodox church. A strong grasp of Geez grammar is crucial for scholars engaged in the study of ancient Ethiopian texts and manuscripts.

Morphology of Geez Language

The morphology of the Geez language covers the structure and formation of words, including how prefixes, suffixes, and root words are combined. It is quite complex and reflects its Semitic language origins. The morphology is based on a system of root consonants and affixes, this means words are formed by inserting vowels between the root consonants, as well as adding prefixes and suffixes. These root consonants are combined with various vowel patterns to form different word stems and derivations [20]. For example, the root ክ-ት-ብ (*k-t-b*) can form words like ከተበ (*kätäbä* 'he wrote') and ክታቢ (*kätabi* 'writer'). Verbs have a rich system of conjugations to indicate person, number, gender, tense, aspect, and mood.

Sentence Structure

The sentence structure of Geez language typically follows the patterns Verb- Subject-Object (VSO) as in the case of ገበረ እግዚአብሔር ሰማየ ወምድረ (*gäbärä ʔəgəziabəherə sämayä wämədərä* 'God created heaven and earth), Subject-Verb-Object (SVO) for the case of እግዚአብሔር ገበረ ሰማየ ወምድረ (*ʔəgəziabəherə gäbärä sämayä wämədərä)*, and Object-Verb-Subject (OVS) ሰማየ ወምድረ ገበረ እግዚአብሔር (*sämayä wämədərä ʔəgəziabəherə gäbärä)*. The nouns, adjectives and articles must agree in gender and number [20].

Simple sentences in Geez consist of a verb, a subject, and potentially objects. For example, አብረሃም ከተበ መጽሓፈ (*ʔabərähamə kätäbä mätsəḥafä* 'Abreham wrote a book').

Compound sentence in Geez language is formed by joining two or more independent clauses using coordinating conjunctions. Some sentences use the coordinating conjunction ወ (*wä* 'and') to link the clauses. Others use the coordinating conjunction ኣላ (*ʔala* 'but') to indicate a contrast between the clauses. Sentences that use the coordinating conjunction ኣው (*ʔawə* 'or') to present two alternative options are also categorized as compound sentences. The coordinating conjunction ዘ (*zä* 'for, because') express a cause-and-effect relationship. Complex sentences in the Geez are formed by incorporating one or more subordinate clauses into a main or independent clause.

Examples are:

- አይድዓኒ ሠናየ ነገረ ወሖረ (*ʔayədəʔani ṣänayä nägärä wäḥorä* 'he told me good news and went')
- መጽሓፈ ዘከተበ አብረሃም ተሳየጥኩ (*mätsəḥafä zäkätäbä ʔabərähamə täsayäṭəku* 'I bought the book that Abraham wrote)

- አይኑ ይሄይሰከ ዝ አው ዝንቱ (*ʔayənu yəheyəsə zə ʔaw zənətu* ‘which one do you prefer? this or that’)
- አልቦ ዘይበውእ እምአፍኣ ውስተ አፉሁ ለሰብእ ዘይክል አርኵሶቶ ዘእንበለ ዳዕሞ ዘይወፅእ እምልቡ ለሰብእ ውእቱ ያረኵሶ ለሰብእ (*ʔaləbo zäyəbäwəʔə ʔəməʔafəʔa wusətä ʔafuhu läsäbəʔə zäyəkələ ʔarəkwisoto zäʔənəbälä daʔəmu zäyəwätsəʔə ʔəməläbu läsäbəʔə wuʔətu yaräkwio läsäbəʔə* ‘There is nothing from without a man, that entering into him can defile him, but the things which come out of him, those are they that defile the man’)

2.3 Tigrigna Language

Tigrigna is a Semitic language spoken in Eritrea and in the Tigray Region of Northern Ethiopia. It is a member of the Ethiopic branch of Semitic languages with about 6 million speakers mainly in Tigray region of Ethiopia and in Central Eritrea [24]. There are also large immigrant communities of Tigrinya speakers in Sudan, Saudi Arabia, the USA, Germany, Italy the UK, Canada, and Sweden, as well as in other countries. Like Geez, Tigrigna is written from left to right. Although the Tigrigna script might look difficult, pronunciation is simple and straightforward, as the phonetic symbols closely resemble pronunciation [25, 24]. Tigrigna is one of the few African languages that still use an indigenous writing system for education and daily communication. The writing system, known as the Geez script, is adopted from Geez language. The Geez script is an abugida system in which each letter (alphabet) represent a consonant-vowel syllable. Gemination is not explicit in the Geez script; however, this limitation does not seem to pose a problem for native speakers [26]. On top of the 26 characters of Geez, Tigrigna needed additional characters to represent sounds that it acquired from Cushitic languages. This was done by placing a small bar on top of seven characters that were inherited from Geez. Tigrigna has modified this by adding vowels alphabets over the language linguistic evolution. It used the whole Geez alphabets and nine additional. The nine alphabets are formed by modifying nine Geez alphabets: explicitly, ሰ(*sä*) to ሸ(*shä*), ቀ(*qä*) to ቐ(q̄ä), ተ(*tä*) to ቸ(*chä*), ነ(*nä*) to ኘ(*gnä*), ከ(*kä*) to ኸ(*ḵä*), ዘ(*zä*) to ዠ(*ẓä*), ደ(*dä*)to ጀ(*jä),* ጠ(*ṭä*) to ጨ(*ṭ̌ä*) and በ(*bä*) to ቨ(*vä*). When these alphabets are added to the pure Geez alphabets it will be 34*7 size syllables. But the alphabets ሠ(*ṣä*), ጸ(*tsä*) and ኀ(*ḫä*) are mostly not used in Tigrigna. In addition to these, there are 5 by 5 labialized letters as ቈ (*qwä*), ቍ(*qwi*), ቋ(*qwa*), ቌ(*qwe*), ቍ(*qwə*), ቘ (*q̄wä*), ቝ(*q̄wi*), ቜ(*q̄wa*), ቚ(q̄we), ቝ(*q̄wə*), ኰ(*kwä*), ኲ(*kwi*), ኳ(*kwa*), ኴ(*kwe*), ኵ(*kwə*),ኸ(*ḵwä*), ኺ(*ḵwi*), ኻ(*ḵwa*), ኼ(*ḵwe*), ኽ(*ḵwə*), ጐ(*gwä*), ጒ (*gwi*), ጓ(*gwa*), ጔ(*gwe*), ጕ(gwə).

Like Geez language, each of the columns in Tigrigna alphabet are labelled as first order (*gəzə*), second order (*kaʔəbə*), third order (*saləsə*), fourth order (*rabəʔə*), fifth order (*ḥaməsə*), sixth order (*sadəsə),* and seventh order (*sabəʔə*) of alphabets. The orders represent the tones of each of the vowels. This shows us the combination of consonants and vowels [1].

2.3.1 Word Class in Tigrigna Language

Nouns

Nouns are words that represent a person, place, thing, or idea. Most nouns in Tigrigna do not have special forms to distinguish masculine and feminine genders. Nouns are masculine or feminine by nature, some nouns are identified by custom and others according to the context [27]. Table 2.6 some examples of nouns at each group.

Table 2.6 Nature, Custom and Context-Based Classification of Nouns in Tigrigna Language

Nature	Custom	Context
ሰብኣይ (*säbəʔayə* ‘man’) (masc)	ፀሓይ (*tsäḥayə* ‘sun’) (femin)	ገዛ (*gäza* ‘house’)
ብዕራይ (b*əʾəraya* ‘ox’) (masc)	ሓዊ (*ḥawi* ‘fire’) (masc)	ቆልዓ (*qoləʔạ* ‘child’)
ሰበይቲ (*säbeyəti* ‘woman’) (femin)	ንፋስ (*nəfasə* ‘wind’) (masc)	ዝብኢ (*zəbəʔi* ‘hyena’)
ላም/ላሕሚ (*lamə/laḥəmi* ‘cow’) (femin)	ኩዕሶ (*kuʔəso* ‘ball’) (femin)	መዓልቲ (*mäʔạləti* ‘day’)

Plurals of noun

Many nouns form their plurals through the addition of a suffix. In the class of nouns whose plural can be formed by rule, those which add ታት (*tatə*) are the most common. For example, take the word እምባ (*ʔəməba* ‘mountain’). The plural form of it is እምባታት (*ʔəməbatatə* ‘mountains’), which is by adding ታት (*tatə*) at the singular form. There are some rules that should be followed to make the plurals of nouns.

- If nouns end in third order (*saləsə*), the last letter must be changed to sixth order (*sadəsə*) before adding *tatə*. For example, the plural form of ክፍሊ (*kəfəli* ‘room’) is ክፍልታት (*kəfələtatə*

‘rooms’). The third order ሊ(*li*) at the end of the singular word ክፍሊ (*kəfəli*) is changed to sixth order which is ል(*lə*). Then the suffix ታት (*tatə*) is added.

- If the noun ends in sixth order, the last letter in most nouns is changed to fourth order and adds only ት(*tə*). In the word ጥራዝ (*ṭərazə* ‘notebook’), the last letter ዝ(*zə*) is in sixth order and thus it is changed to ዛ(*za*), fourth order. Then the suffix ት(*tə*) is added. The plural form ጥራዛት(*ṭərazatə* ‘notebooks’) is then formed.
- Some plurals add the suffix ት(*tə*) but incorporate other changes as well. This class of words can be named as irregulars. If we consider the word ስም (*səmə* ‘name’), its plural form is ኣስማት(*ʔasəmatə* ‘names’), and the word ወዲ (*wädi* ‘boy’) is changed to its plural form as ኣወዳት (*ʔawädatə* ‘boys’).

Tigrigna exhibits internal or broken plurals, which are formed by altering the word itself rather than adding suffixes. Although there are some patterns that can be observed, the similarities between two plurals are often recognizable but difficult to generalize into a rule. Table 2.7 lists some examples of these broken plurals.

Some nouns in Tigrigna have two plural forms. The words መበለት(*mäbälätə* ‘widow’) and ጓል(*gwalə* ‘girl’), their plural forms መበላሉ (*mäbälalu* ‘widows’) or መበለታት (*mäbälätatə* ‘widows) and ኣዋልድ(*ʔawaləd*ə ‘girls’) or ኣጓላት(*agwalatə* ‘girls’) respectively.

Table 2.7 Internal or Broken Plurals in Tigrigna

Singular	Plural
ሰብኣይ(*säbəʔayə* ‘man’)	ሰብኡት(*säbəʔutə* ‘men’)
ላም(*lamə* ‘cow’)	ኡሓ(*ʔaḥa* ‘cows’)
ገመል(*gämälə* ‘camel’)	ኣግማል(*ʔagəmalə* ‘camels’)
ኣድጊ(*ʔadəgi* ‘donkey’)	ኣእዱግ (*ʔaʔədug* ‘donkeys’)

Adjectives

Tigrigna adjectives precede the nouns or pronouns which they modify and agrees with it in gender and number. In the phrase ፅቡቕ ወዲ (*tsəbuq̄ə wädi* ‘handsome boy’), the noun ወዲ (*wädi* ‘boy’) is modified by the adjective ፅቡቕ (*tsəbuq̄ə* ‘handsome’). Similar to this, ነዋሕ (*näwaḥə* ‘tall’) is

modifying ጓል(*gwal* ‘girl’) in ነዋሕ ጓል (*näwaḥə gwal* ‘a tall girl’) and in ፀለምቲ ኣባጊዕ (*tsäläməti abagiʔə* ‘black sheep’), ፀለምቲ (*tsäläməti* ‘blak’) modifies ኣባጊዕ (*abagiʔə* ‘sheep’).

Pronouns

Pronouns are words that substitute for nouns or noun phrases. Tigrinya has subject and object pronoun forms. The pronouns in Tigrigna are listed in Table 2.8. The pronoun system reflects exhaustive number, gender, and person agreement values. Both subject and object pronoun stems are marked with similar pronominal suffix forms that are distinct for the gender, number, and person combination they mark [28].

Table 2.8 List of Pronouns in Tigrigna

Person	Singular		Plural		Polite	
1st person	Masculine	Feminine	Masculine	Feminine	Masculine	Feminine
	ኣነ(*anä* ‘I’)	ኣነ (*anä* ‘I’)	ንሕና(n*əḥəna* ‘we’)	ንሕና(n*əḥəna* ‘we’)	-	-
2nd person	ንስኻ(*nəsəḵa* ‘you’)	ንስኺ(*nəsəḵi* ‘you’)	ንስኻትኩም (*nəsəḵatəkumə* ‘you’)	ንስኻትክን (*nəsəḵatəkənə* ‘you’)	ንስኹም (*nəsəḵumə* ‘you’)	ንስኽን(*nəsəḵənə* ‘you’)
3rd person	ንሱ(*nəsu* ‘he’)	ንሳ(*nəsa* ‘she’)	ንሳቶም(*nəsatomə* ‘they’)	ንሳተን(*nəsatänə* ‘they’)	ንሳቶም(*nəsatomə* ‘they’)	ንሳተን(*nəsatänə* ‘they’)

Demonstrative Pronouns

The demonstrative pronouns, እዚ(*ʔəzi* ‘this’), እቲ(*ʔəti* ‘that’) have masculine and feminine forms in both singular and plural. In spoken language, the polite forms of demonstratives and possessives are frequently used as plurals. The demonstrative pronouns in Tigrigna are depicted in Table 2.9.

Table 2.9 Demonstrative Pronouns in Tigrigna

Gender	Singular	Plural	Polite
Masculine	እዚ(*ʔəzi* ‘this’)	እዚኣቶም(*ʔəziʔatom* ‘these’)	እዚኣም(*ʔəziʔomə* ‘these’)

	እቲ(*ʔəti* ‘that’)	እቲኣቶም(*ʔətiʔatom* ‘those’)	እቲኣም(*ʔətiʔomə* ‘those’)
Feminine	እዚኣ(*ʔəziʔa* ‘this’)	እዚኣተን(*ʔəziʔatänə* ‘these’)	እዚኤን(*ʔəziʔenə* ‘these’)
	እቲኣ(*ʔətiʔa* ‘that’)	እቲኣተን(*ʔətiʔatänə* ‘those’)	እቲኤን(*ʔətiʔenə* ‘those’)

Verb

A verb in Tigrinya is a linguistic term that indicates an action, state, or event and functions as the main element of a sentence's predicate. In Tigrinya, verbs are usually positioned at the end of a sentence, while in Geez, they are commonly placed in the middle. Both languages use verbs with two types of endings, one related to the subject and the other related to the object. The verb ኢዩ(*iyu*) is to be used when you are defining a person or object. The closest English equivalent is the verb ‘to be’. This verb (like all Tigrigna verbs) is divided by person, number, and gender and is summarized in Table 2.10.

Table 2.10 List of verb-to-be in Tigrigna

Gender	Person	Number	
		Singular	Plural
Masculine	First	እየ(*ʔəyä* ‘I am’)	ኢና(*ʔina* ‘we are’)
	Second	ኢኻ(*ʔiḵä* ‘you are’)	ኢኹም(*ʔiḵumə* ‘you are’)
	Third	ኢዩ(*ʔiyu* ‘he is’)	ኢዮም(*ʔiyomə* ‘they are’)
Feminine	First	እየ(*ʔəyä* ‘I am’)	ኢና(*ʔina* ‘we are’)
	Second	ኢኺ(*ʔiḵi* ‘you are’)	ኢኺን(*ʔiḵinə* ‘you are’)
	Third	እያ(*ʔəya* ‘she is’)	ኢየን (*ʔiyänə* ‘they are’)

2.3.2 Tigrigna Grammar

Tigrigna Morphology

Like Geez language, the foundation of verbs and most nouns in Tigrigna is characterized by a sequence of consonants or radical roots. These roots are then used to construct actual words by adding vowels and non-root consonants around the root consonant. The particular morphological category determines which vowels and consonants are added. As it is the case for Geez, the parts of speech in Tigrigna exhibit a high degree of inflection [14]. For instance, the morphological variations of a verb are formed by affixing suffixes to the verb stem to indicate the person, gender, and number to the verb stem. To take an example, consider the root ሰ-ር-ሐ. From this words like ሰርሐ (*särəḥä* 'he worked'), ሰሪሑ (*säriḥu* 'he worked'), ሰሪሓ (*säriḥa* 'she worked'), ሰሪሐ (*säriḥä* 'I worked') ሰሪሖም(*säriḥomə* 'they worked/masc'), ሰሪሐን (*säriḥänə* 'they worked/femn'), ሰሪሕካ (*säriḥəka* 'you worked/masc'), ሰሪሕኪ (*säriḥəki* 'you worked/femn'), ሰሪሕኩም (*säriḥəkumə* 'you worked/plural') are formed.

Sentence Structure in Tigrigna Language

The basic word order in Tigrigna sentence is Subject-Object-Verb (SOV). To illustrate this with an example let us consider the sentence እቶም ጳጳስ ካብ ኣክሱም መጺኦም (*ʔətomə ṗaṗasə kabə ʔakəsumə mätsi'omə* 'the bishop came from Aksum'). Here the subject is ጳጳስ (*ṗaṗasə* 'bishop'). The object and the verb are ኣክሱም (*ʔakəsumə* 'Aksum') and መጺኦም (*mätsiʔomə* 'came') respectively. Sentences can be simple, compound or complex.

A simple sentence in Tigrigna contains a single independent clause, a subject and a verb. It can stand alone as a complete thought. For example, consider the sentence ኣቦይ መጽሓፍ ጽሒፉ (*ʔaboyə mätsəḥafə tsiḥifu* 'my father wrote a book').

A compound sentence in Tigrigna contains two or more independent clauses joined by a coordinating conjunction like ስለዚ (*səläzi* 'because'), ወይ (*wäyə* 'or'), ከምኡ ድማ (*kämәʔu dəma 'also*'), etc. Each independent clause could stand alone as a simple sentence. For example, ኣነ ተምሃራይ ኢየ፣ ስራሕ ድማ ኣለኒ (*ʔanä tämәharayə ʔiyä, səraḥə dəma ʔaläni* 'I am a student, and I also have a job'). This compound sentence can be divided into two simple sentences, ኣነ ተምሃራይ ኢየ (*ʔanä tämәharayə ʔiyä* 'I am a student') and ስራሕ ኣለኒ (*səraḥə ʔaläni* 'I have a job'). These sentences are joined by ድማ (*dəma 'also*') and form a single compound sentence.

A complex sentence in Tigrigna contains of an independent clause and one or more dependent clauses. The dependent clause cannot stand alone and relies on the independent clause to complete the thought. The conjunctions here consist of እንተ (*ʔənätä* ‘if’), መዓስ (m*äʔąsə* ‘when’), ምስ (*məsə* ‘with’), ከመይ (*kämäyə* ‘how’), እቲ (*ʔəti* ‘which’), etc. example of a complex sentence can be **እቲ ተምሃራይ ዝተመዝገበሉ ቤት ትምህርቲ ብዙሓት መምህራን ኣለውዎ** (*ʔəti täməharayə zətämäzəgäbälu betə təməhərəti bəzuḥatə mämähəranə ʔaläwuwo* ‘the school which the student enrolled has many teachers’). Here **እቲ ተምሃራይ ዝተመዝገበሉ ቤት ትምህርቲ** (*ʔəti täməharayə zətämäzəgäbälu betə təməhərəti)* cannot stand alone.

2.4 Overview of Machine Translation

Machine Translation (MT) is a classic sub-field in NLP that investigates how to use computer software to translate a text or a speech from one language to another without human involvement [29]. The idea of MT was raised several decades ago. As stated by Quinn Lanners [30] and Sonali Sharma et.al [31], Warren Weaver was the first to propose the idea of machine translation in 1949. Cheragui and Mohamed Amine [5] also stated that the starting period of MT is traced back to 1949. Machine translation is the task of converting given text from one language to another language using automatic algorithms. In contrast to human translations, once it is built, an MT system can be used repeatedly for many purposes without additional labour costs [32]. MT saves time, money, and effort.

According to [33, 34] the encoder-decoder transformer or sequence-to-sequence model is the commonly used architecture for machine translation. Its purpose is to generate a sentence in a target language based on a given sentence in a source language. Supervised machine learning is employed in this approach, where the system is trained using a large dataset of parallel sentences, with each source language sentence paired with its corresponding target language sentence. To facilitate the process, sentences are often broken down into sub-word tokens instead of individual words, allowing for more efficient translation. The systems are then trained to maximize the probability of the sequence of tokens in the target language $y_1, ..., y_m$ given the sequence of tokens in the source language $x_1,..., x_n$.

$$P(y_1, \dots, y_m | x_1, \dots, x_n) \quad (1)$$

Where P(y|x) is probability distribution of y given x.

Rather than using the input tokens directly, the encoder-decoder architecture consists of two components, an encoder, and a decoder. The encoder takes the input words $x = [x_1, ..., x_n]$ and produces an intermediate context h. At decoding time, the system takes h and, word by word, generates the output y.

$$h = \text{encoder}(x) \tag{2}$$

$$y_{i+1} = \text{decoder}(h, (y_1, \dots, y_i)) \tag{3}$$

Where h is context vector, x is the input to the encoder, and y is output of the decoder.

Encoder-decoder networks, also known as sequence-to-sequence networks, are models able to produce output sequences of any length that are contextually appropriate, given an input sequence. The general structure of the encoder-decoder architecture is displayed in Figure 2.1as it is discussed in[33]. These networks have been used in a variety of applications, such as summarization, question answering, and dialogue, but are especially popular for machine translation [35, 34]. The fundamental concept behind these networks is the use of an encoder network to create a contextualized representation of an input sequence, known as the context, which is then passed to a decoder to generate a specific output sequence for a given task.

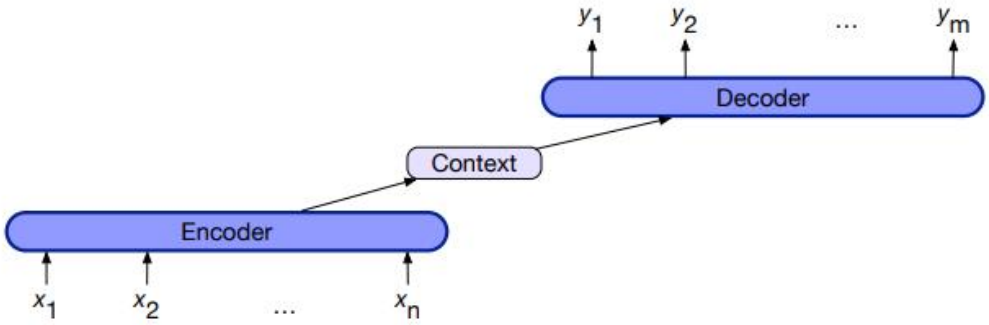


Figure 2.1 The General Structure of the Encoder-Decoder Architecture

The context is a function of the hidden representations of the input and is used by the decoder in a variety of ways. As depicted on [33, 36], encoder-decoder networks consist of three components:

i. An encoder that accepts an input sequence, x_1^n, and generates a corresponding sequence of contextualized representations, h_1^n. Recurrent Neural Networks (RNNs), Long Short-Term Memories (LSTMs), convolutional networks, and transformers can all be employed as encoders.

ii. A context vector, c, which is a function of h_1^n, and conveys the essence of the input to the decoder.

iii. A decoder, which accepts c as input and generates an arbitrary length sequence of hidden states h_1^m, from which a corresponding sequence of output states y_1^m, can be obtained. Just as with encoders, decoders can be realized by any kind of sequence architecture.

So, in any language model, we can break down the probability as follows:

$$\boldsymbol{P(y) = P(y_1)P(y_2|y_1)P(y_3|y_1, y_2\ ) \ldots P(y_m|y_1, \ldots, y_{m-1})} \quad (4)$$

Where P(y) is the probability of the sequence of tokens in the target language.

2.5 Attention Mechanism

Attention is a mechanism like human cognition that allows the machine to identify which specific words in a sentence or sentences are most important in relation to other words. According to [37], the primary concept of attention is to prioritize important aspects of the input source text during the translation learning process. This mechanism establishes efficient connections between the input source text and target text using learned weights, ultimately improving the ability of the decoder to accurately translate longer phrases. Bahdanau et al [38] explain the attention mechanism as the way of providing a direct short cut between target and source sentences by scoring each word in the source sentence on its importance for predicting the current token. The perception behind attention is to mimic human translation. Through human translation the source sentence is constantly evaluated to assess whether the current translation accurately maps the source. The attention mechanism attempts to solve the problem of having thought vector of a fixed length as the exclusive representation of the input sequence. Without the attention mechanism, long translations becoming increasingly poor as the entire sequence is condensed in a single thought vector, which is updated after every encoding step. The attention mechanism attends the source encoding in every decoding step, which brings information from the source side and improves the translation quality especially for long sentences [39].

2.6 Machine Translation Approaches

MT is a complex field with a variety of approaches that have been adopted and explored by researchers across the research community. These approaches can broadly be classified into two

major categories: rule-based approaches and corpus-based approaches, based on the way they acquire and utilize knowledge for the translation process as it is detailed on Figure 2.2. In the rule-based approach to machine translation, human language experts and linguists specify a comprehensive set of formal rules and guidelines that describe the various linguistic transformations and mapping required to translate text from one language to another. This rule-based system relies heavily on the extensive involvement and input from human experts to carefully organize the complexities of natural language and the translation process. The primary advantage of this approach is the ability to incorporate deep linguistic knowledge, but it requires a substantial investment of time and effort from human experts to construct the rule systems [40]. Conversely, the corpus-based approach to machine translation takes a more data-driven approach. In this paradigm, the translation knowledge is automatically extracted by analysing a large parallel corpus of human-translated text examples. By identifying patterns, correspondences, and statistical relationships in these real-world translation samples, corpus-based MT systems can learn to perform translations without the need for extensive rule authoring by human experts. This approach allows the system to leverage empirical data and statistical models to handle the inherent complexities of natural languages. Building on the strengths of these two major categories, a third type of MT approach has emerged, the hybrid machine translation approach [41]. Hybrid systems combine elements of both rule-based and corpus-based approaches, aiming to associate the complementary benefits of each.

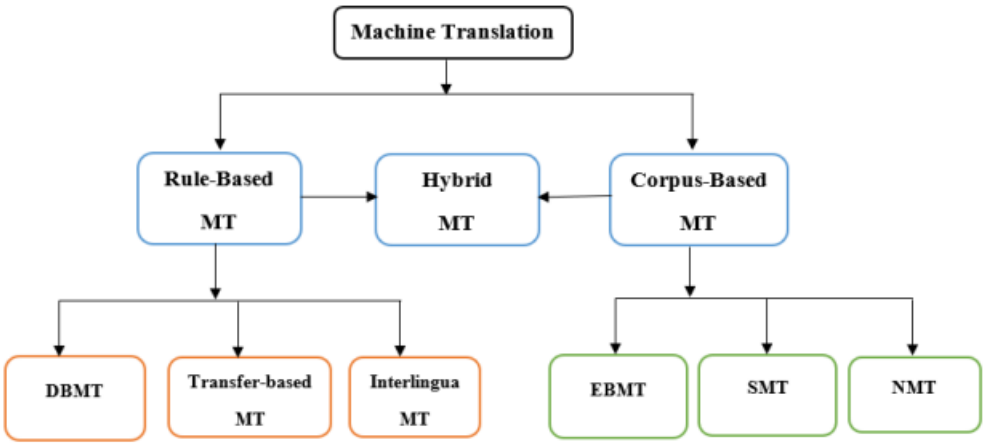


Figure 2.2 Classification of MT Approaches Based on Knowledge Acquisition

Corpus-Based Machine Translation (CBMT) refers to a situation in which the software depends on a stored data corpus to translate from one language into another language. Corpus refers to a huge collection of data and texts of both the source and the target languages. The parallel collected corpus is employed for the purpose of the translation. The idea behind corpus-based form of translation is that it is based on bilingual translation tables. The rules do not have to be defined manually, as in the case of rule-based system, and the preparation of the machine translation is therefore less time-consuming. Since corpus-based approaches do not require deep linguistic analysis of the source and target languages, it is the preferred approach for under-resourced languages of the world, including Ethiopian languages [42].

As it is illustrated on [43], CBMT has delivered progressively improved quality translations since its foundation. Nowadays, it is the most widely used area in machine translation as it has a high level of accuracy when compared to the other approaches [44, 7]. It requires a large amount of bilingual content in the source language and the target language. These parallel data are used by the machine translation system for acquiring translation knowledge. The corpus-based machine translation is further classified into statistical machine translation, example-based machine translation and neural machine translation [41, 45].

2.6.1 Rule Based Machine Translation

Rule-Based Machine Translation (RBMT) is the first approach developed in the field of machine translation and it is based on linguistic information, based on linguistic rules that are prepared and entered by linguists [29, 31]. It is also known as a knowledge-driven approach. The translation system consists of a collection of grammar rules, a lexicon and software programs to process the rules. It produces more predictable output for grammar since it deals with syntactic, semantic, and morphological analysis in both source language and target language [40]. As all the rules of the language need to be applied and there is a need of huge linguistic knowledge, building rule-based machine translation is expensive (very time-consuming and labour-intensive). But once it is built it can be deeply analysed at syntax and semantic level. There are three different types of rule-based machine translations. These are direct translation, transfer based translation and Interlingua translation [41, 45].

i. Direct Translation: as its name indicates it is a word-by-word translation approach. It directly translates the source language to the target language. It is unidirectional bilingual machine

translation. It requires a huge amount of morphological analysis but only a little syntax and semantic analysis is required [44].

ii. Transfer-based Translation: this involves three stages which includes analysis, transfer, and generation. In the analysis stage, the source language is analyzed and converted into syntactic representation of source language. In transfer stage, it transfers the syntactic representation of source language to a syntactic representation of target language. In the generation stage, the target language is generated using morphological analyzer [5].

iii. Interlingua Translation: is a combination of two Latin words Inter and Lingua which means intermediary and Language respectively. The source language is transformed into intermediate language then the intermediate language is transformed into target language [5].

2.6.2 Statistical Machine Translation

Statistical Machine Translation (SMT) method treats the translation as a mathematical reasoning problem. It uses the statistical model made by analysis of bilingual corpus. The statistical machine translation involves three key models: a language model, a translation model, and a decoder model. The language model generates potential translations for each word or phrase in the input sentence, assigning a probability to each translation. The translation model calculates the conditional probability of target sentences given the source sentence. The decoder model seeks the best possible translation by maximizing the product of the probabilities from the language and translation models [46].

SMT integrates a translation model with a target language model to convert sentences from a source language to a target language. The translation model maps words and phrases from the source language to the target language, while the language model captures statistics on the likelihood of specific word sequences in the target language. SMT aims to maximize the probability of selecting a target sentence that accurately translates the source sentence. These statistical models are developed from a comprehensive corpus of source-to-target language translations. Before the introduction of SMT, machine translation relied on experts defining linguistic and syntactic rules to facilitate the translation process. SMT was a step forward in automatic machine translation where such rules are statistically learned from a large collection of bilingual data. One of the major complications with SMT is that it can only perform well when translating text like the training corpus or domain. For new input text from a different domain,

SMT may not translate well. Another disadvantage is that we need a large collection of bilingual training data, which may be difficult for rare language pairs [47].

2.6.3 Example-Based Machine Translation

The Example-Based Machine Translation (EBMT) system utilizes corpora to identify similar examples between the source and target languages. This approach, also known as memory-based translation, employs a direct mapping with a similarity measure such as word, syntactic, or semantic similarity to identify sentences that closely match each other. The translation system is categorized into two modules: retrieval module and adaption module. For a given input sentence, the retrieval module retrieves the similar sentences and its translation from the corpus. From the retrieval module the adaption module finds out the part of translation that can be reused. If the input sentence and the retrieval sentence match, then the correct translated output is given. If it does not match exactly, the relevant match correspond to the source language is used instead [41]. As it is explained in [12], An EBMT system takes a series of sentences in the source language and corresponding translations of each sentence into the target language with point-to-point mapping. These examples are used to translate similar types of sentences in the source language into the target language. The basic idea of translation used in example-based machine translation is a translation by analogy. The principle of analogy translation is encoded in the machine translation based on examples through the example translations used to train such a system.

2.6.4 Neural Machine Translation

Neural machine translation is the art of using Artificial Neural Network (ANN) models to learn a statistical model for machine translation. It is a deep neural network model that happened to be the state-of-the-art machine translation nowadays [29]. The main benefit to using NMT is that the system is trained on the source and target text directly without having to leverage any specialized systems which SMT systems used. As stated in [38], NMT builds and trains a single, large neural network that reads a sentence as input and produces a correct translation as output in an end-to-end fashion by mapping from input sentence to its corresponding output sentence. Separate models such as the language model, translation model, and reordering model are not needed, it is just a large single sequence model (neural network) that predicts one word at a time. This end-to-end structure is demonstrated on Figure 2.3. It explains that the architecture of NMT models often consists of an encoder and a decoder. First, each word in the input sentence is fed separately into

the encoder to encode the source sentence into an internal fixed-length representation called the context vector. This context vector contains the meaning of the sentence. then, the decoder decodes the fixed-length context vector and then predicts the output sequence.

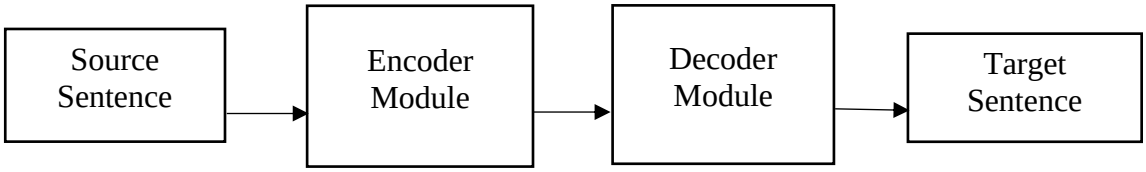


Figure 2.3 End-to-End Structure of Neural Machine Translation

The encoder of the NMT infers a continuous space representation of the source sentence, while the decoder is a neural language model conditioned on the encoder output. The parameters of both models are learned jointly to maximize the likelihood of the target sentences given the corresponding source sentences from a parallel corpus [48]. NMT uses a neural network to learn to translate text from a source language into a target language. Unlike SMT, one key advantage of NMT is that it needs only one model to translate end-to-end from a source language into a target language. More importantly, NMT works on whole segments of the source text rather than chunks or phrases as in SMT. This is achieved by learning context through word embeddings. Therefore, NMT performs translation while preserving the context of the original text as well [47].

Nowadays, NMT is the primary algorithm used in industry to perform machine translation [34, 7]. The architecture behind neural machine translation is composed of two recurrent neural networks used together in tandem to create an encoder-decoder structure. Attention mechanisms have been developed to further increase the accuracy of these models. Before NMT, SMT provided the most state-of-the-art results [30]. The field of neural machine translation has seen the development of several distinct architectural approaches, each with its own unique characteristics and capabilities. The three primary types of NMT systems that have emerged are Convolutional Neural Network (CNN) based NMT, Recurrent Neural Network (RNN) based NMT, and Transformer-based NMT.

Convolutional Neural Network

Convolutional neural networks are a prevailing class of deep learning models extensively applied in numerous tasks, incorporating object detection, speech recognition, computer vision, image classification, and bioinformatics [49, 50]. CNNs are feedforward neural networks that influence convolutional structures to extract features from data. Unlike traditional methods, CNNs

automatically learn and recognize features from the data without the need for manual feature extraction by humans [51].

Recurrent Neural Network

A recurrent neural network is a network that has a loop in its connections, allowing the output of a unit to be used as an input later, either directly or indirectly [33]. These networks are a class of deep learning models that possess internal memory, enabling them to capture sequential dependencies. Traditional neural networks treat inputs as independent entities, while RNNs consider the temporal order of inputs, making them suitable for tasks involving sequential information. RNNs use a loop to apply the same operation to each element in a series, with the current computation depending on both the current input and the previous computations. This ability to utilize contextual information is valuable in tasks such as natural language processing, video classification, and speech recognition. For example, in language modelling, understanding the preceding words in a sentence is crucial for predicting the next word, and RNNs excel at capturing such dependencies due to their recurrent nature. However, a limitation of simple RNNs is their short-term memory, which restricts their ability to retain information over long sequences. To overcome this, more advanced RNN variants have been developed, including Long ShortTerm Memory (LSTM), bidirectional LSTM, Gated Recurrent Unit (GRU), bidirectional GRU, Bayesian RNN, and others [49].

In RNN language modelling, at a particular time t, we pass the prefix of t−1 tokens through the language model, using forward inference to produce a sequence of hidden states, ending with the hidden state corresponding to the last word of the prefix. We then use the final hidden state of the prefix as our starting point to generate the next token. More formally, if g is an activation function like tanh or ReLU, a function of the input at time t and the hidden state at time t−1, and f is a softmax over the set of possible vocabulary items, then at time t the output y_t and hidden state h_t are computed as [33]:

$$h_t = g(h_{t-1}, x_t) \tag{5}$$

$$y_t = f(h_t) \tag{6}$$

Where h_t is a hidden state at time t, g is activation function y_t is the predicted output.

Transformer Neural Network

The transformer was initially proposed by Vaswani et el. [34, 52] and defined as a model architecture avoiding recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output. The standard transformer architecture developed by Vaswani et al. is depicted in Figure 2.4. These deep neural networks substitute CNNs and RNNs with self-attention. Self-attention allows transformers to easily transmit information across the input sequences. The transformer allows for significantly more parallelization and can reach a new state-of-the art in translation quality [53]. As it is shown in [54], preceding to the transformer, the most successful language translation systems had relied on RNN, including especially LSTM and GRU that process input sequences in order. Recurrent networks are named for their ability to process input information in a sequential, looping manner, allowing them to capture temporal dependencies. This is in contrast to traditional feedforward neural networks, which process inputs in a parallel fashion. RNNs process sequences step by step, which hinders parallelization and makes it challenging to learn long-term dependencies within input and output sequences. Hence, the transformer model replaces RNNs with self-attention layers to get rid of them.

Self-attention is an attention mechanism linking different positions of a single sequence to find out a representation of the sequence. Transformer relies entirely on self-attention to compute representations of its input and output without using sequence aligned RNNs or convolution. It proposes to encode each position, apply self-attention in both decoder and encoder, enhance the idea of self-attention by calculating multi-head attention [55]. The self-attention of TNNs completely replaces the RNNs in both the encoder and decoder and allows the model to use the whole input sequence in the encoder at once [56]. This alleviates one of the larger issues accompanying with RNNs, their inability to be used in parallel. The entire encoder can be run in parallel and all at once, which allow for large speedups of the network in both training and predicting. In TNNs, the decoder is still sequential and thus must be run one element at a time. Models using TNNs, such as Bidirectional Encoder Representations from Transformers (BERT) [57, 58], are extremely robust and have accuracies rivalling other sequential models. As illustrated on [35, 59], TNNs have a very similar flow to the general encoder-decoder, but with a few key differences. First, the input sequences are given a numerical representation and positional

information is added to the input sequence to generate embedded inputs. This input is then passed into a self-attention layer that learns to pay attention to certain aspects of the input sequence.

The attention the encoder uses is also said to be multi-headed attention, as the encoder uses multiple attention blocks. This is to allow each block to pay attention to different aspects of the input sequence and, ideally, preserve different levels of meaning at each. These are then passed into a feed-forward network that combines all the information learned in each attention head. After both the multi-head attention block and the feed-forward network, a residual connection is added to improve performance of the model. The output from each section (i.e. multi-headed attention and dense network) is combined with the input, and the result is normalized. This allows the encoder to be more robust and allows it to learn better at each step. The encoded output sequences generated, containing a numerical representation for the meaning of the input sequence. This is then passed to the decoder. The decoder first passes all current and previously generated outputs to itself and applies multi-headed masked attention to them. The input is then fed into another multi-headed Attention block, but this time it is combined with the encoded output from the encoder block. The output of this layer is fed to a feed-forward network, like in the encoder block. There are again residual connections around both multi-headed attention blocks and the feed-forward network that are combined with the original input and normalized [60]. Finally, the decoder output is passed to a linear layer with a softmax activation function to get a probability distribution as the output. This is then passed back to the beginning of the decoder as the input of the next layer. TNNs have quickly become the most important neural network for sequence-to-sequence tasks.

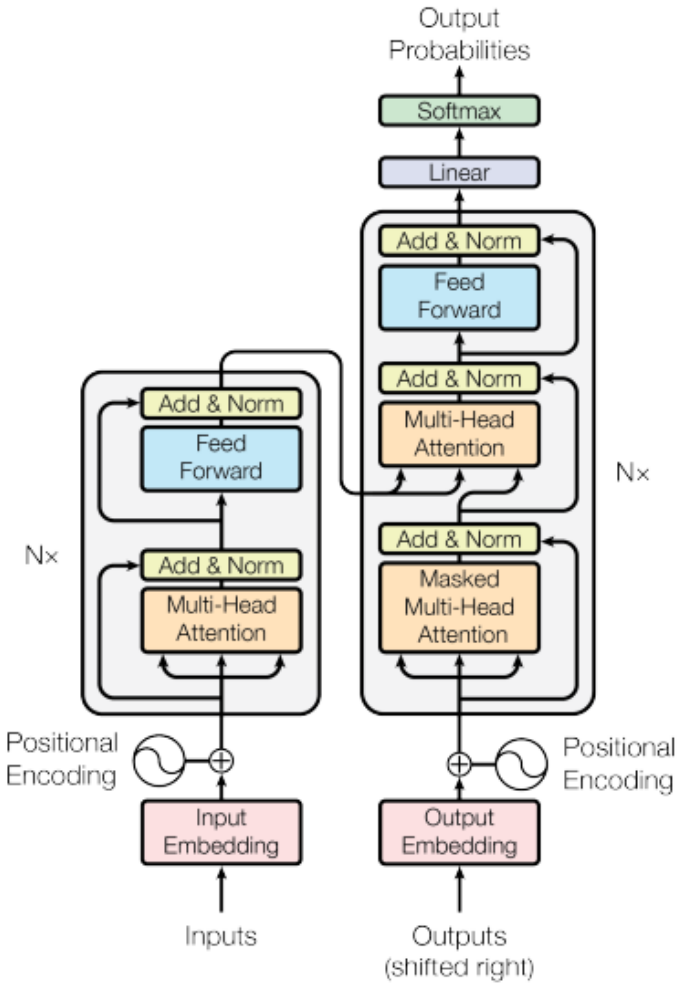


Figure 2.4 The First Transformer Model Architecture Developed by Vaswani et al.

Soydaner [61] explains transformer as an architecture composed of encoder-decoder stacks each of which has six identical layers within itself. The most impressive point about this architecture is that it contains neither recurrence nor convolution. The transformer performs well by replacing the conventional recurrent layers in encoder-decoder architecture used for NMT with self-attention. Each stack includes only attention mechanisms and feedforward neural networks. As this architecture does not include any recurrent or convolutional layer, information about the relative or absolute positions in the input sequence is given at the beginning of both encoder and decoder using positional encodings. The calculation of self-attention uses three vectors namely query, key, and value for each word [34, 35]. These vectors are computed by multiplying the input with weight matrices W_q, W_k and W_v which are learned during training. In general, each value is weighted by

a function of the query with the corresponding key. The output is computed as a weighted sum of the values. Based on this idea, two attention mechanisms are proposed: In the first one, called scaled dot-product attention, the dot products of the query with all keys are computed. Each result is divided to the square root of the dimension of the keys to have more stable gradients. They pass into the softmax function, thus the weights for the values are obtained. Finally, each softmax score is multiplied with the value. Then the attention on a set of queries simultaneously is computed by taking queries and keys of dimension d_k, and values of dimension d_v as inputs. The keys, queries and values are packed together into matrices K, Q and V. Finally, the output matrix is obtained by the formula:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{7}$$

Where Q, K and V are the query vectors, key vectors, and value vectors respectively, and d_k is the dimension of query and key vectors.

This calculation is accomplished by every word against the other words. This leads to having values of each word relative to each other. For instance, if the word x_2 is not relevant for the word x_1, then the softmax score gives low probability scores. As a result, the corresponding value is decreased. This leads to an increase in the value of relevant words, and those of others decrease. In the end, every word obtains a new value for itself. The transformer model does not directly use scaled dot-product attention. But the attention mechanism it uses is based on these calculations. The second mechanism proposed, called the multi-head attention, linearly projects the queries, keys, and values h times with different, learned linear projections to d_q, d_k and d_v dimensions, respectively. The attention function is performed in parallel on each of these projected versions of queries, keys, and values, i.e., heads. By this way, d_v-dimensional output values are obtained. To get the final values, they are concatenated and projected one last time. By this way, the self-attention is calculated multiple times using different sets of query, key, and value vectors [34]. Thus, the model can jointly attend to information at different positions.

$$\text{MultiHead}(Q,K,V) = \text{Concat}[\text{head}_1, \dots, \text{head}_h]W_0 \tag{8}$$

Where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ and all the Ws are learnable parameter matrices.

In the decoder part of the transformer, masked multi-head attention is applied first to ensure that only previous word embeddings are used when trying to predict the next word in the sentence. Therefore, the embeddings that shouldn't be seen by the decoder are masked by multiplying with zero [61].

2.6.5 Hybrid Machine Translation

As the name suggests, Hybrid Machine Translation (HMT) is a method of machine translation that incorporates the use of multiple different machine translation approaches within a single machine translation system. The underlying motivation behind the use of HMT is the fact that a failure of a single machine translation technique should not stop the system from achieving the required level of accuracy.

The hybrid approach uses a combination of rule-based and statistical approaches. It is with the advantages of the statistical-based approach and rule-based approach that a hybrid approach was established. The hybrid-based translation can be done in different ways. In few cases translations are performed first using rule-based approach, later the output is produced by correcting the output using the statistical approach and in few cases the rules are used to preprocess the input data as well used to modify the output. In the hybrid approach of the machine translation the drawbacks of both the approaches were eradicated to provide the promising translation with high efficiency [46]. Rule based approach has a high accuracy as it deeply analyses syntax and semantic level, but it is very expensive as it necessitates a huge linguistic rule. On the other hand, the statistical based approach is less expensive as it uses the mathematical reasoning problem but needs huge corpora which are not available for low resourced languages [41].

Figure 2.5, adopted from [31] , summarizes the evolution of MT over the years. From the beginning till 1980s was the period of RBMT. After that EBMT emerged in 1980-1990. SMT became the hot research idea during the period from 1990 to 2014. And now the NMT has dominated the machine translation community.

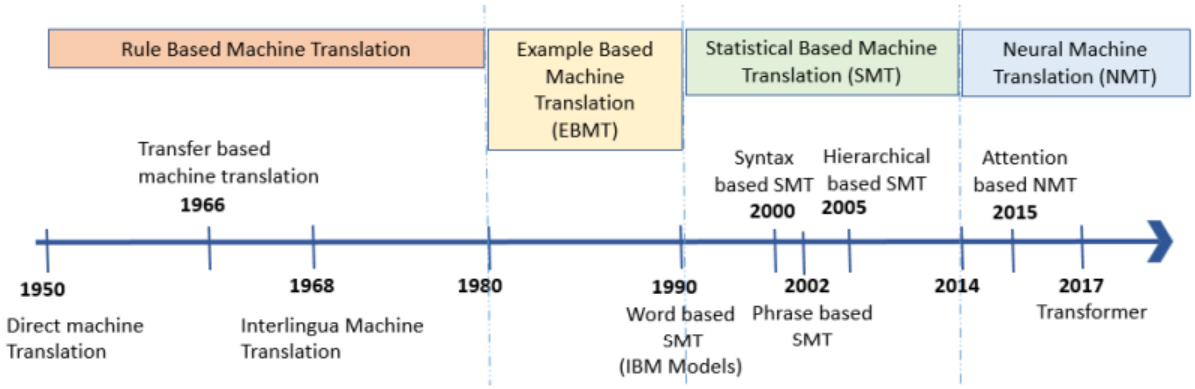


Figure 2.5 Evolution of Machine Translation Over the Years

2.7 Evaluation of Machine Translation

Machine translation systems undergo comprehensive evaluations using various techniques to assess their quality. These techniques serve as valuable tools for evaluating the efficiency of machine translation output. The evaluation process involves a thorough examination of the performance and quality of machine translation systems, accomplished by comparing their output to reference translations that have been particularly constructed by human translators. By conducting such comparative analyses, the primary goal of evaluation is to measure the accuracy, fluency, and overall quality of machine translation systems [62].

Some of the commonly utilized machine translation evaluation techniques include automated metrics, such as BLEU, which quantitatively measures the similarity between machine translations and reference translations. Other metrics, like Translation Edit Rate (TER), focus on quantifying the number of edits required to transform a machine translation into a reference translation. Additionally, human evaluation methods are employed, where expert human evaluators assess the quality of machine translations based on predefined criteria, such as adequacy, fluency, and overall preference. These techniques collectively provide a comprehensive and multifaceted evaluation of machine translation systems, enabling researchers, developers, and users to make informed assessments about their performance, identify areas for improvement, and facilitate advancements in the field of machine translation [63].

BLEU Score: BLEU is the prevailing standard for evaluating the quality of machine translation automatically. It is a measurement that compares the output of a machine translation system with one or more reference translations. The BLEU score of an output of a system is determined by

counting the occurrences of word sequences, known as n-grams, in the output that also appear in the set of reference translations. BLEU is a precision-focused metric, that quantifies the accuracy of the output rather than assessing whether the references are entirely replicated in the output. The adoption of BLEU as a metric has stimulated progress in the field of machine translation research, and it now serves as the benchmark against which all new metrics are evaluated. The BLEU score ranges from 0 to 1, with a higher score indicating a greater degree of overlap between the machine-translated text and the reference translation. As the score approaches 1, the machine translation becomes more like a human translation [62, 64].

Translation Edit Rate: TER is an automated metric commonly used for evaluating the quality of machine translation output. It measures the number of edits required to transform the machine translation into the reference translation. A lower TER score indicates a higher similarity between the machine translation and the reference translation, suggesting better quality. However, it is important to note that TER does not capture the semantic or contextual accuracy of translations. It focuses primarily on the structural changes required to align the machine translation with the reference translation.

Human Evaluation: Human evaluation is a crucial aspect of machine translation evaluation that involves expert human evaluators assessing the quality of machine translations. It is considered an essential complement to automated metrics as it provides a more comprehensive assessment of translation quality, considering factors that automated metrics may not capture accurately. In human evaluation, trained evaluators, often professional translators, or linguists, assess the machine-translated output in comparison to the reference translation or the original source text. They provide subjective judgments based on predefined criteria [65]. While human evaluation is resource-intensive and subject to individual biases, it remains an indispensable component of machine translation evaluation, ensuring that the translations meet the desired linguistic quality standards and are suitable for their intended purposes. It helps bridge the gap between automated metrics and human perception, providing a more general and accurate assessment of machine translation systems.

Chapter 3: Related Work

3.1 Introduction

In this chapter earlier works on various MT approaches that help to grasp the general and specific targets of the thesis work are revised. This task mainly concentrates on systematically going through thesis papers, PhD dissertations, and online publications. Due to the exchange of information between various regions using different regional languages, the demand for translation has been increasing and plenty of studies have been done for such objectives. Several translation systems among different languages are also developed and some of those developed machine translation systems are discussed. The way this chapter revises the previously developed systems is based on the types of the translation methods. Papers based on rule-based approach are listed and discussed first followed by the corpus-based architectures. And lastly, related works based on a hybrid approach are presented.

3.2 Machine Translations Using Rule-based Approach

Agbeyangi et.al [66] worked on the English to Yoruba MT using rule-based approach. Since transfer rule-based machine translation allows to use manual tagging of the part of speech, it was used in the development of the system. Rewrite rules were developed for the two languages being studied. The data was collected from home domain vocabularies. The re-write rule was verified using Natural Language Toolkits (NLTKs) and implemented using python programming language. The system interface gives the user an opportunity to type simple English language sentence, and the resulting Yoruba translation was displayed. The result shows that the system performance is close to the expert opinion, having considered the scope for which the system is developed. The software was designed as a window application. The grammar of the language is written to follow the re-writes rule developed for the translation process. The corpus for the research was from simple sentence spoken in the home environment. The sentences were further broken down into their part of speech and stored in pairs. The system performance was compared with Google translator since Google translator is a standard machine translator that include an English to Yoruba translation. The result from the output from Google translator shows that the proposed system states a better translation and can be considered as a standard translation for Yoruba languages. In this specific study it was indicated that rule-based approach was a good approach for

MT system used for languages with lots of grammar which Yoruba language is one. The popularity of Yoruba language among the three main languages in Nigeria called for the need to computerise the language.

Goyal and Singhlehal [67] developed Hindi to Punjabi machine translation system. Both languages are closely related, and an ideal approach for this translation process was direct approach. The paper presented the evaluation results of Hindi to Punjabi machine translation system. Every Machine translation undergoes an evaluation process for testing its accuracy to know its success. The survey was done by 50 people of different professions. 20 persons were from Villages who only knows Punjabi language and do not know Hindi and 30 persons were from different professions having knowledge of both Hindi and Punjabi languages. Average ratings for the sentences of the individual translations were then summed according to intelligibility and accuracy to get the average scores. Percentage of accurate sentences and intelligent sentences was also calculated separately by counting down the number of sentences. From the analysis, it was concluded that the overall accuracy of Hindi to Punjabi machine translation system was found to be 95.12%. Robust pre-preprocessing and post-processing of the system were listed as a performance improvement measures. This system was comparable with other existing systems and its accuracy was better.

3.3 Machine Translations Using Statistical Approach

Habtamu Mekonen [68] worked on Amharic-Awngi machine translation using statistical approach. The parallel corpus for this study was collected from Amharic texts, mass media agency and the Bible. The author used a minimum of 1500 simple sentences, 1000 compound and 1000 complex sentences and maximum of 5000 sentences for each sentence type to train the system. 90% by 10% splitting ratio was used to divide the corpus into train and test set respectively. A minimum of 5700 and maximum of 14491 monolingual sentences of Awngi language were used for language modelling. For the implementation of the system, tools like Moses for Mere Mortal for translation process, MGIZA++ for alignment and IRSTLM for language model were used. Experimental results showed that a performance of 37% BLUE score was registered using complex sentences. In Amharic language, a word in sentences can have more than one meaning as Amharic is morphologically rich language. While translating, the challenge of this study was not translating the meaning of the given sentences according to the context. But this study has not solved that

challenge which needs further study to show all meanings of word depending on the context properly.

Million Meshesha and Yitayew Solomon [69] developed English-Afaan Oromo statistical machine translation. The study investigated the effect of word, phrase, and sentence levels of alignment on English-Afaan Oromo statistical machine translation. The corpora were collected from the Criminal code, FDRE constitution, Megleta Oromia and Holly Bible. A total of 6400 simple and complex sentences were used. Mosses was used for the translation process, MGIZA++, Anymalign, and hunalign tools for word phrase, and sentence level alignments respectively, and IRSTLM for language modelling. Data cleaning was performed during the pre-processing stage to make the data set ready for alignment and experimentation. 19300 and 12200 sentences were used as a monolingual corpus for creating English and Afaan Oromo language models, respectively. The experimental result showed that 27% BLUE score were recorded at phrase level alignment with maximum phrase length of 16. The study concluded that alignment has a great effect on the quality and accuracy of statistical machine translation between both language pairs.

3.4 Machine Translations Using Neural Network

CNN Based Translations

Arfaso Birhanu [70] worked on a bi-directional text-based machine translation for English and Afan Oromo language pairs using convolutional neural networks. The study started with the objective of improving the previous work on English to Afan Oromo machine translation by making the translation bi-directional utilizing convolutional neural network on translations between these language pairs. A total of 5550 sentences of parallel corpus from different sources were collected. The sources of the parallel corpus include religious books like the Holy Bible, data collected from some published conversational books, data of online available conversational sentences and data from governmental sources like constitutions, data from Oromia regional revenue and data from Oromia health sector. After collecting datasets, aligning, and storing the data, the data elements were shuffled before classifying into training set and testing set. The division of the dataset into training and test set was on 80 % to 20% basis respectively. Three systems were implemented where the first system uses a word-based statistical approach that was used as a baseline, while the second system with recurrent neural network approach was used as a competitive model and lastly, the third system with convolutional neural networks for the bi-

directional translation between Afan Oromo and English languages. After training and testing these systems on corresponding training and testing datasets, the convolutional neural network achieved 3.86 BLEU score improvement on translation from English to Afan Oromo and 3.32 BLEU score on translation from Afan Oromo to English translation than baseline system. In addition to this, convolutional neural network approach has shown an improvement of 1.58 BLEU score on translation from English to Afan Oromo and 1.51 BLEU score on translation from Afan Oromo to English translation than recurrent neural network approach. The study concluded the explanation by showing convolutional neural network approach performing faster on training than recurrent neural network approach.

RNN Based Translations

Genet Worku [12] worked on an NMT encoder-decoder translation model for Ge'ez-Amharic translation based on attention mechanism that contains two LSTM layers with 500 hidden units both in the encoder and decoder parts. The model takes source sentence as input in the encoder side and generates a target sentence as output in the decoder side, producing a single word at a time. The study used attention mechanism to handle long term dependencies in long sentences by paying attention to the parts of the input sentence which contain relevant information in generating a single word in the target sentence. Ge'ez-Amharic parallel sentences were collected from religious institutions and other related sources such as Ethiopian Orthodox Tewahdo church religious books including the Holy Bible (old, and new testaments), wudasie Mariam, Dirsane Gebriel, and Dirsane Michael. An OpenNMT was used for developing the model and BLEU in evaluating the translation quality of the model. The model was trained for different experiments on Colab and a BLEU score of 15.4% was recorded.

Workineh Wogaso [71] developed an attention-based Amharic-to-Wolaita neural machine translation. The system was developed based on encoder-decoder architecture also called sequence-to-sequence model by integrating an RNN and GRU. For comparison of the translation performance, non-attention-based Amharic-to-Wolaita NMT was developed. An encoder in non-attention-based encoder-decoder architecture encodes the complete information of the source sequence into a single real-valued vector called context vector which is passed to the decoder to produce an output sequence. The total dataset used for the system training and testing was 9280 parallel sentences collected from different literature domains mainly from religious books. To

classify the total size of the corpus into training and testing set, the 80% by 20% rule were used. A BLEU score evaluation technique was used for system evaluation and a BLEU score of 59.6% for non-attention-based system and 62.58% BLEU score for attention-based system is recorded.

Ibrahim Gashaw and Shashirekha [72] worked on Amharic-Arabic neural machine translation. An Open NMT system was used to construct the NMT model and translate the text in Amharic to the Arabic language. After extensive experiments, the maximum source and target sequence length were set to 44, maximum batch size for training and validation was set to 80 and 40 respectively and learning rate to 0.001 with Adam optimization for both LSTM and GRU recurrent neural network type. The remaining parameters were used as default. The system saves the model for each of 10,000 training samples and then computes accuracy and perplexity of each model 10 times. Since Amharic and Arabic languages lack parallel corpus for the purpose of developing NMT, small size Amharic-Arabic parallel text corpora have been constructed to train the Amharic to Arabic NMT system by splitting the verses manually into separate sentences of Amharic language as a source sentence and Arabic language as a target sentence. The parallel corpus for this experimentation were prepared from the parallel Quranic text corpus by modifying the existing monolingual Arabic text and its equivalent translation of Amharic language text corpora available on Tanzile. Using the constructed corpus LSTM and GRU based NMT models and Google Translation system were compared and found that LSTM based OpenNMT outperforms GRU based OpenNMT and Google Translation system, with a BLEU score of 12%, 11%, and 6% respectively.

Esan et.al [73], conducted a recurrent neural network model for English to Yoruba machine translation. Parallel corpus was acquired from the English and Yoruba bible corpus. The developed model was tested and evaluated using both manual and automatic evaluation techniques. Results from manual evaluation by ten human evaluators show that the system was adequate and fluent. The result from automatic evaluation confirms that the developed model has decent and good translation as well as higher accuracy as it has better correlation with human judgment. The overall average adequacy and fluency metrics were 52.65% and 54.72%, respectively. This confirms that using NMT yields better performance.

TNN Based Translations

Andargachew Mekonnen, et al. [74] worked on neural machine translation for Amharic-English translation and described neural machine translation between orthographically and morphologically divergent languages. It is obvious that Amharic has a rich morphology. A new transliteration technique was used to facilitate vocabulary sharing for Amharic. Sub-words have also been used to tackle the highly inflectional morphology and make an open vocabulary translation. Furthermore, the research was conducted on low data conditions. A transformer-based neural machine translation architecture was used by tuning the hyperparameters for low-data conditions. The models were trained on the public benchmark dataset, Amharic-English parallel corpus, consisting of 140 thousand sentence pairs. The development and test sets have 2864 and 2500 sentence pairs. All the Amharic datasets were transliterated with AT4MT, they were tokenized with an in-house tokenizer. The English data sets were tokenized with Moses' script. For word-based models, the authors used a shared vocabulary of the top 44000 most frequent tokens. For sub-word-based models; word-piece, Byte Pair Encoding (BPE), and Unigram Language Model (ULM) were used. For all techniques, the segmentation models were built using the training datasets of both languages separately. To make a comparison among the segmentation schemes, tokens were segmented into 32000 sub word vocabularies using word-piece, BPE, or ULM. In the automatic evaluation of the strong baseline, word-based, and sub-word-based models trained on a public benchmark dataset, the best sub-word-based models outperform the baseline models by approximately six up to seven BLEU.

Nguyen et al. [75] developed transformer model which radically and fundamentally changes machine translation with its self-attention and cross-attention mechanisms compared with traditional statistical machine translation models and other neural machine translation models. The self-attention and cross-attention mechanisms effectively model token alignments between source and target sentences. It has been reported that the transformer model provides accurate posterior alignments. This work, empirically proved the reverse effect, showing that prior alignments help transformer models produce better translations. Experiment results on Vietnamese-English news translation task show not only the positive effect of manually annotated alignments on transformer models but also the surprising outperformance of statistically constructed alignments reinforced with the flexibility of token-type selection over manual alignments in improving transformer models. Statistically constructed word-to-lemma alignments were used to train a word-to-word transformer model. Fairseq library was selected to carry out the experiments, as it allows to build

both transformer models trained with/without prior alignments. Following the architecture and training procedure for transformer models, the study applied Adam optimizer with learning rate 0.0002 to train them in 10,000 steps of 3200 tokens. The model translates all Vietnamese sentences from the testing dataset, deploying a beam search of size 5. The predicted English sentences were then compared with the corresponding reference English sentences from the testing dataset via BLEU score. The study found that the transformer model trained with manual prior alignments significantly outperformed the baseline transformer model which scores a BLEU of 16.26% and 14.52% respectively. A novel hybrid transformer model improves the baseline transformer model and transformer model trained with manual alignments. In addition to BLEU score, limited human judgment were made on translation results. Strong correlation between human and machine judgment confirmed the findings.

3.5 Machine Translations Using Hybrid Approach

Biruk Abel [76] worked on developing a Hybrid Geez to Amharic Machine Translation system using serial coupling of rule-based Geez language word reordering followed by a standard SMT system. The proposed system was composed of two main components a rule based Geez corpus pre-processor and a baseline SMT. The rule-based pre-processor takes the manually POS tagged Geez corpus and produces another corpus that contains reordered Geez sentences having similar structure with that of Amharic sentences. The dataset used contains 976 manually POS tagged Geez sentences perfectly aligned with their possible Amharic translations. This component contains a set of activities that process each Geez sentence in the input corpus one by one to determine POS pattern and subsequently apply the corresponding reordering rule. It first reads all sentences from the input file and iterates through all sentences, then it determines POS pattern and applies the corresponding reordering rule. After each sentence is processed the output corpus along with the Amharic corpus will be supplied as an input to the Baseline SMT. Then using the input corpora, the actual translation of Geez sentence to Amharic sentences were performed by the decoder of the baseline SMT by using the language model of Amharic and the translation model. The translation quality of the system was evaluated using BLEU evaluation metrics and compared with that of the Baseline SMT. Two experiments were conducted, one to test the baseline SMT and the other to test the hybrid system. To test the baseline SMT both Geez and Amharic corpus without POS were used while to test the hybrid system Geez corpus with POS and Amharic corpus

with no POS were used. Based on the test results the baseline SMT scored a BLEU of 72% and the proposed system outscores it by 4% and scored 76% owing to the reordering rules applied on Geez corpus. Though the score of this hybrid method is very high and recommended to use this approach, the incorporation of the rule based technique makes it resource intensive.

Jabesa Daba [77] operated on the objective of developing a bidirectional English–Afaan Oromo machine translation using hybrid approach. This study resulted in the development of a bidirectional English-Afaan Oromo machine translation system using a hybrid approach and the research work was implemented using a hybrid of rule based and statistical approaches. The first experiment was carried out by using a statistical approach and the result obtained from the experiment has recorded a BLEU score of 32.39% for English to Afaan Oromo translation and 41.50% for Afaan Oromo to English translation. The second experiment carried out by using a hybrid approach and the result obtained has a BLEU score of 37.41% for English to Afaan Oromo translation and 52.02% for Afaan Oromo to English translation. The researchers recommended that the rules which were developed and used in the system were only used for syntax reordering and additional results could be accomplished by further exploring the rules, especially by developing morphological rules.

Aukubazgi GebreMariam [14] developed a translation from Amharic to Tigrigna using hybrid approach. The corpus was composed of simple and complex sentences collected manually from different sources such as literal documents, social medias, and websites and revisit data. The size of the corpus was 1582 parallel sentences. The sentences were translated to each other with the help of language experts and prepared in parallel. Other sentences were taken from Bible, federal and regional constitution which contains Amharic and Tigrigna sentences in parallel. 1482 parallel sentences were used to train the system. Moses, which is freely available software was used to train the system from Amharic to Tigrigna machine translation. Two major experiments were conducted using two different approaches and their results were documented. The first experiment carried out using a statistical approach and the result obtained from the experiment has a BLEU score of 7.02%. The second experiment, carried out using hybrid approach, increased the performance of the system and obtained a BLEU score of 17.47%. This low score can be improved by using latest state-of-the art techniques and increasing the corpus size.

Pal et al. [43] worked on developing a hybrid MT framework for English to Germany. The study utilized all the parallel training data provided by the WMT 2015 shared task organizers for English–German translation. The training data include Europarl, News Commentary and Common Crawl. The provided corpus was noisy and contains some non-German as well as non-English words and sentences. Language identifiers were applied on both bilingual English–German parallel data and monolingual German corpora and parallel sentences detected as belonging to some different language by the language identifier were discarded. The same method was also applied to the monolingual data. The effectiveness of the work was demonstrated by using the standard log-linear PB-SMT model as a baseline system. For building the baseline system, a phrase of maximum length of 7 and a 5-gram language model were used. The other experimental settings were: SymGIZA++ aligner, which is a modified version of GIZA++ word alignment models by updating the symmetrizing models between chosen iterations of the original word alignment training algorithms and phrase extraction. The reordering model was trained on hier-mslr-bidirectional, that was using both forward and backward models. The hybrid system brought an improvement over the baseline arrangement by incorporating additional knowledge such as extracted bilingual named entities and bilingual phrase pairs induced from example-based methods. The baseline approach marks a BLEU score of 16.7% and 22.6% BLEU score was registered for the hybrid scheme.

3.6 Summary

In the realm of neural machine translation, the Geez-to-Tigrigna language pair remains largely unexplored. Despite the growing body of research in the field, no comprehensive studies have been conducted to develop a dedicated neural machine translation system specifically designed for this language pair. This research gap presents an exciting opportunity to investigate the application of state-of-the-art transformer-based models in this context. However, the low-resource nature of both the source (Geez) and the target (Tigrigna) languages poses exclusive challenges to the development of an effective translation system. The scarcity of available parallel corpora further compounds the complexity of the research work. By addressing this gap, we can make important advancements in Geez-to-Tigrigna neural machine translation, paving the way for new discoveries and contributions to the broader field of machine translation.

Chapter 4: Architecture of the Geez to Tigrigna NMT

4.1 Introduction

This chapter discusses the detailed procedures of how this thesis work is designed and implemented. It aims to provide a comprehensive understanding of the system architecture and its components, explaining the inner workings of this transformative technology. By investigating into the design of the system, we explore the processes involved in the automatic translation of text from Geez to Tigrigna language. The chapter outlines the overall system architecture, highlighting the key components and their interaction. Additionally, the algorithms employed to accomplish the objectives of the thesis are thoroughly examined, considering their role in achieving accurate and reliable translations. Through a careful exploration of the system's methodology and algorithms, we gain valuable insights into the underlying mechanisms that drive the Geez to Tigrigna machine translation system, paving the way for enhanced communication, and linguistic preservation. Transformer based NMT is the state of the art and employed for the accomplishment of this thesis.

4.2 Architecture of the System

The architecture of a transformer based NMT model for language translation consists of an encoder and a decoder. The encoder processes the source language sentence to obtain a representation that captures contextual information, while the decoder generates the translated sentence word by word using the representation generated by the encoder. The encoder consists of self-attention layers, and feed-forward neural networks. The decoder also consists of self-attention layers, and feed-forward neural networks. In addition to self-attention, the decoder applies masking to ensure that each word can only attend to previous words during decoding. The encoder-decoder attention layer computes attention weights between the current word being generated and the representations generated by the encoder, helping the decoder focus on relevant parts of the source sentence. The feed-forward neural network captures complex patterns in the data after attention. So, the encoder-decoder architecture allows the model to effectively encode the source sentence and generate the translated sentence by attending to relevant parts of the input sentence. The network maps an input sequence $X = \{x_1, x_2, x_3, \ldots, x_n\}$ to an output sequence $Y = \{y_1, y_2, y_3, \ldots, y_n\}$. The encoder takes the input sequence and yields the intermediate output sequence $Z = \{z_1, z_2, z_3, \ldots, z_n\}$. Given Z,

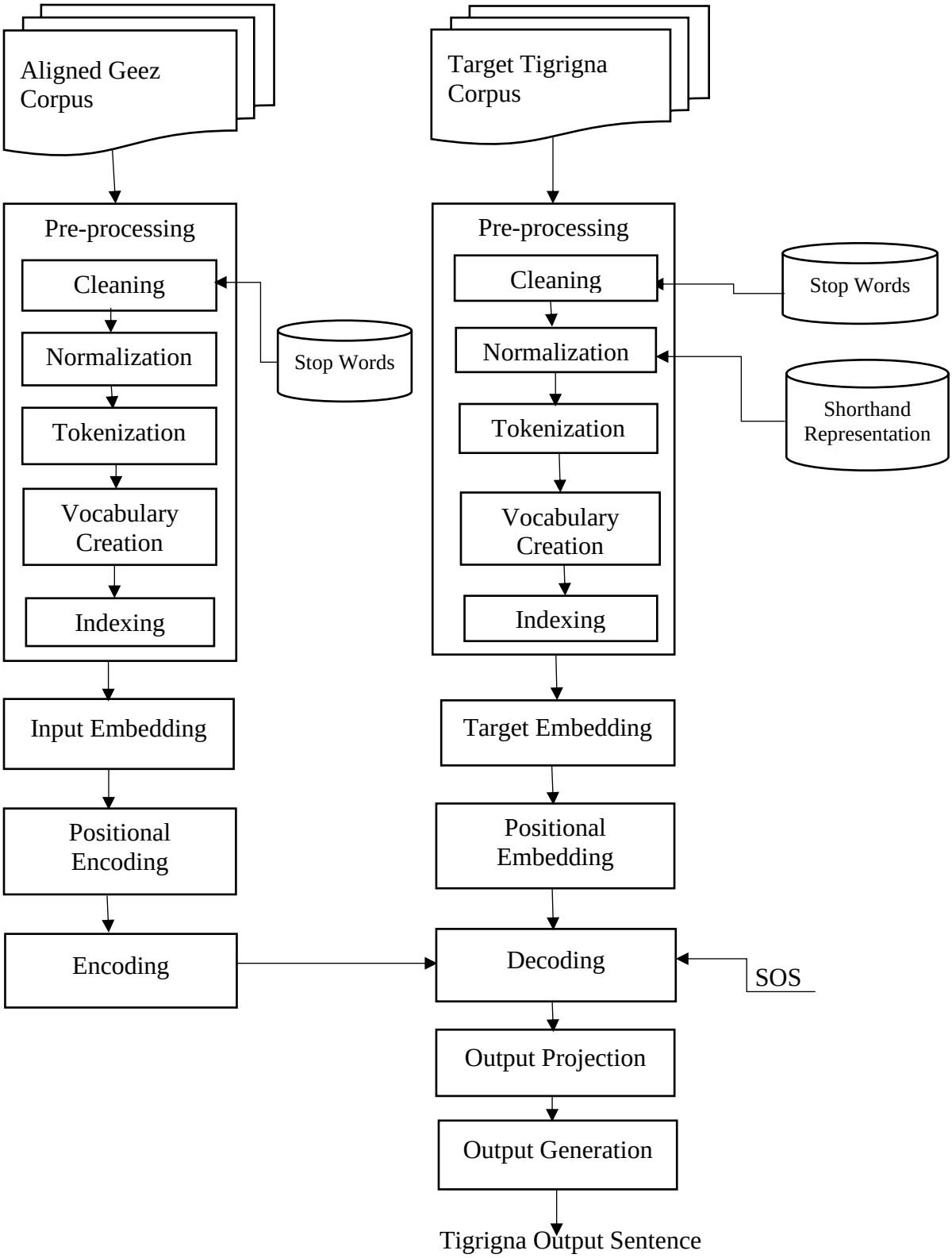


Figure 4.1 Architecture of the Transformer Based Geez to Tigrigna NMT

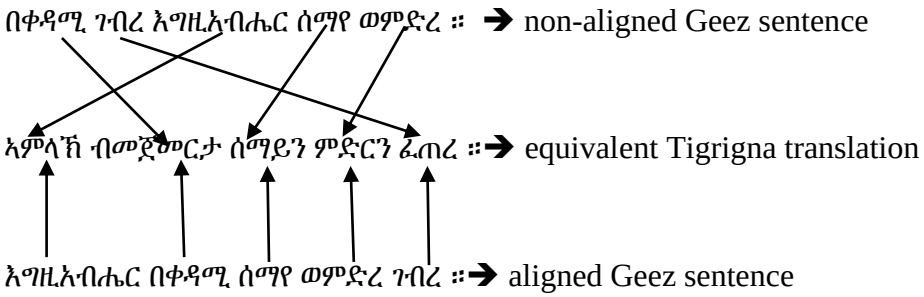
the decoder then generates the final output Y. Transformer uses this general structure with a stack of encoders and decoders. The process begins with the Geez text being inputted into the encoder,

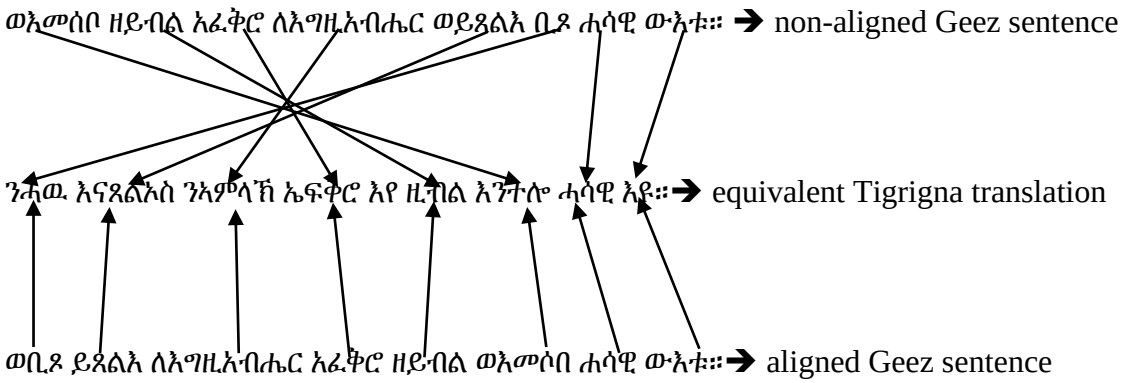
where it is converted into a thought vector that captures the essence of the sentence. This condensed representation is then inputted into the decoder, along with the original translation in the Tigrigna language, during the training process. This acts as a conditioning factor for the decoder, allowing it to learn the appropriate translation based on the original input during training. Because encoder-decoder structure addresses the translation task from source data directly to the target result, which means there is no visible result in the middle process, this is also called end-to-end translation. Thus, the principle of encoder-decoder structure of NMT is viewed as mapping of the source sentence with the target sentence via an intermediate vector in semantic space and the intermediate vector represents the same semantic meaning in both languages. Figure 4.1 depicts the architecture of the system.

As explained in Section 2.6, transformer is a model that does not rely on recurrence and convolution, rather on self-attention mechanism. It also allows parallelization. The attention mechanism enables the transformers to have extremely long-term memory. While self-attention layer computes attention weights for each word in the input sentence allowing the model to focus on different parts of the sentence when encoding it, feed-forward neural network captures complex patterns in the data after self-attention. It is the self-attention mechanism in transformers that enables better modelling of long-range dependencies, making it a powerful architecture for NMT tasks. Word alignment of the Geez sentence using the Tigrigna equivalent is applied in this thesis.

Word alignment is the way of adjusting the order of the words in a source language sentence to match the target language's word arrangement. Word alignment plays a crucial role in the performance of transformer based NMT systems. The quality of word alignment directly affects the ability of the model to accurately translate between languages. Accurate word alignment helps the model to better understand the relationship between words in the source and target languages. This, in turn, leads to more accurate translations with better fluency and coherence. A well-aligned dataset helps in improving the robustness of the transformer based NMT model, making it more robust to variations and noise in the input data. Word alignment helps in capturing the correspondence between words in the source and target sentences. By aligning words correctly, the model generates more accurate translations. This enables the model to understand the relationships and dependences between words in different languages, leading to more precise and

contextually appropriate translations. The two examples below help visualize the complexity of the aligned and non-aligned sentences.





As we can simply observe it from the descriptions, the alignment of the words has a great impact on the performance of the translation. The overload of pairing each source word with its equivalent translation is greatly reduced while the alignment of the input is adjusted with the order of the output. This rearrangement of words has a great role on the improvement of our translation performance. It helps the model better capture dependencies between different parts of the input and output sequences, which lead to more accurate translations. The aligned source sentences are tokenized and fed into an embedding layer to obtain source embeddings. Target sentence remains the same as it is in its original arrangement and fed into an embedding layer to obtain target embeddings. Positional encoding vectors are added to both source and target embeddings. Source embeddings are passed through multiple encoder layers. Self-attention allows each source word to attend to all other source words. The encoder then generates hidden states or the context vectors, these are numerical representation of the source tokens.

Each element of the transformer architecture has their own task to perform and feed their intermediate results to the next component till the output is computed. After the preliminary

processing of the data, the input text is fed to be encoded. This process starts with input embedding and embedding layer is responsible for this.

4.2.1 Components of the Architecture

4.2.1.1 Pre-processing

Pre-processing is the way raw data is prepared and applied to the machine learning model. It is the initial step in any machine translation endeavour. The main purpose of the data pre-processing stage is to facilitate network training. It involves several steps to prepare the input and output sentences for training the model. The idea is to perform preliminary processing of the data to make it easier for the neural network training to extract the relevant information. The sub-tasks that we need to execute for our model under data pre-processing are cleaning, normalization, tokenization, vocabulary creation, indexing, and padding.

Cleaning: cleaning is the process of preparing raw text for NLP, the stage where noisy symbols are removed from the dataset. The algorithm that performs cleaning is given in Algorithm 4.1. It includes removal of stop words, special characters, numerals, punctuation marks, unwanted spaces, and other non-Geez characters. The list of some of the stop words is given in Appendix E. The data cleaning process also involve removing other parts of the data that did not add value to the translation meaning.

Normalization: text normalization is a pre-processing step aimed at improving the quality of the text and making it suitable for machines to process. In this thesis, we consider about short form expansion. Shorthand representations with a forward slash or a period are expanded into their longer form of writing. For example, ዓ/ም (*ʔạ/mə*) is expanded as ዓመተ ምህረት (*ʔạmätä məḥərätə*), and ቤት ት/ቲ (*betə tə/ti*) is written in its longest form as ቤት ትምህርቲ (*betə təməhərəti*). Geez alphabets have unique meaning by their nature even though they are pronounced the same way. For example, the Geez words ሰኣሊ (*säʔali*) and ሰዓሊ (*säʔạli*) have different meanings; አመት (*ʔamätə*) and ዓመት (*ʔạmätə*) are another example with different meanings, but the same sound '*säʔali*' and '*ʔamätə*' respectively. In the modern Tigrigna alphabet table, there are no characters that are scripted in different symbol and pronounced the same way. But in some literatures, the letters ሠ(*ṣä*) and ሰ(*sä*), ጸ(*tsä*) and ፀ(*tṣä*) are used interchangeably. Letter ኀ(*ḫä*)is also one of those omitted in Tigrigna. A character level normalization for these three letters is done in Tigrigna corpus normalization. The algorithm for normalization is given in Algorithm 4.2.

Algorithm 4.1: Cleaning Algorithm

```
Open the document
While (Not end of the file)
    Read the character
    If the character is found in the stop words list
        Remove it
    Else if character is one of the special characters
         Remove it
    Else if character is non-Geez
         Remove it
    Else
       Continue processing
    End if
 End while
```

Tokenization: tokenization is basically splitting a sentence into smaller pieces, such as individual words. Each of these smaller units is called a token. Tokenization transforms the actual normalized sequence of sentences into space-separated token strings. We applied Byte Pair Encoding (BPE) for tokenization. It works by iteratively merging the most frequent pairs of characters or sub-words in the corpus, gradually building a vocabulary of tokens. In tokenization, words are typically split into sub-word units based on their frequency and linguistic properties. For illustration, when the Geez sentence "በተአምኖ ዐደውዋ ለባሕረ ኤርትራ ከመ ዘውስተ ምድር ይቡስ ወኮነቶሙ መከራ ለግብጽ ተሠጢሞሙ" is tokenized, we get the output "በተአምኖ", "ዐደውዋ", "ለባሕረ", "ኤርትራ", "ከመ", "ዘውስተ", "ምድር", "ይቡስ", "ወኮነቶሙ", "መከራ", "ለግብጽ", "ተሠጢሞሙ". The meaning of the sentence in Tigrigna "ብእምነት ከም ብንቐጽ ምድሪ ገይሮም ንባሕሪ ኤርትራ ተሳገርዎ ግብጻውያን ግና ከምኡ ኺገብሩ ምስ ፈተኑ ተዋሕጡ" is also tokenized as "ብእምነት", "ከም", "ብንቐጽ", "ምድሪ", "ገይሮም", "ንባሕሪ", "ኤርትራ", "ተሳገርዎ", "ግብጻውያን", "ግና", "ከምኡ", "ኺገብሩ", "ምስ", "ፈተኑ", "ተዋሕጡ". The tokenization is based on the occurrence of punctuation marks and white space. A method from Tokenizer class (*tf.keras.preprocessing.text.Tokenizer*) is used for tokenization in our case. The tokenized words are then fed into the model for further processing, such as feature extraction or input embedding, before generating translations.

Algorithm 4.2: Normalization Algorithm

```
Open the document
While (not end of the file)
  Read character
  If character is one of punctuation marks, white space and other alphabets
     Continue
   Else if character is /or .
       Read characters before and after /or .
       If characters found in the short words list
            Replace the short form with its expanded form
       Else return
    Else if character is ሠ replace it with ሰ
          Return the replaced character
    Else if character is ኀ replace it with ሀ
          Return the replaced character
    Else if character is ጸ replace it with ፀ
          Return the replaced character
          Do for the rest six families of the three alphabets
          ሡ ሢ ሣ ሤ ሥ with ሱ ሲ ሳ ሴ ስ ሶ
           ኍ ኂ ኃ ኄ ኅ ኆ with ሁ ሂ ሃ ሄ ህ ሆ
          ጱ ጲ ጳ ጴ ጵ ጶ with ፁ ፂ ፃ ፄ ፅ ፆ
       Else continue
     End if
 End while
```

Vocabulary Creation: we create our vocabulary by collecting all unique tokens from the source and target sentences. The most frequent tokens are typically included in the vocabulary. Algorithm 4.3 shows the creation of the vocabulary. Creating an effective vocabulary is just one aspect of building a robust NMT system. This vocabulary acts as a look up table for mapping tokens to numerical representations used by the model. The *counter* class from the *collections* module provides a convenient way to count the occurrences of elements in a collection or a list and returns the result as a dictionary like object. This vocabulary construction algorithm reads the corpus file,

splits it into sentences, tokenizes the sentences, counts the token frequencies, selects the most common tokens, and creates the final vocabulary list. The vocabulary list includes the <sos> and <eos> tokens, which are commonly used in natural language processing tasks.

Algorithm 4.3: Vocabulary Creation Algorithm

```
Open the file
While (not end of the file) do
    Read the sentence
    Tokenize the sentence
    Append the new tokens to the tokens list
    Count the frequency of each token in the tokens list
    Create the vocabulary list by adding the <sos> and <eos> tokens
    Return the vocabulary list
End while
```

Indexing: in NMT, the input and output sequences are typically represented as sequences of integers, where each integer corresponds to a specific word in the vocabulary. So, indexing is a step that converts the tokens into numerical representations that can be processed by the model. We use one-hot representation for this purpose, and it is a technique used to represent these sequences as a dense vector. Each token in the source and target sentences is assigned a unique index based on its position in the vocabulary. Let us consider the Geez sentences **በቀዳሚ ንብረ እግዚአብሔር ሰማየ ወምድረ ።** (*bäqädami gäbärä ʔəgəziabəherə sämayä wämədärä* 'in the beginning God created heaven and earth') and **ወኮነ ብርሃን ።** (*wäkonä bərəhanä,*'it became light'). From Tigrigna sentences **ኣምላኽ ብመጀመርታ ሰማይን ምድርን ፈጠረ ።** (*bäqädami gäbärä ʔəgəziabəherə sämayä wämədärä* 'in the beginning God created heaven and earth') is used. Then the unique words in these sentences are listed under the vocabulary as:

```
Vocabulary: ['ምድርን', 'ሰማየ', 'ሰማይን', 'በቀዳሚ', 'ብመጀመርታ', 'ብርሃን', 'ኣምላኽ', 'እግዚአብሔር', 'ወምድረ', 'ወኮነ', 'ንብረ', 'ፈጠረ', '።']
```

The one-hot encoded vector is created by taking the number of tokens in each sentence as a row and size of the whole vocabulary as column. In this example we have 6 tokens in '**በቀዳሚ ንብረ እግዚአብሔር ሰማየ ወምድረ ።**' and 3 in '**ወኮነ ብርሃን ።**'. Similarly, for '**ኣምላኽ ብመጀመርታ ሰማይን ምድርን ፈጠረ ።**' the total number of tokens are 6. Therefore, the size of the vocabulary will be 13. Figure 4.2 shows the one-hot encoded matrix of the three sentences.

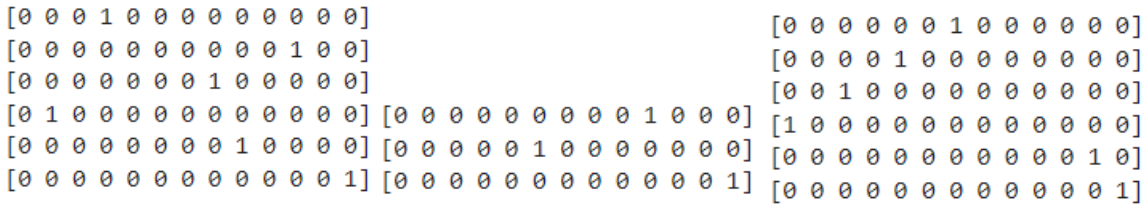


Figure 4.2 One-hot Encoding

(A) ‘በቀዳሚ ገብረ እግዚአብሔር ሰማየ ወምድረ ።’

(B) ‘ወኮነ ብርሃን ።’

(C) ‘ኣምላኽ ብመጀመርታ ሰማይን ምድርን ፈጠረ ።’

Padding: Transformers operate on a fixed-length sequences, and to ensure that all sentences have the same length, padding is applied by adding special tokens such as ‘padding’ token to shorter sentences. By padding we ensure that the input sequences have consistent dimensions, which is necessary for efficient batch processing. The padded tokens do not carry any semantic meaning and are just used to fill up the extra space in shorter sequences. During training, the model learns to ignore the padding tokens and focuses on the actual content of the input sequence. The function *tf.keras.preprocessing.sequence.pad_sequences* is used for padding.

4.2.1.2 Input Embedding

Since words cannot be passed as they are, we begin the transformation by changing the input tokens into a continuous vector representation which capture the semantic and syntactic information about the words. Embedding layer converts the words into dense representations, which are then transformed into a projection of all the words in the target vocabulary by the decoder. It is the mapping of input tokens to sequence of vectors. The final translated words are picked from the projection based on the predicted probabilities. So, each input word will be turned into vectors using an embedding algorithm. These vectors are learned based on the co-occurrence patterns of words in a corpus of text. For this thesis we used the FastText word embedding algorithm, which is a library for efficient learning of word embeddings. Its main purpose is to generate vector representations or embeddings for words in the text corpus. FastText approach of considering character n-grams also makes it particularly useful for handling morphologically rich languages. Let us consider the sentence ‘በቀዳሚ ገብረ እግዚአብሔር ሰማየ ወምድረ’ (*bäqädami gäbärä ʔəgəziabəherə sämayä wämədərä*) and look at how the embedding is performed. FastText handles

the sentence by breaking it down into sub-word units and generating representations based on those units. And let us set the character-level n-gram size to 3, the word ‘በቀዳሚ’ can be broken down into the following sub-word units: ‘በ’, ‘በቀ’, ‘ቀዳ’, ‘ዳሚ’, and ‘ሚ’. Similarly, the other words in the sentence can also be represented by their respective sub-word units. With the sub-word units identified, FastText generates word embeddings by considering both whole words and their sub-word units. Each sub-word unit is associated with a learned vector representation. For instance, the sub-word unit ‘በቀ’ would have a corresponding vector representation. To obtain the representation for a whole word, the vector representations of its sub-word units is summed up. In the case of the sentence ‘በቀዳሚ ንብረ እግዚአብሔር ሰማየ ወምድረ’, each word is represented as the sum of their sub-word unit vectors. This handles Out of Vocabulary (OOV) words too. In addition, FastText captures the morphological information present in the sub-word units and generate meaningful representations, even for OOV words. This is particularly useful for this thesis as Geez and Tigrigna are low-resource languages with complex morphology where there may be many rare or unseen words. Algorithm 4.4 illustrates the embedding procedure.

Algorithm 4.4 Embedding Algorithm

```
Open the file
While (not end of the file)
    Initialize a fastText model with the tokens list
    For (each token in the vocabulary)
      If the token is in the model vocabulary
            Retrieve the embedding vector for the token from the model
            Append the embedding vector to the embeddings list
      Else
         Generate a random embedding vector of appropriate size
         Append the random embedding vector to embeddings list
    Return the embeddings list
End while
```

After embedding the words in the input sequence, each of them flows through each of the two layers of the encoder, first through the self-attention and then the feed forward layer. To summarize the concept, the target of word embedding is to turn text into numbers. It is a means of building a

low-dimensional vector representation from corpus of text, which preserves the contextual similarity of words. In transformer, the words in each position flow following their own path in the encoder and hence a means to tell the encoder of the position of each word comes into consideration.

4.2.1.3 Positional Encoding

As explained in Section 2.6.2, transformers process all the sequence once in parallel and have no perception of word order, and thus there may be disagreement or a mismatch on the position of the tokens. The way transformers handle this situation is by positional encoding. Positional encoding is added to provide positional information to the model since transformers do not have inherent word order. So, the main point of positional encoding is to trigger a neural network to the position of each word in a sentence and to each embedding-vector cell for each word. Positional encoding can be achieved by first constructing an array of floating-point values and then adding that array of numbers to the input sequence. As a result, we obtain a special embedding with positional information. To incorporate positional information, transformers use sine and cosine functions with different frequencies and offsets [35, 59]. The positional encoding is added to the word embeddings at each position in the input sequence. The formula for calculating the positional encoding is then given by equations (9) and (10):

$$\mathrm{PE}_{(\mathrm{pos},2\mathrm{i})} = \sin\left(\frac{\mathrm{pos}}{1000^{\frac{2\mathrm{i}}{\mathrm{d}_{\mathrm{model}}}}}\right) \tag{9}$$

$$\mathrm{PE}_{(\mathrm{pos},2\mathrm{i}+1)} = \cos\left(\frac{\mathrm{pos}}{1000^{\frac{2\mathrm{i}}{\mathrm{d}_{\mathrm{model}}}}}\right) \tag{10}$$

where pos represents the position of the token in the sequence, i represents the dimension index, and d_{model} is the dimensionality of the word embeddings.

Now let us consider the sentence ‘በቀዳሚ ንብረ እግዚአብሔር ሰማየ ወምድረ’ and demonstrate how positional encoding vectors can be applied. The individual tokens are [‘በቀዳሚ’, ‘ንብረ’, ‘እግዚአብሔር’, ‘ሰማየ’, ‘ወምድረ’]. The positional encoding vectors to each token is then assigned based on its position in the sequence. By taking a d_model of 512, we will calculate the positional encoding vectors for the given tokens using the formulas. The token at position zero is ‘በቀዳሚ’ and the simplified calculation of the positional encoding vector is shown in Table 4.1.

The positional encoding vectors are then added to the corresponding word embeddings for each token to form the final input embeddings. The input embeddings, enriched with positional information, are then fed into the encoding layers of the transformer model for further processing. This allows the model to capture sequential information and understand the order of words in the input, which is crucial for accurate translation. At the end of this step, FastText generated word embeddings along with the positional encoding are used as input to the encoder allowing the model to benefit from the semantic information captured by FastText. Figure 4.3 demonstrates this process.

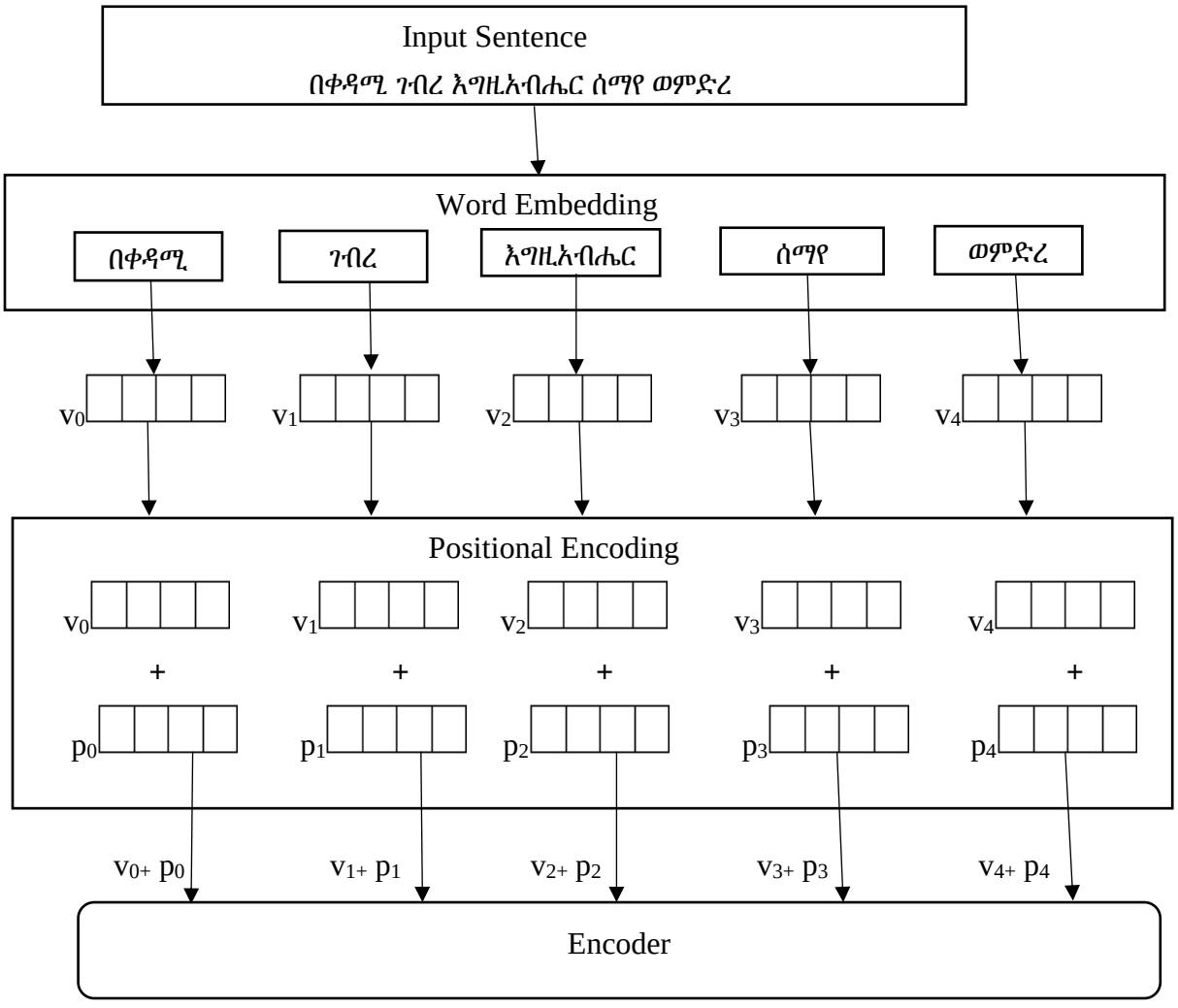


Figure 4.3 Summary of Word Embedding and Positional Encoding

While the V's in the block diagram represent the vector representations of the word embeddings, the P's are for positional encoding vector representations.

Table 4.1 Example of Calculating Positional Encoding Vector

Position	Token	Positional Encoding Vector
0	በቀዳሚ	[sin (0/10000^(0/512)), cos(0/10000^(0/512))]
1	ጉበረ	[sin (1/10000^(0/512)), cos(1/10000^(0/512))]
2	እግዚአብሔር	[sin (2/10000^(0/512)), cos(2/10000^(0/512))]
3	ሰማየ	[sin (3/10000^(0/512)), cos(3/10000^(0/512))]
4	ወምድረ	[sin (4/10000^(0/512)), cos(4/10000^(0/512))]

4.2.1.4 Encoding

The process of encoding in transformer NMT involves converting the continuous vector representations of input tokens into contextualized representations that capture the relationships between the tokens. This is achieved by using an encoder, which includes multiple layers of self-attention and feed forward neural networks. A vector with positional information is fed into the encoder as it is shown in Figure 4.3. It processes this list by passing these vectors into a self-attention layer and then into a feed forward neural network. The output of the intermediate encoder is then sent to the next encoder as each of the encoding block is a stack of many encoders. We used four encoder stacks in this thesis. As the model processes each input sequence, self-attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word. The stack of encoders receives one vector per token, and it returns a new vector per token as a result. The last encoder output is fed into every decoder. There are two layers in a single encoder layer and these layers perform different operations and feed the next encoder layer. Self-attention layer is a layer that helps the encoder look at other words in the input sentence as it encodes a specific word. An attention function is described as a mapping of a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. Hence the self-attention mechanism is a key component of the transformer model, and it allows each word in the input sequence to attend to all other words, capturing dependencies between them. This is achieved by computing attention weights for each word based on its similarity to other words in the sequence. To illustrate this with an example, let us take the input sentence ‘በቀዳሚ ጉበረ እግዚአብሔር

ሰማየ ወምድረ' and calculate the attention scores. We have reduced the size of the word embeddings for simplicity and each token is mapped to word embedding vector as follows.

በቀዳሚ ➔ [0.2, 0.4, 0.1], ገብረ ➔ [0.3, 0.6, 0.5], እግዚአብሔር ➔ [0.7, 0.9, 0.2], ሰማየ ➔[0.4, 0.8, 0.3], ወምድረ ➔[0.1, 0.5, 0.6]

The word embeddings are linearly transformed to obtain Query (Q), Key (K), and Value (V) vectors for each word and is given in Table 4.2.

Table 4.2 Linear Transformation of the Embeddings

Token	Query(Q)	Key(K)	Value(V)
በቀዳሚ	[0.9, 0.2, 0.7]	[0.3, 0.9, 0.5]	[0.6, 0.3, 0.7]
ገብረ	[0.3, 0.5, 0.6]	[0.8, 0.7, 0.8]	[0.2, 0.8, 0.4]
እግዚአብሔር	[0.8, 0.1, 0.4]	[0.2, 0.4, 0.6]	[0.5, 0.9, 0.3]
ሰማየ	[0.6, 0.2, 0.9]	[0.7, 0.3, 0.9]	[0.1, 0.7, 0.6]
ወምድረ	[0.4, 0.7, 0.3]	[0.8, 0.5, 0.2]	[0.9, 0.4, 0.2]

At this stage we compute the attention scores as the dot product between the query and key vectors, followed by scaling.

-Score (በቀዳሚ, በቀዳሚ) = $(Q \cdot K) / \sqrt{d_k} = (0.9*0.3 + 0.2*0.9 + 0.7*0.5) / \sqrt{3} = 0.4619$

-Score (በቀዳሚ, ገብረ) = $(Q \cdot K) / \sqrt{d_k} = (0.9*0.8 + 0.2*0.7 + 0.7*0.8) / \sqrt{3} = 0.8198$

-Score (በቀዳሚ, እግዚአብሔር) = $(Q \cdot K) / \sqrt{d_k} = (0.9*0.2 + 0.2*0.4 + 0.7*0.6) / \sqrt{3} = 0.3926$

-Score (በቀዳሚ, ሰማየ) = $(Q \cdot K) / \sqrt{d_k} = (0.9*0.7 + 0.2*0.3 + 0.7*0.9) / \sqrt{3} = 0.7621$

-Score (በቀዳሚ, ወምድረ) = $(Q \cdot K) / \sqrt{d_k} = (0.9*0.8 + 0.2*0.5 + 0.7*0.2) / \sqrt{3} = 0.5543$

We did the same for each word pair in the input sentence and attention weight comes next. By passing the scores through a softmax function, we get the attention weights. The attention weights are used to compute a weighted sum of the value vectors, resulting in the attended representation of the input sentence. It is this result that is going to be fed to the feed forward layer. This output

captures the contextual information and dependencies between words in the input sequence. Thus, the self-attention calculation is given by equation (11).

$$\text{Attention}(Q, K, V) = \text{saftmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{11}$$

Where Q, K and V are query, key and value vectors respectively, d_k is the dimension of the query and its corresponding key.

The second layer of an individual encoder is the Feed Forward Neural network (FFN). In transformer based NMT model, the FFN is a key component that is applied after self-attention mechanism in each layer. Its purpose is to further process the information obtained from the self-attention layer and capture additional patterns and dependencies in the input sequence. The FFN consists of two linear transformations with a non-linear activation function in between. Specifically, the input to the FFN is a tensor obtained from the self-attention layer, which typically has a shape of (sequence_length, hidden_size). The FFN applies a fully connected linear transformation to this tensor, followed by a non-linear activation function. A Rectified Linear Unit (ReLU) activation is applied in this case. Mathematically, let X be the input tensor to the FFN. Then the FFN output is given by equation (12).

$$\text{FFN}(X) = \max(0, XW_1 + b_1)\, W_2 + b_2 \tag{12}$$

where W_1 and W_2 are weight matrices, b_1 and b_2 are bias vectors, and max (0, x) represents the ReLU activation function.

The output of the FFN is a tensor of the same shape as the input. This output tensor captures higher-level features and interactions among tokens in the input sequence. Finally, layer normalization is applied after each FFN layer. This helps in stabilizing the training process and improving the gradient flow through the network. Overall, the feed-forward neural network in the transformer NMT model plays a crucial role in capturing complex patterns and dependencies in the input sequence, and the combination of multi-head self-attention, residual connections, and layer normalization allows for effective information flow and training stability in the model. We used *tf.keras.Sequential* class from TensorFlow to implement the FFN.

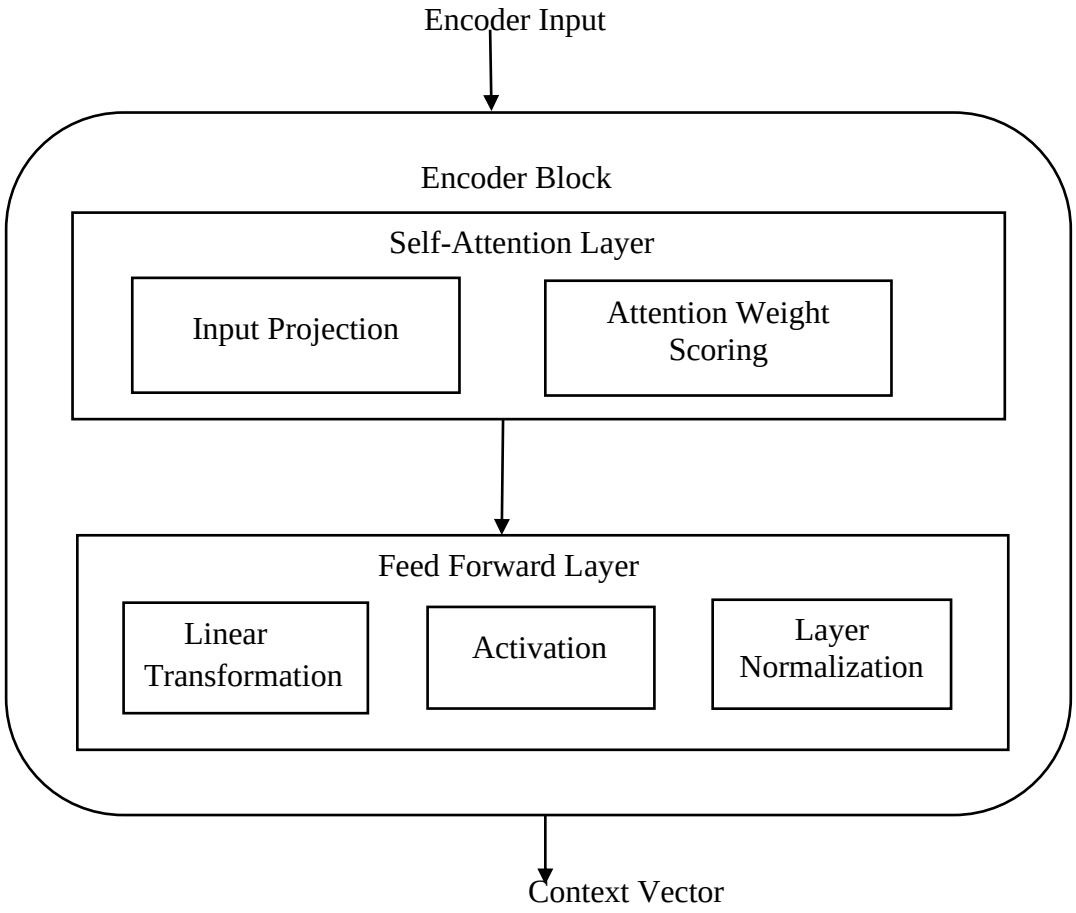


Figure 4.4 Specific Tasks in the Encoder

There is always a residual connection and layer normalization among each layer of the encoding process. Residual connections are added around each self-attention layer. It allows the original input to bypass the layer and be added to the output of the layer. This helps in preserving the information from the input and facilitates the flow of information through the layers. Like the self-attention layer, a residual connection is added around the FFN. This means that the original input tensor X is added elementwise to the output of the FFN. Mathematically, the output of a layer with a residual connection is computed as in equation (13).

$$LayerOutput = LayerInput + Sublayer(LayerInput) \quad (13)$$

where LayerInput is the input to the layer (output of the previous layer), sublayer represents either the self-attention or feed-forward network. The context vector, along with another decoder inputs are then fed to the decoding unit.

4.2.1.5 Decoding

The decoder does the job of the encoder in a reverse direction, that is, decoding the context vector from a mathematical representation to a human-readable sentence in the target language. For the implementation of this thesis, the decoder is also composed of a stack of four identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, called the encoder-decoder attention, which performs multi-head attention over the output of the encoder stack. It helps the decoder to focus on relevant parts of the input sentence. The self-attention sub-layer in the decoder stack is also modified to prevent positions from attending to subsequent positions. This masking, combined with a fact that the output embeddings are offset by one position, ensures that the predictions for the current position can depend only on the known outputs at positions less than it.

The decoder input sequence consists of the encoder output, target language embedding, the predicted word from the previous time step, and the SOS token. The decoder takes the encoded representation of the source language from the encoder and generates the translation in the target language. The output of the top encoder is then transformed into a set of attention vectors K and V. These are to be used by each decoder in its encoder-decoder attention layer which helps the decoder focus on appropriate places in the input sequence.

During training, the target language sequence is provided as input to the decoder, but during inference or translation, the decoder generates each token of the translation one by one based on the previously generated tokens and the encoded source representation. The block diagram on Figure 4.5 shows the decoder block. In the diagram Q_d represents the decoder query and the K_e-V_e pairs represent the encoder key-value pairs. Context $Vector_e$ and Context $Vector_d$ also represent the outputs of the encoder and the self-attention layer of the decoder respectively.

Tasks of the Decoder Block

Like the source language (Geez sentence) pre-processing, target language (Tigrigna sentence) pre-processing includes the steps discussed in Section 4.2.2. All the pre-processing steps are performed on the target language text to prepare it for input into the transformer NMT model for training. Once the pre-processing tasks are completed, the Tigrigna language corpus is ready to be used during the training and decoding phases of the transformer model. During training, the target language is used as ground truth to calculate loss and update the parameters. But, while decoding,

the target language is used to generate translations by predicting the next token based on the previously generated tokens. Comparable to the input embedding done for Geez language vectoring, output embedding is applied on the Tigrigna language. It is the embedding layer that is responsible for changing the words into their corresponding tensors. Thus, the mapping of Tigrigna words into their vectored representations is performed. Positional information is also added like the source language data.

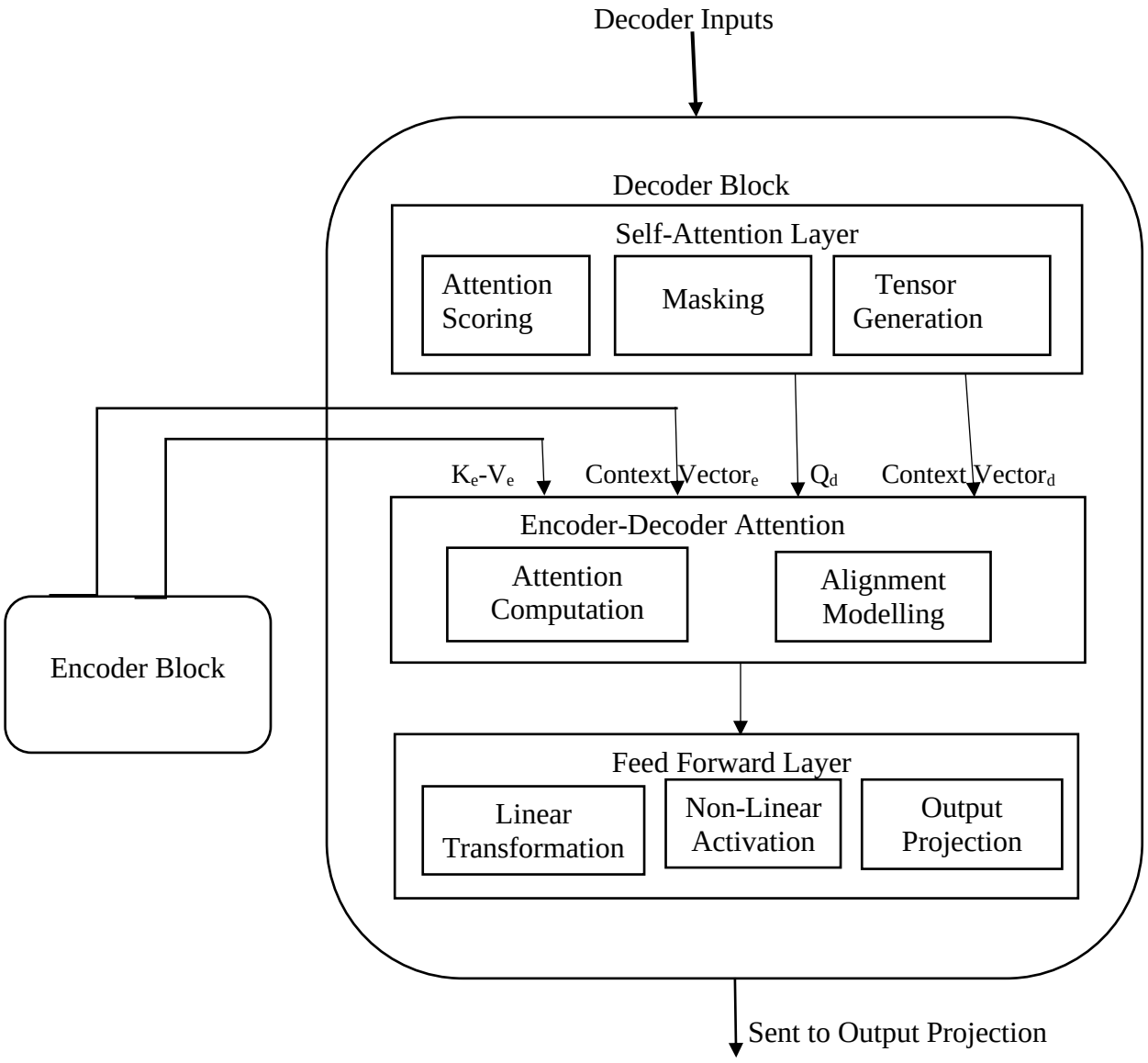


Figure 4.5 The Decoder Block

Another input to the decoder is the SOS token. The decoder network needs to be initialized with this special token. In transformer NMT models, the SOS token is typically added at the beginning

of the decoder input sequence during the decoding process. This token serves as an indicator to the model that it should start generating the translation. Its position in the decoder input sequence is usually at index zero.

Before decoding, a masking mechanism is applied to prevent the model from attending to future positions during self-attention. This ensures that each word can only attend to previous words in the sequence and hence prevent the model from cheating by looking ahead and generating tokens based on information it should not have to access it.

During the decoding process in the transformer model, the encoder-decoder attention mechanism is applied to enable the model to focus on significant parts of the input sentence. It helps the model to associate the source sentence with the target sentence and produce accurate translations. The mechanism works by computing attention weights between the decoder's query and the encoder's key-value pairs. These represent the current state of the decoder, and the output of the encoder respectively. The attention weights are computed by taking the dot product between the query and the keys, followed by scaling the scores by the square root of the dimension of the model. This step helps to make the gradients stable during training. The attention scores are then scaled and normalized using softmax function, resulting in attention weights that show the significance of each previous target token. The encoder-decoder attention layer works just like multiheaded self-attention, except it creates its queries matrix from the layer below it and takes the keys and values matrix from the output of the encoder stack. During translation it is essential to establish alignments between the words in the source and target sentences. The encoder-decoder attention mechanism helps model these alignments by assigning higher attention weights to the source words that have a stronger influence on predicting the current target word. This alignment modelling aids in capturing the dependencies between the source and target languages.

The hidden states from both self-attention and encoder-decoder attention are passed through feed-forward neural networks to capture non-linear relationships and generate more expressive representations. The task of feed-forward networks in decoding is to take the output of the encoder-decoder attention mechanism and process it further to generate the final output for the current time step. This involves passing the combined vector through a feed-forward neural network, which consists of one or more hidden layers with non-linear activation functions. The feed-forward network helps to capture complex patterns and relationships in the input data, allowing the decoder

to make more informed decisions about the next target token to generate. The output of the feed-forward network is then used to generate the next target token.

4.2.1.6 Output Projection

The final hidden states are projected onto the target vocabulary size using a linear layer followed by softmax activation. This generates a probability distribution over the target vocabulary, indicating the likelihood of each word being the next output token. In other words, the decoder's hidden representation is mapped to a probability distribution over the target vocabulary. The most probable next word in the target sequence is generated.

The decoder stack outputs a vector of floats, and these floating vectors are turned into a word in the final linear layer which is followed by a softmax layer. The Linear layer is a simple fully connected neural network that projects the vector produced by the stack of decoders into a much larger vector that it learned from its training dataset. Each cell corresponds to the score of a unique word. The softmax layer then turns those scores into probabilities. The cell with the highest probability is chosen, and the word associated with it is produced as the output for current time step.

4.2.1.7 Output Generation

During decoding, the model generates one word at a time by considering the previously generated words. The predicted word from the previous time step is fed as input to the decoder along with the encoder output and the positional encoding. This allows the model to attend to the relevant parts of the source sentence and generate the next word in a context-aware manner. By incorporating the predicted word into the decoder input sequence, the model can capture dependencies and context information while generating each word. This autoregressive approach ensures that the model generates translations sequentially, considering the previously generated words, and allows for more coherent and fluent translations. Here autoregressive generation refers to the process of generating one word at a time in a sequential manner, taking the previously generated words. This is achieved by adding the predicted word from the previous time step to the decoder input sequence during the decoding process. The generated output tokens are concatenated to form the translated sentence. These steps are repeated for each input sentence in the batch during the decoding process of transformer NMT.

Backpropagation and Gradient Descent

Backpropagation and gradient descent play a crucial role in training the transformer based NMT model. Backpropagation is a technique used to calculate the gradients of the loss function with respect to the parameters of the model, including the weight matrices. It works by propagating the error or loss from the output layer back through the layers of the model, adjusting the weights at each layer based on their contribution to the overall error. This allows the model to learn how changes in the weight matrices affect the loss and adjust accordingly.

Gradient descent is an optimization algorithm that utilizes the gradients calculated through backpropagation to update the weight matrices iteratively. It works by taking small steps in the direction opposite to the gradient, gradually minimizing the loss function. By continuously updating the weight matrices based on the gradients, gradient descent helps the model converge towards an optimal set of values that minimize the loss and improve translation performance.

In the context of transformer based NMT, backpropagation and gradient descent are used to update the weight matrices of the dense layers in the multi-head attention mechanism. These weight matrices are adjusted during training to find the values that enable effective encoding and processing of input sequences, leading to accurate translations during inference. By iteratively updating these weight matrices using backpropagation and gradient descent, the model learns to optimize its parameters and improve its translation capabilities. Adam optimizer is considered in this thesis.

Chapter 5: Experiment and Result

5.1 Introduction

This chapter presents the implementation details of the Geez to Tigrigna machine translation architecture based on transformer model. The chapter begins by describing the process of data collection, encompassing written and digital sources to ensure comprehensive language coverage. Subsequently, the corpus is prepared through preprocessing techniques to clean and normalize the data. The chapter then explores into parameter selection, where careful choices are made to optimize the performance of the transformer model. Python is employed as programming language for implementation. This chapter ends up by tabulating the results obtained.

5.2 Data Collection

It is apparent that working with machine learning heavily requires a very massive amount of training and test dataset to solve the given problem accurately. The process of data collection involves gathering enough data that will be used to train the NMT model. The goal is to collect a diverse and representative dataset that covers the target language pairs. Sample corpus is given in Appendix B. In our case a total of 10362 Geez-Tigrigna parallel sentences were manually collected (tabulated) mainly from religious books (the Holy Bible, prayer books, the battle of Saints), and from the social media. Out of the 10362 sentences, 103 Tigrigna sentences are taken from Akubazgi Gebremariam's [14] corpus and translated them to Geez by language experts. From the total sentences 9325 are used for training and the remaining 1037 for testing.

5.3 Corpus Preparation

Corpus preparation involves pre-processing the collected data to make it suitable for training the transformer based NMT model. This step typically includes removing any noise or irrelevant information from the dataset. A set of pre-processing tasks such as normalizing the text, and tokenizing sentences are implemented on the collected parallel corpus. Corpus preparation ensures that the data is in a format that can be readily used by the NMT model during training. After this the experimentation will run smoothly.

5.4 Environmental Setup

Selecting the appropriate tools and programming language are the fundamental tasks in machine translation implementation. We used Python programming language for coding as Python is very efficient in deep learning applications. Python programming language supports a set of freely available library in the deep learning algorithm. Python is a high-level, interpreted programming language that is widely used for various purposes such as web development, data analysis, artificial intelligence, and many more. It is known for its simplicity, readability, and versatility. Python has a large standard library and a vast ecosystem of packages, making it suitable for a wide range of applications. As we cannot train our system on ordinary CPU, a Graphical Processing Unit (GPU) is used.

A GPU is a specialized electronic circuit that is primarily designed to handle and accelerate graphics processing. However, due to their parallel processing capabilities, GPUs have become increasingly popular for general-purpose computing tasks, including machine learning. Compared to traditional CPUs, GPUs can perform many calculations simultaneously, resulting in significantly faster execution times for certain types of computations.

Google Collaboratory (Google Colab) is a cloud-based development environment provided by Google that allows users to write and execute Python code in their web browsers. It offers a Jupyter Notebook-like interface where users can create and share documents containing live code, equations, visualizations, and explanatory text. One of the key advantages of Google Colab is that it provides free access to powerful GPUs, enabling users to leverage the computational capabilities of GPUs without having to set up their own hardware. This makes it particularly useful for machine learning tasks that require intensive computations. The Python code, the corpus we prepared and all other necessary files that are included in the code are uploaded to Google drive. Once it is on Google drive, the corpus can be loaded by Google Colab. The Python code for the actual implementation is developed and run plenty of times by changing the hyperparameters. The parallel sentences are arranged on a notepad. All the documentation and the experiment of this thesis is executed on 11th Generation Intel Core i7 and 2.80GHz processor windows operating system computer of Installed RAM size 32.0 GB.

5.5 Parameter Selection and Experiment

Hyperparameter Optimization

Hyperparameter optimization is the process of selecting the optimal values for hyperparameters in a machine learning model and it is crucial for achieving optimal performance. These parameters are not learned directly from the data during the training process but are set before training and influence the behaviour and performance of the model. The goal of hyperparameter optimization is to find the combination of hyperparameter values that maximizes the performance or effectiveness of the machine learning model on a specific task. This can involve finding the hyperparameter values that minimize the validation error, maximize the evaluation metric, or achieve the best trade-off between different metrics. The choice of hyperparameters significantly improve the performance of the model, speed up training, and enhance its ability to generalize to unseen data. Hyperparameter optimization is typically performed through an iterative process. The hyperparameters that we set in the implementation of this thesis are embedding dimension, number of encoder and decoder layers, number of attention heads, feed-forward network hidden dimension, dropout rate, learning rate, batch size and epoch number.

The data splitting ratio is selected to be 90% by 10% as it yields a better result for this specific case. This is because it provides more training lines than the 80% by 20% splitting strategy. The batch size takes the value of 64. First the training is tried at a batch size 32. But the number of batches at a single epoch becomes 291 and it takes too much time (about 4 hours) to iterate for 100 epochs, it also fails to iterate through the whole 100 epoch during repeated trials. Most of the time it fails to train when it reaches at epoch numbers greater 70, while epoch number 64 performs well. The time it takes to train for 100 epochs reduced to 3 hours 19 minutes and 38 seconds.

The first experiment is based on attention based RNN. The embedding dimension size is set to 256 and the hidden size considered to be 512. We used Adam optimizer which is a popular optimization algorithm used in deep learning and set the learning rate is 0.001. In this model the loss approaches zero at epoch number 52 as can be seen at Appendix D. Therefore, we set the number of epochs to 50 and it takes 2 hours 57 minutes and 49 seconds to complete the training. The training loss over epochs graph for this configuration is displayed on Figure 5.2. Sample translation results performed in this model are given on Figure 5.1. The average BLEU score is 0.46.

```
[ ] translate(u'ወትእምርተሰ ኢይሁብዎ ዘእንበለ ትእምርተ ዮናስ ነቢይ')

Input Geez sentence: <start> ወትእምርተሰ ኢይሁብዎ ዘእንበለ ትእምርተ ዮናስ ነቢይ <end>
Predicted Tigrigna translation: ብዘይ ትእምርቲ ነብዪ ዮናስ ድማ ትእምርቲ ኣይውሃቦን <end>
BLEU score result is: 0.45

[ ] translate(u'ውናሁ ዘየዐቢ እምዮናስ ዝየ')

Input Geez sentence: <start> ውናሁ ዘየዐቢ እምዮናስ ዝየ <end>
Predicted Tigrigna translation: እንሆ ካብ ዮናስ ዚዓቢ እዮም <end>
BLEU score result is: 0.48

[ ] translate(u'ወእንዘ ይወፅኡ እምኢያሪኮ ተለውዎ ሰብእ ብዙኃን')

Input Geez sentence: <start> ወእንዘ ይወፅኡ እምኢያሪኮ ተለውዎ ሰብእ ብዙኃን <end>
Predicted Tigrigna translation: ካብ ያሪኮ ኺወፁ ኸለው ብዙሓት ህዝቢ ሰዐብዎ <end>
BLEU score result is: 0.45714285714285713

[ ] translate(u'ይፈትን ልበ ወኵልያተ እግዚአብሔር')

Input Geez sentence: <start> ይፈትን ልበ ወኵልያተ እግዚአብሔር <end>
Predicted Tigrigna translation: ኣምላኽን ንልብን ኩላሊትን ይምርምር <end>
BLEU score result is: 0.41379310344827586

[ ] translate(u'ወተሰጥዎ ኢዮብ ለእግዚእ  ብሔር ወይቤ')

Input Geez sentence: <start> ወተሰጥዎ ኢዮብ ለእግዚእ ብሔር ወይቤ <end>
Predicted Tigrigna translation: እዮብ ንእግዚኣብሄር ከምዚ ኢሉ መለሰሉ <end>
BLEU score result is: 0.4838709677419355

[ ] translate(u'ወእንዘ ይነግሮሙ ዘንተ በቅድመ ኵሉ ሕዝብ እገዙ ጸሐፍት ወፈሪሳውያን ያመንዝዝዎ ወይትቀየምዎ')

Input Geez sentence: <start> ወእንዘ ይነግሮሙ ዘንተ በቅድመ ኵሉ ሕዝብ እገዙ ጸሐፍት ወፈሪሳውያን ያመንዝዝዎ ወይትቀየምዎ <end>
Predicted Tigrigna translation: እዚ ምስ ተዛረበ እቶም ፃሓፍትን ፈሪሳውያንን ብፅኑዕ ኪቕየምዎ ጀመሩ <end>
BLEU score result is: 0.44

[ ] translate(u'እስመ ዚአከ ይእቲ መንግሥት ኀይል ወስብሓት ለዓለም ዓለም')

Input Geez sentence: <start> እስመ ዚአከ ይእቲ መንግሥት ኀይል ወስብሓት ለዓለም ዓለም <end>
Predicted Tigrigna translation: መንግስትን ሓይልን ክብርን ንዘለኣለም ናትካ እዩ <end>
BLEU score result is: 0.4864864864864865
```

Figure 5.1 RNN Based Translation Samples

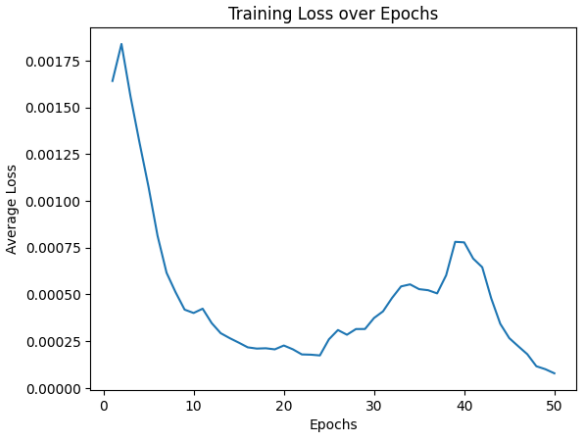


Figure 5.2 Training Loss Over Epochs Graph for the RNN Model

The second experiment is based on transformer NMT with the non-aligned corpus. First, we set the parameters as 90/10 split ratio, batch size = 64, hidden unit =1024, epoch number =25, learning rate=0.001, embedding dimension= 512, Attention head = 8, Number of layers=6, Feedforward dimension= 2048. For this combination of hyperparameters, the total training time takes 2 hour 56 minutes and 17 seconds. For our total training data, and a batch size of 64, we have 145 batches in each epoch. In this experiment the performance of our model becomes too low. Most of the sentences are translated incorrectly. Given the sentence "ወሰመዮ እግዚአብሔር ለውእቱ ጠፈር ሰማየ" the correct translation is "ኣምላኽ ከኣ ነቲ ጠፈር ሰማይ ኣውጽኣሉ". But the predicted Tigrigna translation is "ኣምላኽ ከኣ ካብ ሰማያት ኣውፅኣሉ". The words "ነቲ ጠፈር" are replaced by a single word "ከኣ" and the word "ሰማይ" by "ሰማያት".

```
[ ]  translate(u'ወእግዚአብሔር  ለውእቱ ጠፈር ሰማየ ሰመዮ')

     Input Geez sentence: <start> ወእግዚአብሔር ለውእቱ ጠፈር ሰማየ ሰመዮ <end>
     Predicted Tigrigna translation: ኣምላኽ ከኣ ካብ ሰማያት ኣውፅኣሉ <end>
     BLEU score result is: 0.3571428571428572
```

A second example for this case is "ክሥት ለእግዚአብሔር ፍኖተከ" whose correct translation is "መገድኻ ንኣምላኽ ኣማዕቀቦ" and predicted as "ብኣምላኽ ብሓጎስ ኣማዕቀቦ".

```
[ ]  translate(u'ክሥት ለእግዚአብሔር ፍኖተከ')

     Input Geez sentence: <start> ክሥት ለእግዚአብሔር ፍኖተከ <end>
     Predicted Tigrigna translation: ብኣምላኽ ብሓጐስ ኣማዕቀቦ <end>
     BLEU score result is: 0.3043478260869566
```

A third example is the sentence “ዘየዐቢ ብርሃን ከመ ይምልክ መዐልተ” which is predicted as “እቲ ዓብዩ ኣምላኽ ካብ ኣምላኽ እዩ”, while the correct translation is “እቲ ዓብዩ ብርሃን ብመዓልቲ ኺስልጥን”. Only two words are paired correctly from five words.

```
[ ]  translate(u'ዘየዐቢ ብርሃን ከመ መዐልተ ይምልክ')

     Input Geez sentence: <start> ዘየዐቢ ብርሃን ከመ መዐልተ ይምልክ <end>
     Predicted Tigrigna translation: እቲ ዓብዩ ኣምላኽ ካብ ኣምላኽ እዩ <end>
     BLEU score result is: 0.27586206896551724
```

A batch size of 32 also yields similar errors and raises the training time to 3 hours 22 minutes and 40 second. The sentence “ይፈትን ልበ ወኵልያተ እግዚኣብሔር” is predicted as “ብስም ኣምላኽ ክኾኖም እየ ይብል” instead of “ኣምላኽ ንልብን ኩላሊትን ይምርምር”.

```
[ ]  translate(u'ይፈትን ልበ ወኵልያተ እግዚኣብሔር')

     Input Geez sentence: <start> ይፈትን ልበ ወኵልያተ እግዚኣብሔር <end>
     Predicted Tigrigna translation: ብስም ኣምላኽ ክኾኖም እየ ይብል <end>
     BLEU score result is: 0.37037037037037035
```

From this we conclude that experimenting with such setting does not fit to our requirements and hence decided to train over larger epoch numbers. The next experiment we carried out is by training the system at epoch number of 50 and the other parameters remains the same. The output of translation here looks like:

```
[ ]  translate(u'እግዚኣብሔር ዚበ ምድር እስመ ኢያዝነመ')

     Input Geez sentence: <start> እግዚኣብሔር ዚበ ምድር እስመ ኢያዝነመ <end>
     Predicted Tigrigna translation: ኣምላኽ ኣብ ልዕሊ ምድሪ እዩ <end>
     BLEU score result is: 0.4
```

Though there is an improvement from training on epoch number of 25, still there is a mismatch between the input and the output. Given the sentence “እግዚኣብሔር ዚበ ምድር እስመ ኢያዝነመ”, we expect to get “ኣምላኽ ኣብ ምድሪ ገና ኣየዝነመን ነበረ”. But what happens is “ኣምላኽ ኣብ ልዕሊ ምድሪ እዩ”. The following also show the defect of this experiment.

```
     Input Geez sentence: <start> ወእምዝ ወፅአ ጳውሎስ እምአቴና ወሖረ ቆሮንቶስ <end>
     Predicted Tigrigna translation: ጳውሎስ ድማ ናብ ቤት ሔት ሰገደሎም <end>
     BLEU score result is: 0.41379310344827586
```

The transformer-based experiments discussed so far are repeated for 80% by 20% splitting strategy, but there is no improvement in the results. Under this configuration, for the input sentence "እስመ ዚአከ ይእቲ መንግሥት ኀይል ወስብሓት ለዓለም ዓለም" the predicted output is "ምስክራትካ ፅባቐን ንዘለአለም ይነብር" while the correct translation is "መንግስትን ሓይልን ክብርን ንዘለኣለም ናትካ እዩ". Similar to this we get "ኣምላኽ ነዲሩ እዩ" instead of "ሕጂ ኽትንስእ እየ ይብል ኣምላኽ" for the input sentence "ይእዜ እትነሣእ ይቤ እግዚአብሔር".

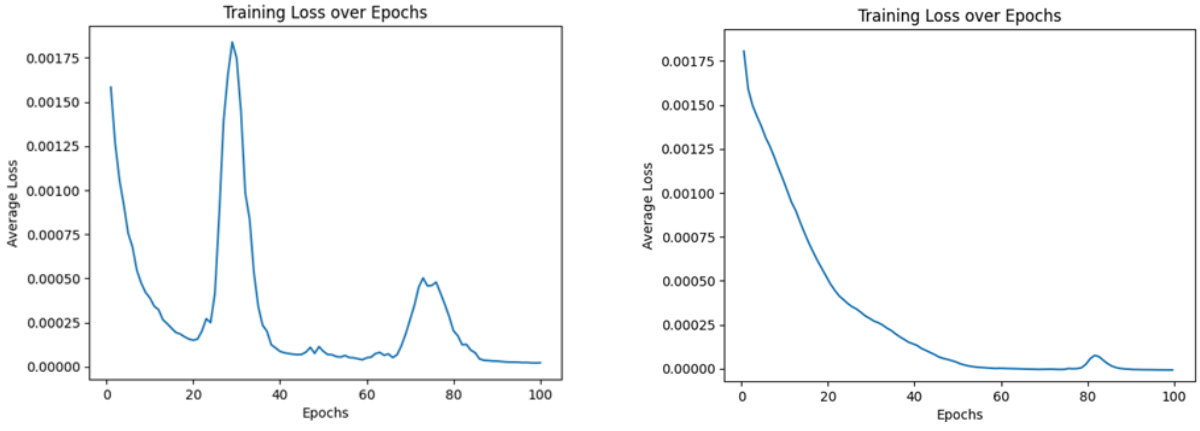
```
[ ]  translate(u'እስመ ዚአከ ይእቲ መንግሥት ኀይል ወስብሓት ለዓለም ዓለም')

     Input Geez sentence: <start> እስመ ዚአከ ይእቲ መንግሥት ኀይል ወስብሓት ለዓለም ዓለም <end>
     Predicted Tigrigna translation: ምስክራትካ ፅባቐን ንዘለአለም ይነብር <end>
     BLEU score result is: 0.4666666666666667
```

```
[ ]  translate(u'ይእዜ እትነሣእ ይቤ እግዚአብሔር')

     Input Geez sentence: <start> ይእዜ እትነሣእ ይቤ እግዚአብሔር <end>
     Predicted Tigrigna translation: ኣምላኽ ነዲሩ እዩ <end>
     BLEU score result is: 0.4444444444444444
```

The last trial for the non-aligned transformer-based experiment is on a split ratio of 90%/10%, batch size = 64, hidden unit= 1024, epoch =100, learning rate=0.001, embedding dimension=512, attention heads=8, number of layers= 4, Feedforward dimension=2048. Total time of training is 3 hours 53 minutes and 16 seconds. The training loss over epoch for the non-aligned and aligned corpus are demonstrated on Figure 5.3. the effect of alignment can also be shown by attention diagrams as depicted on Figure 5.6.



a) For Non-Aligned Corpus

b) For Aligned Corpus

Figure 5.3 Training Loss Over Epoch for Non-Aligned and Aligned Corpus

The third experiment is on a split ratio of 90%/10%, batch size = 64, hidden unit= 1024, epoch =100, learning rate=0.001, embedding dimension=512, attention heads=8, number of layers= 4, Feedforward dimension=2048 and with the aligned corpus. The total time of training is 3 hours 19 minutes and 38seconds. Sample translation outputs for this case are displayed in Figure 5.4 and Figure 5.5 for Non-aligned and aligned corpus respectively. The average BLEU scores are also registered to be 0.54174 and 0.63745 in order.

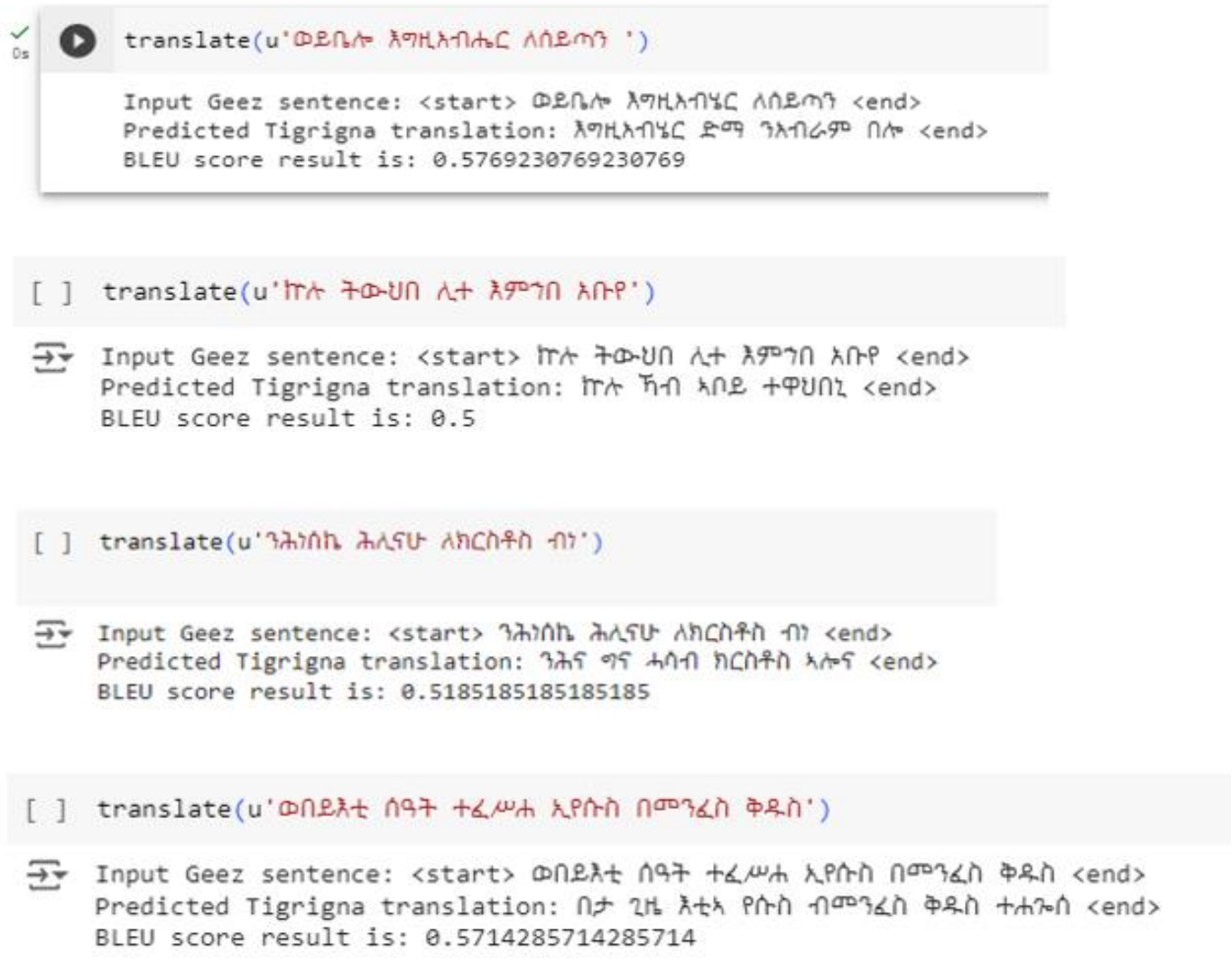


Figure 5.4 Sample Translation Outputs of Transformer Based Non-Aligned Corpus

```
[68] translate(u'ወይቤሎ እግዚአብሔር ለሰይጣን ')

Input Geez sentence: <start> ወይቤሎ እግዚአብሄር ለሰይጣን <end>
Predicted Tigrigna translation: ሽሉ እግዚአብሄር ድማ ንሰይጣን በሉ <end>
BLEU score result is: 0.6206896551724138
```

```
[ ] translate(u'ወበይእቲ ሰዓት ተፈሥሐ ኢየሱስ በመንፈስ ቅዱስ')

Input Geez sentence: <start> ወበይእቲ ሰዓት ተፈሥሐ ኢየሱስ በመንፈስ ቅዱስ <end>
Predicted Tigrigna translation: በታ ጊዜ እቲኣ የሱስ ብመንፈስ ቅዱስ ተሐጐሰ <end>
BLEU score result is: 0.666666666666666
```

```
[ ] translate(u'ትእምርተ ተንሥሥ')

Input Geez sentence: <start> ትእምርተ ተንሥሥ <end>
Predicted Tigrigna translation: ትእምርቲ ይደሊ <end>
BLEU score result is: 0.625
```

Figure 5.5 Sample Translation Outputs of Transformer Based Aligned Corpus

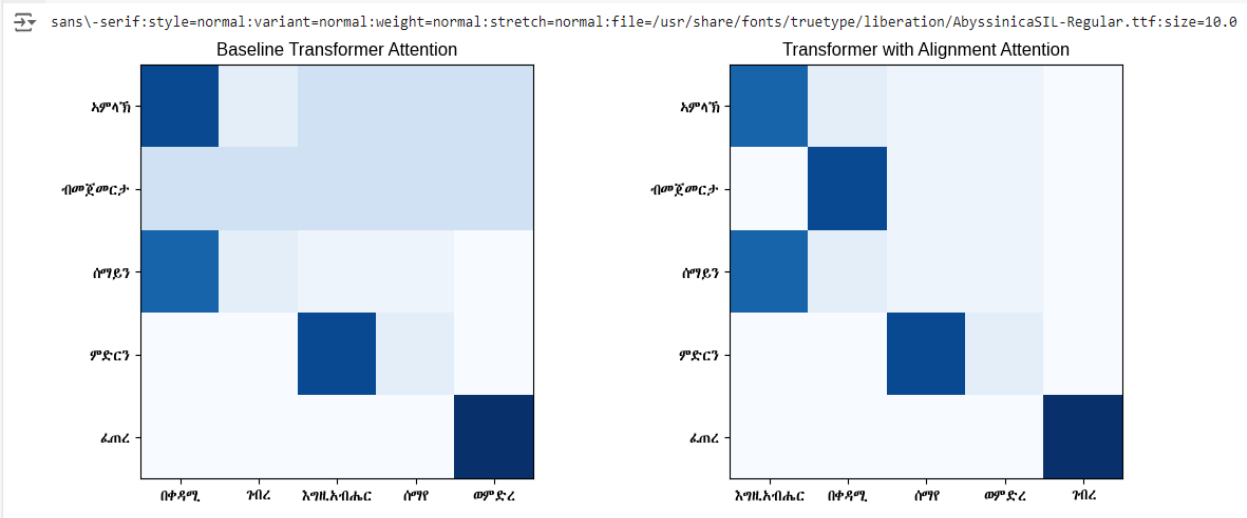


Figure 5.6 Attention Diagrams for the Aligned and Non-aligned Corpus

5.6 Evaluation Metrics

Evaluating the trained model performance by a BLEU score measuring metrics is performed. This step helps us to assess the quality of translations produced by our model. BLEU is a widely used automatic evaluation metric for machine translation quality. It measures the similarity between the machine-generated translations and reference translations.

5.7 Results and Discussion

After designing the transformer based Geez to Tigrigna neural machine translation and preparing the corpus, the next task is experimenting with different NMT approaches. The whole corpus is divided into training and testing sets for experimentation purpose based on 90% by 10% ratio respectively. The parameters are selected by trial and errors on different combinations of hyperparameters beginning with the values set as default on various NMT applications. This task of repeated experiment was a little bit tedious and bulky, but interesting. So, the parameters listed as default on different research papers are considered as an initial training value for this thesis. Mainly the default values from the original work of transformers by Vaswani et.al and then by repeatedly experimenting for this specific case the hyperparameters are set.

We conducted three distinct experiments to assess the effectiveness of different NMT strategies. In our first experiment, we implemented a standard attention-based NMT model. This configuration achieved a BLEU score of 46%. While this result provided a baseline for our accomplishments, it illustrated the potential for significant enhancement through more advanced methods. The second experiment utilized a transformer architecture without alignment in the training corpus. This configuration yielded a BLEU score of 54.174%, indicating a notable improvement over the initial attention-based model. This result highlighted the advantages of utilizing the self-attention mechanism of the transformer architecture, which allows the model to better capture contextual relationships within the input sequences. Our final experiment employed a transformer model with an aligned corpus, resulting in a BLEU score of 63.745%. This approach demonstrated the highest performance among the three configurations and represents a significant milestone in our thesis. The alignment of training data fosters more accurate translation due to the enhanced contextual learning facilitated by the transformer architecture.

When we compare the results of this thesis with what we have seen in the literature review section there is an improvement in the result in almost all of the assessed works. But the work by Biruk Abel [76] which is on developing a Hybrid Geez to Amharic Machine Translation system using serial coupling of rule-based Geez language word reordering followed by a standard SMT system is an exception. The BLEU score of the study was 76%. This result is better than what we have done currently. But the deficiencies of the rule-based approach discussed earlier are evidence not to use that technique for the Geez to Tigrigna translation.

The results from these experiments indicate that the transformer-based NMT model outperformed the attention-based model, particularly when utilizing an aligned corpus. The current translation performance is encouraging, suggesting that our approach matches or surpasses existing benchmarks in translation quality for the Geez to Tigrigna language pair. However, while the achieved BLEU scores signify a considerable advancement in translation capabilities, there remains room for further improvement. Factors such as data quality, the diversity of translation examples, and detailed cultural contexts still warrant an attention. Future work could involve refining the dataset or exploring additional techniques such as fine-tuning with transfer learning or incorporating linguistic features specific to the Tigrigna language.

In conclusion, this thesis not only demonstrates the effectiveness of transformer-based architectures in translating Geez to Tigrigna but also sets the stage for continued exploration in enhancing translation accuracy and fluency. The results achieved herein contribute valuable insights to the field of neural machine translation.

Chapter 6: Conclusion and Future Works

6.1 Conclusion

The aim of this thesis work is to develop Geez to Tigrigna neural machine translation with a transformer based neural machine translation approach. The documentation starts by highlighting the need of language translation and discussing the different approaches to machine translation. Then, it explains the structures of both Geez and Tigrigna languages. Various earlier works on machine translation using different techniques are also revised. As can be observed from the literatures, neural machine translation outperforms most of the other techniques. Neural machine translation has become the dominant approach to machine translation in both research and practice. After that, the design and implementation of the Geez to Tigrigna NMT is presented.

The architecture of the proposed neural machine translation system is based on the transformer model, which has shown state-of-the-art performance on many language pairs. To address the challenges of translating between low-resource languages like Geez and Tigrigna, an alignment-based approach is integrated into the standard transformer architecture. This alignment mechanism aims to better capture the relationships between source and target language elements during the translation process. The word-level alignments between the parallel sentences are done manually.

A parallel corpus for the implementation of the translation is prepared from various resources, critically the Holy Bible. Different pre-processing techniques are applied on the collected data to make the dataset ready for machine translation. This set of parallel sentences is divided into training and testing dataset. Python programming language is selected because of its robustness in machine translation tasks. Three different experiments are conducted, and the transformer based NMT model for Geez to Tigrigna neural machine translation with aligned corpus has shown promising results with a BLEU score of 0.63745. The model has demonstrated the potential to accurately translate between the two languages, and with further optimization and training, it could potentially achieve even higher accuracy.

6.2 Contributions

- The corpus we prepared is the primary contribution.
- As this research work is the first machine translation from Geez to Tigrigna, it can guide different researchers as a baseline for their further study.
- This thesis work paves a way for Ethiopian researchers to conduct their works on Ethiopic language pairs.
- Incorporation of data alignment techniques within the encoder section of the transformer architecture.
- As ensuring high-quality alignment can significantly enhance the performance of NMT models, our aligned corpus is another contribution of this work.

6.3 Future Works

Based on the experimental results, it is recommended to continue refining and optimizing the transformer based NMT model for Geez to Tigrigna translation. This could involve increasing the size of the training dataset, fine-tuning the model hyperparameters, and implementing advanced techniques. Additionally, incorporating linguistic knowledge and exploring ways to integrate morphological, syntactic, or semantic information into the transformer-based NMT model to improve its understanding of the source and target languages can be done. Further research into language-specific traces and linguistic patterns could also help improve the translation accuracy. With these improvements, it is estimated that the model could achieve even better results and be a valuable tool for translating between Geez and Tigrigna. One can extend the work to a bidirectional neural machine translation.

References

[1] Gebremeskel Hagos Gebremedhin, Abera Asefa Mebrahtu, "Linguistic Evolution of Ethiopic Language: A Comparative Discussion," *International Journal of Interdisciplinary Research and Innovations,* vol. 8, no. 1, pp. 1-9, 2020.

[2] Joseph O'Connor, John Seymour, Introducing NLP Neuro-Linguistic Programming, San Francisco, California: The Aquarian Press.

[3] Frankenfield, Jake, "Investopedia," 2 September 2021. [Online]. Available: www.investopedia.com. [Accessed 23 October 2021].

[4] "Tutorials Point," 2019. [Online]. Available: www.tutorialspoint.com. [Accessed 23 October 2021].

[5] Cheragui, Mohamed Amine, "Theoretical Overview of Machine translation," in *Proceedings ICWIT*, African University, Adrar, Algeria, 2012.

[6] Jiajun Zhang, Chengqing Zong, "Neural Machine Translation: Challenges, Progress and Future," Beijing, China, 2020.

[7] Denny Britz, Anna Goldie, Minh-Thang Luong, Quoc Le, "Massive Exploration of Neural Machine Translation Architectures," Google Brain, 2017.

[8] G. Dires, "Language policy of Ethiopia," Centeral Connecticut State University, 2019.

[9] A. Bell, "Britannica.com," Encyclopedia Britannica, [Online]. Available: www.britannica.com. [Accessed 26 10 2021].

[10] R. Tadese, አንድሮሜዳ - Andromeda: የጥንት ኢትዮጵያውያን የሕዋ ምርምር ከዘመናዊው ሳይንስ አንጻር, Kindle Edition, May 19th 2020.

[11] Schirripa, Pino, "The Medical System in Tigray," in *Competing Orders of Medical Care in Ethiopia: From Traditional Healers to Pharmaceutical Companies*, Rowman \& Littlefield, 2012, p. 51.

[12] G. Worku, "Ge'ez-Amharic Machine Translation using Deep Leaning," Bahir Dar Institute of Technology, Bahir Dar, Ethiopia, 2021.

[13] S. Tefaye, "A Hybrid Approach for Machine Translation from Geez to Amharic language," St. Mary's University, Addis Ababa, 2020.

[14] A. Gebremariam, "Amharic-to-Tigrigna Machine Translation Using Hybrid Approach," Addis Ababa University, Addis Ababa, Ethiopia, 2017.

[15] Z. Adhana, መርሆ - ስዋስው ዘልሳነ ግእዝ, Addis Ababa: Holy Trinity Theological College, 2008.

[16] Z. Adhana, ልሳናተ ሴም ፥ ግእዝ፣ ትግርኛ፣ አማርኛ (ንጽጽራዊ መዝገበ ቃላት), Addis Ababa: Holy Trinity Theological College, 2009.

[17] D. Keleb, The Rivival of Geez (ትንሳኤ ግእዝ), Addis Ababa: Mahibere Kidusan, 2010.

[18] D. Girma, "Ge'ez Literature and Medieval Ethiopian Hagiographies in the course of Ethiopian Literature," *International Journal of English Literature and Culture,* vol. 7, no. 5, pp. 121-129, 2019.

[19] W. Laura, "wendybelcher.com," [Online]. Available: https://wendybelcher.com/. [Accessed 9 September 2023].

[20] D. Keleb, "The Re-creation of Geez," [Online]. Available: https://www.scribd.com/document/484378393/Geez#. [Accessed 10 January 2022].

[21] Jennifer Abdella, Michael Anderson, Adam Augustyn, Jonathan Daly, "Encyclopeadia," 3 October 2021. [Online]. Available: https://www.britannica.com/topic/Ethiopic-alphabet. [Accessed 13 February 2022].

[22] E. Baryagabir, "Eyugeez Media," 19 July 2022 . [Online]. Available: https://www.youtube.com/watch?v=rTPizBeJ1U8. [Accessed 4 october 2022].

[23] H. Zefer, "MEMHĀRA LESĀNA GE'EZ, ምምሃረ ልሳነ ግዕዝ," University of Washington, Seattle, 2012.

[24] "Ethiopian Argument," 15 May 2014. [Online]. Available: http://zethio.blogspot.com/2014/05/tigrinya-or-tigrigna-alphabet.html. [Accessed 13 June 2021].

[25] A. Negash, "The Origin and Development of Tigrinya Language Publications," Santa Clara University, California, 2016.

[26] Yemane Tedla, Kazuhide Yamamoto, "Morphological Segmentation ith LSTM Neural Networks for Tigrinya," *International Journal on Natural Language Computing (IJNLC),* vol. 7, no. 2, pp. 29-44, April 2018.

[27] J. Mason, ስዋሶው ትግርኛ Tigrinya Grammar, New Jersy: the Red Sea Press, 1996.

[28] N. Amlesom, "Tigrinya Applicatives in Lexical-Functional Grammar," University of Bergen, Norway, 2011.

[29] Shuoheng Yang, Yuxin Wang, Xiaowen Chu, "A Survey of Deep Learning Techniques for Neural Machine Translation," Hong Kong Baptist University, Hong Kong, China, 2020.

[30] Quinn Lanners, Dr. Thomas Laurent, "Neural Machine Translation," Loyola Marymount University, Los Angeles, 2019.

[31] Sonali Sharma, Manoj Diwakar, Prabhishek Singh, Vijendra Singh, Seifedine Kadry, Jungeun Kim, "Machine Translation Systems Based on Classical-Statistical-Deep-Learning Approaches," *Electronics,* vol. 12, no. 7, p. 29, 2023.

[32] Y. Kim, "Neural Machine Translation for Low-Resource Scenarios," Seoul, 2022.

[33] Daniel Jurafsky, James H. Martin, "RNNs and LSTMs," in *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Colorado, University of Colorado at Boulder, 2022, pp. 185-197.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin, "Attention Is All You Need," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, USA, 2017.

[35] J. Alammar, "The Illustrated Transformer – Visualizing machine learning one concept at a time," 2020. [Online]. Available: https://jalammar.github.io/illustrated-transformer/. [Accessed 20 April 2023].

[36] S. Srivastava, "Machine Translation (Encoder-Decoder Model)," 31 October 2019 . [Online]. Available: https://medium.com/analytics-vidhya/machine-translation-encoder-decoder-model 7e4867377161. [Accessed 20 February 2022].

[37] Rajesh Arumugam, Rajalingappaa Shanmugamani, Hands-On Natural Language Processing with Python A practical guide to applying deep learning architectures to your NLP applications, Birmingham, Mumbai: Packt Publishing, 2018.

[38] Dzmitry Bahdanau, KyungHyun Cho, Yoshua Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *ICLR*, Germany, 2015.

[39] H. Xu, "Transformer-Based NMT: Modeling, Training and Implementation," Saarland University, Henan, China, 2021.

[40] Okpor, "Machine Translation Approaches: Issues and Challenges," *IJCSI International Journal of Computer Science Issues,* vol. 11, no. 5, 2014.

[41] Vanlalmuansangi Khenglawt, Lalṭanpuia, "Machine translation and its approaches," *Advances in Engineering Research,* vol. 178, pp. 141-145, 2018.

[42] Solomon Teferra, Michael Melese, Martha Yifiru, Million Meshesha, Solomon Atinafu, Wondwossen Mulugeta, Yaregal Assabie, Hafte Abera, Biniyam Ephrem, Tewodros Abebe, Wondimagegnhue Tsegaye, Amanuel Lemma, Tsegaye Andargie, Seifedin Shifaw, "Parallel Corpora for bi-lingual English-Ethiopian Languages:Statistical Machine Translation," in *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA, 2018.

[43] Santanu Pal, Sudip Kumar Naskar, Josef van Genabith, "English–German Hybrid Machine Translation System," in *Proceedings of the Tenth Workshop on Statistical Machine Translation*, Lisboa, Portugal, 2015.

[44] Sneha Tripathi, Juran Krishna Sarkhel, "Approaches to machine translation," *Annals of Library and Information Studies,* vol. 57, pp. 388-393, 2011.

[45] Eassa Ali Mohammed Ali, Ameen Ali Mohammed Al-Gamal, "Corpus-Based Machine Translation," *International Journal of Research and Analytical Reviews,* vol. 8, no. 3, p. 4, 2021.

[46] Sindhu, Sagar, Rajashekar Murthy, "Survey on Machine Translation and its Approaches," *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 3, no. 6, pp. 7317-7320, June 2014.

[47] Rajesh Arumugam, Rajalingappaa Shanmugamani, Hands-On Natural Language Processing with Python, Birmngham - Mumbai: Packt Publishing Ltd, 2018.

[48] Sergey Edunov, Myle Ott, Michael Auli, David Grangier, "Understanding Back-Translation at Scale," New York, 2018.

[49] Farhad Mortezapour Shiri, Thinagaran, Norwati Mustapha, Raihani Mohamed, "A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU," University Putra Malaysia, Malaysia, 2023.

[50] Anirudha Ghosh, Abu Sufian, Farhana Sultana, Amlan Chakrabarti, Debashis De, "Fundamental Concepts of Convolutional Neural Network," University of Gour Banga, India, January 2020.

[51] Zhixing Tana, Shuo Wanga, Zonghan Yanga, Gang Chena, Xuancheng Huanga, Maosong Suna, Yang Liua, "Neural Machine Translation: A Review of Methods, Resources, and Tools," Beijing Academy of Artificial Intelligence, Beijing, China, 2020.

[52] T. Nguyen, "Machine Translation with Transformers," University of Stuttgart, Germany, 2019.

[53] Hongyi Cui, Shohei Iida, Po-Hsuan Hung, Takehito Utsuro, Masaaki Nagata, "Mixed Multi-Head Self-Attention for Neural Machine Translation," in *Proceedings of the 3rd Workshop on Neural Generation and Translation (*, Hong Kong, China, November 4, 2019.

[54] D. Crater, "Hebrew Transformed: Machine Translation of Hebrew Using the Transformer Architecture," Harvard University, Cambridge, Massachusetts, USA, March, 2022.

[55] Aditya Mandke, Onkar Litake, and Dipali Kadam, "Analyzing Architectures for Neural Machine Translation Using Low Computational Resources," SCTR's Pune Institute of Computer Technology, Maharashtra, India, 2020.

[56] Stephane Clinchant, Kweon Woo Jung, Vassilina Nikoulina, "On the use of BERT for Neural Machine Translation," in *Proceedings of the 3rd Workshop on Neural Generation and Translation*, Hong Kong, China, 2019.

[57] Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, Jinpeng Dong, "BERT Enhanced Neural Machine Translation and Sequence Tagging Model for Chinese Grammatical Error Diagnosis," in *Proceedings of the 6th Workshop on Natural Language Processing*, Suzhou, China, December 4, 2020.

[58] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," Google AI Language, 2019.

[59] T. Saha, 31 October 2021. [Online]. Available: https://tamoghnasaha-22.medium.com/transformers-illustrated-5c9205a6c70f. [Accessed 20 April 2023].

[60] Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, Kentaro Inui, "Incorporating Residual and Normalization Layers into Analysis of Masked Language Models," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Punta Cana, Dominican Republic, 2021.

[61] D. Soydaner, "Attention Mechanism in Neural Networks: Where it Comes and Where it Goes," Department of Brain and Cognition, University of Leuven (KU Leuven), Belgium, 2021.

[62] Bonnie Dorr, Matt Snover, Nitin Madnani, Machine Translation Evaluation, Computer Science, Linguistics, 2010.

[63] Aaron Li-Feng Han, Derek F. Wong , Lidia S. Chao, "Machine Translation Evaluation: A Survey," *Natural Language Processing and Machine Translation,* p. 17, 04 March 2018.

[64] Matthew Snover, Bonnie Dorr, Richard Schwartz, John Makhoul, Linnea Micciulla, Ralph Weischedel, A Study of Translation Error Rate with Targeted Human Annotation, University of Maryland, 2015.

[65] Seungjun Lee, Jungseob Lee, Hyeonseok Moon , Chanjun Park , Jaehyung Seo, Sugyeong Eo,Seonmin Koo, Heuiseok Lim, "A Survey on Evaluation Metrics for Machine Translation," *Multidisciplinary Digital Publishing Institute,* no. 1, p. 22, 17 January 2022.

[66] Agbeyangi, Abayomi, Eludiora Safiriyu, Adenekan, Olujide, "English to Yorùbá Machine Translation System using Rule-Based Approach," *Journal of Multidisciplinary Engineering Science and Technology (JMEST),* vol. 2, no. 8, pp. 2275-2280, August - 2015.

[67] Vishal Goyal, Gurpreet Singhlehal, "Evaluation of Hindi to Punjabi Machine Translation System," *IJCSI International Journal of Computer Science Issues,* vol. 4, no. 1, pp. 36-39, 2009.

[68] H. Mekonnen, "Amharic-Awngi Machine Translation: An Experiment Using Statistical Approach," *International Journal of Computer Sciences and Engineering,* vol. 7, no. 8, pp. 2347-2693, Aug 2019 .

[69] Million Meshesha, Yitayew Solomon, "English-Afaan Oromo Statistical Machine Translation," *International Journal of Computational Linguistic (IJCL),* vol. 9, no. 1, 2018.

[70] A. Birhanu, "Bi-Directional English-Afan Oromo Machine Translation Using Convolutional Neural Network," Addis Ababa University, Addis Ababa, October 14, 2019.

[71] W. Wogaso, "Attention based Amharic-to-Wolaita Neural Machine Translation," Addis Ababa University, Addis Ababa, 2020.

[72] Ibrahim Gashaw, H L Shashirekha, "Amharic-Arabic Neural Machine Translation," 2019.

[73] Adebimpe Esan, John Oladosu, Christopher Oyeleye, Ibrahim Adeyanju, Olatayo Olaniyan, Nnamdi Okomba, Bolaji Omodunbi, Opeyemi Adanigbo, "Development of a Recurrent Neural Network Model for English to Yoruba Machine Translation," *International Journal of Advanced Computer Science and Applications,* vol. 11, no. 5, pp. 602-609, 2020.

[74] Andargachew Mekonnen, Andreas Nürnberger, Tesfaye Bayu, "Neural Machine Translation for Amharic-English Translation," *Science and Technology Publications,* vol. 1, pp. 526-532, 2021.

[75] Thien Nguyen ,Lam Nguyen, Phuoc Tran,Huu Nguyen, "Improving Transformer-Based Neural Machine Translation with Prior Alignments," in *Science and Technology Publications*, Ho Chi Minh, 2021.

[76] B. Abel, "Geez to Amharic Machine Translation," Addis Ababa University, Addis Ababa, Ethiopia, 2018.

[77] J. Daba, "Bidirectional English – Afaan Oromo Machine Translation Using Hybrid Approach," Addis Ababa university, Addis Ababa, 2013.

[78] Abhaya Agarwal, Alon Lavie, "Evaluation Metrics for High-Correlation with Human Rankings of Machine Translation Output," in *Proceedings of the Third Workshop on Statistical Machine Translation, Association for Computational Linguistics*, Columbus, Ohio, USA, 2008.

[79] "Attention Mechanism in Neural Networks: Where it Comes and Where it Goes," University of Leuven (KU Leuven), Leuven, Belgium, 2022.

Appendix

Appendix A: Geez Alphabets (*ግዕዝ ፊደል*)

	Ge'ez ä	*Ka'eb* u	*Salis* ï	*Rab'e* a	*Hamis* é	*Sadis* i	*Sab'e* o
h	ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
l	ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
ḥ	ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
m	መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
s	ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
r	ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
s	ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
q	ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
b	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
t	ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ
h	ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
n	ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
a	አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ

k	ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
w	ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ
ạ	ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ
z	ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ
y	የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ
d	ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
g	ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ
ṭ	ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
p	ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ
ts	ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ
ts	ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
f	ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
p	ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ

Appendix B: Sample Corpus

Non-aligned Corpus

ግዕዝ	ትግርኛ
በቀዳሚ ፡ ገብረ ፡ እግዚአብሔር ፡ ሰማየ ፡ ወምድረ ።	ኣምላኽ ብመጀመርታ ሰማይን ምድርን ፈጠረ።
ወምድርሰ ፡ ኢታስተርኢ ፡ ወኢኮነት ፡ ድሉተ ፡ ወጽልመት ፡ መልዕልተ ፡ ቀላይ።	ምድሪ ድማ በረኻን ጥራያን ነበረት፡ ጸልማት ከኣ ኣብ ልዕሊ መዓሙቕ ነበረ።
ወመንፈሰ ፡ እግዚአብሔር ፡ ይጼልል ፡ መልዕልተ ፡ ማይ ።	መንፈስ ኣምላኽ ድማ ኣብ ልዕሊ ማያት ይዝምቢ ነበረ።
ወይቤ ፡ እግዚአብሔር ፡ ለይኩን ፡ ብርሃን ።	ኣምላኽ ከኣ ብርሃን ይኹን፡ በለ።
ወኮነ ፡ ብርሃን ።	ብርሃን ድማ ኾነ።
ወርእዮ ፡ እግዚአብሔር ፡ ለብርሃን ፡ ከመ ፡ ሠናይ።	ኣምላኽ ድማ እቲ ብርሃን ጽቡቕ ከም ዝኾነ ረኣየ።
ወፈለጠ ፡ እግዚአብሔር ፡ ማእከለ ፡ ብርሃን ፡ ወማእከለ ፡ ጽልመት ።	ኣምላኽ ከኣ ነቲ ብርሃን ካብ ጸልማት ፈለዮ።
ወሰመዮ ፡ እግዚአብሔር ፡ ለብርሃን ፡ ዕለተ።	ኣምላኽ ነቲ ብርሃን መዓልቲ ኣውጽኣሉ።
ሰመዮ ለጽልመት ፡ ሌሊተ ፡	ነቲ ጸልማት ለይቲ ኣውጽኣሉ።
ወኮነ ፡ ሌሊተ ፡ ወጸብሐ ፡ ወኮነ ፡ መዓልተ ፡ አሃደ ።	ምሸት ኾነ ብጊሓትውን ኾነ፡ ሓንቲ መዓልቲ።
ይቤ ፡ እግዚአብሔር ፡ ለይኩን ፡ ጠፈር ፡ ማእከለ ፡ ማይ ፡ ከመ ፡ ይፍልጥ ፡ ማእከለ ፡ ማይ።	ኣምላኽ ድማ ንማያት ካብ ማያት ዚፈሊ ጠፈር ኣብ መንጎ ማያት ይኹን፡ በለ።
ወወኮነ ፡ ከማሁ::	ከምኡ ድማ ኮነ።
ወሰመዮ ፡ እግዚአብሔር ፡ ለውእቱ ፡ ጠፈር ፡ ሰማየ።	ኣምላኽ ከኣ ነቲ ጠፈር ሰማይ ኣውጽኣሉ።
ወርእየ ፡ እግዚአብሔር ፡ ከመ ፡ ሠናይ ፡	ኣምላኽ ድማ ጽቡቕ ከም ዝኾነ ረኣየ።
ወኮነ ፡ ሌሊተ ፡ ወጸብሐ ፡ ወኮነ ፡ ካልእተ ፡ ዕለተ ።	ምሸት ኾነ ብጊሓትውን ኾነ፡ ካልኣይቲ መዓልቲ።

ወይቤ ፡ እግዚአብሔር ፡ ለይትጋባእ ፡ ማይ ፡ ዘመትሕተ ፡ ሰማይ ፡ ውስተ ፡ አሐዱ ፡ መካን፡ወያስተርኢ ፡ የብስ ፡	ኣምላኽ ድማ፥ እቲ ንቐጽ ምእንቲ ኺርኤስ፡ እቲ ኣብ ትሕቲ ሰማይ ዘሎ ማያት ናብ ሓንቲ ቦታ ይተኣኻብ፡ በለ።
ወሰመዮ ፡ እግዚአብሔር ፡ ለየብስ ፡ ምድረ።	ኣምላኽ ከኣ ነቲ ንቐጽ ምድሪ ኣውጽኣሉ።
ወለምእላዲሁ ፡ ለማይ ፡ ሰመዮ ፡ ባሕረ።	ነቲ እኩብ ማያት ድማ ባሕሪ ኣውጽኣሉ።
ወኮነ ፡ ሌሊተ ፡ ወጸብሐ ፡ ወኮነ ፡ ሣልስተ ፡ ዕለተ ።	ምሸት ኰነ ብጊሓትውን ኰነ፡ ሳልሰይቲ መዓልቲ።
ወይቤ ፡ እግዚአብሔር ፡ ይኩኑ ፡ ብርሃናት ፡ ውስተ ፡ ጠፈረ ፡ ሰማይ ፡ ከመ ፡ ያብርሁ ፡ ዲበ ፡ ምድር ፡	ኣምላኽ ድማ ንመዓልቲ ኻብ ለይቲ ዚፈልዩ ብርሃናት ኣብ ጠፈር ሰማይ ይኹን በለ።
ወይፍልጡ ፡ ማእከለ ፡ ዕለት ፡ ወማእከለ ፡ ሌሊት ፡ ወይኩኑ ፡ ለተኣምር ፡ ወለዘመን ፡ ወለመዋዕል ፡ ወለዓመታት ።	ንዘበናትን ንመዓልታትን ዓመታትን ከኣ ንመፈለጥታ ይኹን።
ወይኩኑ ፡ ለአብርሆ ፡ ውስተ ፡ ጠፈረ ፡ ሰማይ ፡ ከመ ፡ ያብርሁ ፡ ዲበ ፡ ምድር ።	ኣብ ምድሪ ንምብራህ ኣብ ጠፈር ሰማይ ብርሃናት ይኹን፡ በለ።
ወገብረ ፡ እግዚአብሔር ፡ ብርሃናተ ፡ ክልኤተ ፡ ዐበይተ።	ኣምላኽ ከኣ ኽልተ ዓበይቲ ብርሃናት ገበረ።
ዘየዐቢ ፡ ብርሃን ፡ ከመ ፡ ይምልክ ፡ መዐልተ።	እቲ ዓብዪ ብርሃን ብመዓልቲ ኺስልጥን።
ወዘይንእስ ፡ ብርሃን ፡ ከመ ፡ ይምልክ ፡ ሌሊተ ፡ ምስለ ፡ ከዋክብቲሁ ።	እቲ ንእሽቶ ብርሃን ድማ ብለይቲ ኺስልጥን፡ ከዋኽብቲውን ገበረ።
ወተፈጸመ ፡ ሰማይ ፡ ወምድር ።	ከምኡ ሰማይን ምድርን ተፈጸሙ።
ወኮሎ ፡ ዓለመ ፡ ፈጸመ ፡ እግዚአብሔር ፡ ገቢረ ፡ ግብሮ።	ኣምላኽ ከኣ ነቲ ዝገበሮ ግብሩ ፈጸሞ።
ወአዕረፈ ፡ እግዚአብሔር ፡ በሳብዕት ፡ ዕለት ፡ እምኵሉ ፡ ግብሩ ።	ብሳብዓይቲ መዓልቲ ድማ ኻብቲ ዝገበሮ ዅሉ ግብሩ ዐረፈ።
ወባረካ ፡ እግዚአብሔር ፡ ለዕለት ፡ ሳብዕት ፡ ወቀደሳእስመ ፡ ባቲ ፡ አዕረፈ ፡ እምኵሉ ፡ ግብሩ ፡ ዘአኀዘ ፡ ይግበር ፡ እግዚአብሔር ።	ኣምላኽ ከኣ ኻብቲ ዝፈጠሮን ዝገበሮን ኵሉ ግብሩ ብእኣ ስለ ዝዐረፈ፡ ነታ ሳብዐይቲ መዓልቲ ባረኻን ቀደሳን።

Aligned Corpus

እግዚአብሔር በቀዳሚ ሰማየ ወምድረ ገብረ።	ኣምላኽ ብመጀመርታ ሰማይን ምድርን ፈጠረ።
ወምድርሰ ፡ ኢታስተርኢ ፡ ወኢኮነት ፡ ድሉተ ፡ ወጽልመት ፡ መልዕልተ ፡ ቀላይ።	ምድሪ ድማ በረኻን ጥራያን ነበረት፡ ጸልማት ከኣ ኣብ ልዕሊ መዓሙቕ ነበረ።
ወመንፈስ እግዚአብሔር መልዕልተ ማይ ይጼልል።	መንፈስ ኣምላኽ ድማ ኣብ ልዕሊ ማያት ይዝምቢ ነበረ።
ወእግዚአብሔር ብርሃን ለይኩን ይቤ።	ኣምላኽ ከኣ ብርሃን ይኹን፡ በለ።
ወብርሃነ ኮነ።	ብርሃን ድማ ኾነ።
ወእግዚአብሔር ለብርሃን ከመ ሠናይ ርእዮ።	ኣምላኽ ድማ እቲ ብርሃን ጽቡቕ ከም ዝኾነ ረኣየ።
ወእግዚአብሔር ማእከለ ብርሃን ወማእከለ ጽልመት ፈለጠ ።	ኣምላኽ ከኣ ነቲ ብርሃን ካብ ጸልማት ፈለዮ።
ወእግዚአብሔር ለብርሃን ዕለተ ሰመዮ።	ኣምላኽ ነቲ ብርሃን መዓልቲ ኣውጽኣሉ።
ለጽልመት ሌሊተ ሰመዮ።	ነቲ ጸልማት ለይቲ ኣውጽኣሉ።
ወሌሊተ ኮነ ወጸብሐ ወኮነ አሃደ መዓልተ።	ምሸት ኰነ ብጊሓትውን ኰነ፡ ሓንቲ መዓልቲ።
ወእግዚአብሔር ማእከለ ማይ ከመ ይፍልጥ ማእከለ ማይ ለይኩን ጠፈር ይቤ።	ኣምላኽ ድማ ንማያት ካብ ማያት ዚፈሊ ጠፈር ኣብ መንነ ማያት ይኹን፡ በለ።
ወከማሁ ኮነ::	ከምኡ ድማ ኮነ።
ወእግዚአብሔር ለውእቱ ጠፈር ሰማየ ሰመዮ።	ኣምላኽ ከኣ ነቲ ጠፈር ሰማይ ኣውጽኣሉ።
ወእግዚአብሔር ከመ ሠናይ ርእየ ።	ኣምላኽ ድማ ጽቡቕ ከም ዝኾነ ረኣየ።
ወሌሊተ ኮነ ወጸብሐ ወኮነ ካልእተ ዕለተ።	ምሸት ኰነ ብጊሓትውን ኰነ፡ ካልኣይቲ መዓልቲ።
ወእግዚአብሔር የብስ ወያስተርኢ ዘመትሕተ ሰማይ ማይ ውስተ አሐዱ መካን ለይትጋባእ ይቤ።	ኣምላኽ ድማ፥ እቲ ንቐጽ ምእንቲ ኺርኤስ፡ እቲ ኣብ ትሕቲ ሰማይ ዘሎ ማያት ናብ ሓንቲ ቦታ ይተኣኻብ፡ በለ።
ወእግዚአብሔር ለየብስ ምድረ ሰመዮ።	ኣምላኽ ከኣ ነቲ ንቐጽ ምድሪ ኣውጽኣሉ።
ወለምእላዲሁ ለማይ ባሕረ ሰመዮ።	ነቲ እኩብ ማያት ድማ ባሕሪ ኣውጽኣሉ።
ወሌሊተ ኮነ ወጸብሐ ወኮነ ሣልስተ ዕለተ ።	ምሸት ኰነ ብጊሓትውን ኰነ፡ ሳልሰይቲ መዓልቲ።

ወእግዚአብሔር ከመ ያብርሁ ዲበ ምድር ብርሃናት ውስተ ጠፈረ ሰማይ ይኩኑ ይቤ።	ኣምላኽ ድማ ንመዓልቲ ኻብ ለይቲ ዚፈልዩ ብርሃናት ኣብ ጠፈር ሰማይ ይኹን በለ።
ወለዘመን ወለመዋዕል ወለዓመታት ወይፍልጡ ማእከለ ዕለት ወማእከለ ሌሊት ወይኩኑ ለተኣምር።	ንዘበናትን ንመዓልታትን ዓመታትን ከኣ ንመፈለጥታ ይኹን።
ወዲበ ፡ ምድር ለአብርሆ ውስተ ጠፈረ ሰማይ ከመ ያብርሁ ይኩኑ።	ኣብ ምድሪ ንምብራህ ኣብ ጠፈር ሰማይ ብርሃናት ይኹን።
ወእግዚአብሔር ክልኤተ ዐበይተ ብርሃናተ ገብረ።	ኣምላኽ ከኣ ኽልተ ዓበይቲ ብርሃናት ገበረ።
ዘየዐቢ ብርሃን ከመ መዐልተ ይምልክ።	እቲ ዓብዪ ብርሃን ብመዓልቲ ኺስልጥን።
ወዘይንእስ ብርሃን ሌሊተ ከመ ይምልክ ምስለ ከዋክብቲሁ።	እቲ ንእሽቶ ብርሃን ድማ ብለይቲ ኺስልጥን፡ ከዋኽብቲውን ገበረ።
ሰማይ ወምድር ተፈጸመ።	ከምኡ ሰማይን ምድርን ተፈጸሙ።
ወእግዚአብሔር ገቢረ ግብሮ ዘሶ ዓለመ ፈጸመ።	ኣምላኽ ከኣ ነቲ ዝገበሮ ግብሩ ፈጸሞ።
ወበሳብዕት ዕለት እምዘሶ ግብሩ እግዚአብሔር አዕረፈ።	ብሳብዓይቲ መዓልቲ ድማ ኻብቲ ዝገበሮ ዅሉ ግብሩ ዐረፈ።
ወእግዚአብሔር እምዘሶ ግብሩ ዘአኀዘ ይግበር እስመ ባቲ አዕረፈ ለሳብዕት ዕለት ባረካ ወቀደሳ።	ኣምላኽ ከኣ ኻብቲ ዝፈጠሮን ዝገበሮን ዅሉ ግብሩ ብእኣ ስለ ዝዐረፈ፡ ነታ ሳብዐይቲ መዓልቲ ባረኻን ቀደሳን።

Appendix C: Tigrigna Short Form Expansion

Short Form	Expanded Form	Short Form	Expanded Form
ቤት ት/ቲ	ቤት ትምህርቲ	ሃ/ስላሴ	ሃይለስላሴ
ቤት ፍ/ዲ	ቤት ፍርዲ	ወ/ር	ወታደር
ት/ቲ	ትምህርቲ	ወ/ሮ	ወይዘሮ
ከፍለ ት/ቲ	ክፍለ ትምህርቲ	ወ/ሪት	ወይዘሪት
መ/ር	መምህር	ወ/ስላሴ	ወልደስላሴ
ፍ/ስላሴ	ፍቅረስላሴ	ቤት ፅ.ት	ቤት ፅሕፈት
ፕ/ር	ፕሮፌሰር	ቀ.ሚ/ስትር	ቀዳማይ ሚኒስትር
ዶ/ር	ዶክተር	ገ/ጊዮርጊስ	ገብረጊዮርጊስ
ቤ/ክርስትያን	ቤተ ክርስትያን	ም/ኣቦወንበር	ምክትል ኣቦወንበር
ቤት ም/ሪ	ቤት ምኽሪ	ተ/ሃይማኖት	ተክለሃይማኖት
ሚ/ር	ሚኒስቴር	ኮ/ል	ኮሌኔል
ሜ/ጀነራል	ሜጀር ጄነራል	ብ/ጀነራል	ብርጋዴር ጄነራል
ሌ/ኮለኔል	ሌቴናል ኮለኔል	ኣ/ኣ	ኣዲስ ኣበባ
ሓ/ማሕበር	ሓረስቶት ማሕበር	ደ.ኣንስትዮ	ደቂ ኣንስትዮ
ኢ/ያ	ኢትዮድያ	ገ/ልምዓት	ገጠር ልምዓት
ሕ.ወኪል	ሕርሻ ወኪል	ላ/ማይጨው	ላዕላይ ማይጨው
ታ.ማይጨው	ታሕታይ ማይጨው	ገ/ማርያም	ገብረማረያም
ገ/ዚሄር	ገረዚሄር	ሓ/ዓሰርተ	ሓለቓ ዓሰርተ
ሓ.ሚኢቲ	ሓለቓ ሚኢቲ	ሓ.ሽሕ	ሓለቓ ሽሕ
ሓ.ዘመን	ሓዲሽ ዘመን	ር/ምምሕዳር	ርእሰ ምምሕዳር
ዓ/ግ	ዓድግራት	ዕ.ሓሙስ	ዕዳጋ ሓሙስ
ማ/ጨው	ማይጨው	ማ/ሰብ	ማሕበረ ሰብ
ዓ.ዓ	ዓመተ ዓለም	ማ/ኮሚቴ	ማእኸላይ ኮሚቴ
ር/መምህር	ርእሰ መምህር	ፕ/ት	ፕሬዚዳንት
ሃ.ተፈጥሮ	ሃፍቲ ተፈጥሮ	ቤት ፍ/ሒ	ቤት ፍትሒ
ሚ/ሕርሻ	ሚኒስቴር ሕርሻ	ቤት ህ/ት	ቤት ህንፀት
ር/ከተማ	ርእሰ ከተማ	ዓ.ም	ዓመተ ምህረት

Appendix D: Training Iterations of RNN Model

Epoch 1

Epoch 1 Batch 1 Loss 1.7550
Epoch 1 Batch 2 Loss 1.6706
Epoch 1 Batch 3 Loss 1.7370
Epoch 1 Batch 4 Loss 1.2517
Epoch 1 Batch 5 Loss 1.3474
Epoch 1 Batch 6 Loss 1.2936
Epoch 1 Batch 7 Loss 1.5075
Epoch 1 Batch 8 Loss 1.3818
Epoch 1 Batch 9 Loss 1.3469
Epoch 1 Batch 10 Loss 1.4409
Epoch 1 Batch 11 Loss 1.4355
Epoch 1 Batch 12 Loss 1.6254
Epoch 1 Batch 13 Loss 1.2394
Epoch 1 Batch 14 Loss 1.3122
Epoch 1 Batch 15 Loss 1.5181
Epoch 1 Batch 16 Loss 1.3705
Epoch 1 Batch 17 Loss 1.4125
Epoch 1 Batch 18 Loss 1.3521
Epoch 1 Batch 19 Loss 1.4047
Epoch 1 Batch 20 Loss 1.3381
Epoch 1 Batch 21 Loss 1.5244
Epoch 1 Batch 22 Loss 1.2465
Epoch 1 Batch 23 Loss 1.2431
Epoch 1 Batch 24 Loss 1.5141
Epoch 1 Batch 25 Loss 1.2758
Epoch 1 Batch 26 Loss 1.4767
Epoch 1 Batch 27 Loss 1.2099
Epoch 1 Batch 28 Loss 1.3252
Epoch 1 Batch 29 Loss 1.5263
Epoch 1 Batch 30 Loss 1.2500
Epoch 1 Batch 31 Loss 1.5107
Epoch 1 Batch 32 Loss 1.3670
Epoch 1 Batch 33 Loss 1.2655
Epoch 1 Batch 34 Loss 1.3632
Epoch 1 Batch 35 Loss 1.3428
Epoch 1 Batch 36 Loss 1.4834
Epoch 1 Batch 37 Loss 1.2752
Epoch 1 Batch 38 Loss 1.1741
Epoch 1 Batch 39 Loss 1.1973
Epoch 1 Batch 40 Loss 1.2545
Epoch 1 Batch 41 Loss 1.4715
Epoch 1 Batch 42 Loss 1.3336
Epoch 1 Batch 43 Loss 1.1955
Epoch 1 Batch 44 Loss 1.2680
Epoch 1 Batch 45 Loss 1.3071
Epoch 1 Batch 46 Loss 1.3334
Epoch 1 Batch 47 Loss 1.2711
Epoch 1 Batch 48 Loss 1.4893
Epoch 1 Batch 49 Loss 1.2932
Epoch 1 Batch 50 Loss 1.5272
Epoch 1 Batch 51 Loss 1.2339
Epoch 1 Batch 52 Loss 1.1476
Epoch 1 Batch 53 Loss 1.2431
Epoch 1 Batch 54 Loss 1.4495
Epoch 1 Batch 55 Loss 1.5650
Epoch 1 Batch 56 Loss 1.3694
Epoch 1 Batch 57 Loss 1.4498
Epoch 1 Batch 58 Loss 1.3984
Epoch 1 Batch 59 Loss 1.1733
Epoch 1 Batch 60 Loss 1.3471
Epoch 1 Batch 61 Loss 1.4020
Epoch 1 Batch 62 Loss 1.5241
Epoch 1 Batch 63 Loss 1.3026
Epoch 1 Batch 64 Loss 1.4267
Epoch 1 Batch 65 Loss 1.1542
Epoch 1 Batch 66 Loss 1.4142
Epoch 1 Batch 67 Loss 1.4769
Epoch 1 Batch 68 Loss 1.4706
Epoch 1 Batch 69 Loss 1.4377
Epoch 1 Batch 70 Loss 1.2496
Epoch 1 Batch 71 Loss 1.3472
Epoch 1 Batch 72 Loss 1.4645
Epoch 1 Batch 73 Loss 1.3200
Epoch 1 Batch 74 Loss 1.4012
Epoch 1 Batch 75 Loss 1.4093
Epoch 1 Batch 76 Loss 1.4436
Epoch 1 Batch 77 Loss 1.4320
Epoch 1 Batch 78 Loss 1.3748
Epoch 1 Batch 79 Loss 1.2430
Epoch 1 Batch 80 Loss 1.3122
Epoch 1 Batch 81 Loss 1.2746
Epoch 1 Batch 82 Loss 1.4415
Epoch 1 Batch 83 Loss 1.4852
Epoch 1 Batch 84 Loss 1.3557
Epoch 1 Batch 85 Loss 1.3539
Epoch 1 Batch 86 Loss 1.2973
Epoch 1 Batch 87 Loss 1.2329
Epoch 1 Batch 88 Loss 1.3569
Epoch 1 Batch 89 Loss 1.4878
Epoch 1 Batch 90 Loss 1.2667
Epoch 1 Batch 91 Loss 1.2844
Epoch 1 Batch 92 Loss 1.1514
Epoch 1 Batch 93 Loss 1.2896
Epoch 1 Batch 94 Loss 1.3819
Epoch 1 Batch 95 Loss 1.1328
Epoch 1 Batch 96 Loss 1.3244
Epoch 1 Batch 97 Loss 1.4323
Epoch 1 Batch 98 Loss 1.3296
Epoch 1 Batch 99 Loss 1.2367
Epoch 1 Batch 100 Loss 1.3081
Epoch 1 Batch 101 Loss 1.4009
Epoch 1 Batch 102 Loss 1.2341
Epoch 1 Batch 103 Loss 1.4074
Epoch 1 Batch 104 Loss 1.2317
Epoch 1 Batch 105 Loss 1.3145
Epoch 1 Batch 106 Loss 1.3485
Epoch 1 Batch 107 Loss 1.2779
Epoch 1 Batch 108 Loss 1.2852
Epoch 1 Batch 109 Loss 1.4789
Epoch 1 Batch 110 Loss 1.3320
Epoch 1 Batch 111 Loss 1.2207
Epoch 1 Batch 112 Loss 1.2021
Epoch 1 Batch 113 Loss 1.1889
Epoch 1 Batch 114 Loss 1.3542
Epoch 1 Batch 115 Loss 1.3225
Epoch 1 Batch 116 Loss 1.2704
Epoch 1 Batch 117 Loss 1.2616
Epoch 1 Batch 118 Loss 1.2526
Epoch 1 Batch 119 Loss 1.2502
Epoch 1 Batch 120 Loss 1.3589
Epoch 1 Batch 121 Loss 1.1435
Epoch 1 Batch 122 Loss 1.2597
Epoch 1 Batch 123 Loss 1.2956
Epoch 1 Batch 124 Loss 1.2784
Epoch 1 Batch 125 Loss 1.4413
Epoch 1 Batch 126 Loss 1.3803
Epoch 1 Batch 127 Loss 1.3367
Epoch 1 Batch 128 Loss 1.3291
Epoch 1 Batch 129 Loss 1.5227
Epoch 1 Batch 130 Loss 1.2818
Epoch 1 Batch 131 Loss 1.2861
Epoch 1 Batch 132 Loss 1.3589
Epoch 1 Batch 133 Loss 1.2176
Epoch 1 Batch 134 Loss 1.3604
Epoch 1 Batch 135 Loss 1.2709
Epoch 1 Batch 136 Loss 1.5129
Epoch 1 Batch 137 Loss 1.4009
Epoch 1 Batch 138 Loss 1.4532
Epoch 1 Batch 139 Loss 1.3078
Epoch 1 Batch 140 Loss 1.2541
Epoch 1 Batch 141 Loss 1.2810
Epoch 1 Batch 142 Loss 1.4785
Epoch 1 Batch 143 Loss 1.2623
Epoch 1 Batch 144 Loss 1.2724
Epoch 1 Loss 0.0210
Time taken for this epoch 216.4023 sec

Epoch 52

Epoch 52 Batch 51 Loss 0.0034 Epoch 52 Batch 52 Loss 0.0034 Epoch 52 Batch 53 Loss 0.0059 Epoch 52 Batch 54 Loss 0.0008 Epoch 52 Batch 55 Loss 0.0004 Epoch 52 Batch 56 Loss 0.0044 Epoch 52 Batch 57 Loss 0.0027 Epoch 52 Batch 58 Loss 0.0010 Epoch 52 Batch 59 Loss 0.0095 Epoch 52 Batch 60 Loss 0.0013 Epoch 52 Batch 61 Loss 0.0025 Epoch 52 Batch 62 Loss 0.0027 Epoch 52 Batch 63 Loss 0.0018 Epoch 52 Batch 64 Loss 0.0016 Epoch 52 Batch 65 Loss 0.0049 Epoch 52 Batch 66 Loss 0.0030 Epoch 52 Batch 67 Loss 0.0013 Epoch 52 Batch 68 Loss 0.0055 Epoch 52 Batch 69 Loss 0.0085 Epoch 52 Batch 70 Loss 0.0005 Epoch 52 Batch 71 Loss 0.0023 Epoch 52 Batch 72 Loss 0.0036 Epoch 52 Batch 73 Loss 0.0007 Epoch 52 Batch 74 Loss 0.0021 Epoch 52 Batch 75 Loss 0.0050 Epoch 52 Batch 76 Loss 0.0032 Epoch 52 Batch 77 Loss 0.0033 Epoch 52 Batch 78 Loss 0.0007 Epoch 52 Batch 79 Loss 0.0015 Epoch 52 Batch 80 Loss 0.0038 Epoch 52 Batch 81 Loss 0.0037 Epoch 52 Batch 82 Loss 0.0028 Epoch 52 Batch 83 Loss 0.0028 Epoch 52 Batch 84 Loss 0.0026 Epoch 52 Batch 85 Loss 0.0053 Epoch 52 Batch 86 Loss 0.0019 Epoch 52 Batch 87 Loss 0.0006 Epoch 52 Batch 88 Loss 0.0016 Epoch 52 Batch 89 Loss 0.0026 Epoch 52 Batch 90 Loss 0.0007 Epoch 52 Batch 91 Loss 0.0039 Epoch 52 Batch 92 Loss 0.0038 Epoch 52 Batch 93 Loss 0.0026 Epoch 52 Batch 94 Loss 0.0030 Epoch 52 Batch 95 Loss 0.0015 Epoch 52 Batch 96 Loss 0.0030 Epoch 52 Batch 97 Loss 0.0024 Epoch 52 Batch 98 Loss 0.0032 Epoch 52 Batch 99 Loss 0.0030 Epoch 52 Batch 100 Loss 0.0031 Epoch 52 Batch 101 Loss 0.0049 Epoch 52 Batch 102 Loss 0.0067	Epoch 52 Batch 51 Loss 0.0034 Epoch 52 Batch 52 Loss 0.0034 Epoch 52 Batch 53 Loss 0.0059 Epoch 52 Batch 54 Loss 0.0008 Epoch 52 Batch 55 Loss 0.0004 Epoch 52 Batch 56 Loss 0.0044 Epoch 52 Batch 57 Loss 0.0027 Epoch 52 Batch 58 Loss 0.0010 Epoch 52 Batch 59 Loss 0.0095 Epoch 52 Batch 60 Loss 0.0013 Epoch 52 Batch 61 Loss 0.0025 Epoch 52 Batch 62 Loss 0.0027 Epoch 52 Batch 63 Loss 0.0018 Epoch 52 Batch 64 Loss 0.0016 Epoch 52 Batch 65 Loss 0.0049 Epoch 52 Batch 66 Loss 0.0030 Epoch 52 Batch 67 Loss 0.0013 Epoch 52 Batch 68 Loss 0.0055 Epoch 52 Batch 69 Loss 0.0085 Epoch 52 Batch 70 Loss 0.0005 Epoch 52 Batch 71 Loss 0.0023 Epoch 52 Batch 72 Loss 0.0036 Epoch 52 Batch 73 Loss 0.0007 Epoch 52 Batch 74 Loss 0.0021 Epoch 52 Batch 75 Loss 0.0050 Epoch 52 Batch 76 Loss 0.0032 Epoch 52 Batch 77 Loss 0.0033 Epoch 52 Batch 78 Loss 0.0007 Epoch 52 Batch 79 Loss 0.0015 Epoch 52 Batch 80 Loss 0.0038 Epoch 52 Batch 81 Loss 0.0037 Epoch 52 Batch 82 Loss 0.0028 Epoch 52 Batch 83 Loss 0.0028 Epoch 52 Batch 84 Loss 0.0026 Epoch 52 Batch 85 Loss 0.0053 Epoch 52 Batch 86 Loss 0.0019 Epoch 52 Batch 87 Loss 0.0006 Epoch 52 Batch 88 Loss 0.0016 Epoch 52 Batch 89 Loss 0.0026 Epoch 52 Batch 90 Loss 0.0007 Epoch 52 Batch 91 Loss 0.0039 Epoch 52 Batch 92 Loss 0.0038 Epoch 52 Batch 93 Loss 0.0026 Epoch 52 Batch 94 Loss 0.0030 Epoch 52 Batch 95 Loss 0.0015 Epoch 52 Batch 96 Loss 0.0030 Epoch 52 Batch 97 Loss 0.0024 Epoch 52 Batch 98 Loss 0.0032 Epoch 52 Batch 99 Loss 0.0030 Epoch 52 Batch 100 Loss 0.0031 Epoch 52 Batch 101 Loss 0.0049 Epoch 52 Batch 102 Loss 0.0067	Epoch 52 Batch 103 Loss 0.0063 Epoch 52 Batch 104 Loss 0.0004 Epoch 52 Batch 105 Loss 0.0017 Epoch 52 Batch 106 Loss 0.0061 Epoch 52 Batch 107 Loss 0.0014 Epoch 52 Batch 108 Loss 0.0026 Epoch 52 Batch 109 Loss 0.0009 Epoch 52 Batch 110 Loss 0.0032 Epoch 52 Batch 111 Loss 0.0055 Epoch 52 Batch 112 Loss 0.0041 Epoch 52 Batch 113 Loss 0.0065 Epoch 52 Batch 114 Loss 0.0061 Epoch 52 Batch 115 Loss 0.0041 Epoch 52 Batch 116 Loss 0.0021 Epoch 52 Batch 117 Loss 0.0054 Epoch 52 Batch 118 Loss 0.0018 Epoch 52 Batch 119 Loss 0.0038 Epoch 52 Batch 120 Loss 0.0024 Epoch 52 Batch 121 Loss 0.0027 Epoch 52 Batch 122 Loss 0.0027 Epoch 52 Batch 123 Loss 0.0078 Epoch 52 Batch 124 Loss 0.0029 Epoch 52 Batch 125 Loss 0.0007 Epoch 52 Batch 126 Loss 0.0024 Epoch 52 Batch 127 Loss 0.0043 Epoch 52 Batch 128 Loss 0.0012 Epoch 52 Batch 129 Loss 0.0042 Epoch 52 Batch 130 Loss 0.0042 Epoch 52 Batch 131 Loss 0.0024 Epoch 52 Batch 132 Loss 0.0006 Epoch 52 Batch 133 Loss 0.0043 Epoch 52 Batch 134 Loss 0.0061 Epoch 52 Batch 135 Loss 0.0028 Epoch 52 Batch 136 Loss 0.0026 Epoch 52 Batch 137 Loss 0.0060 Epoch 52 Batch 138 Loss 0.0049 Epoch 52 Batch 139 Loss 0.0018 Epoch 52 Batch 140 Loss 0.0027 Epoch 52 Batch 141 Loss 0.0010 Epoch 52 Batch 142 Loss 0.0013 Epoch 52 Batch 143 Loss 0.0046 Epoch 52 Batch 144 Loss 0.0059 Epoch 52 Loss 0.0000 Time taken for this epoch 200.1153 sec

Appendix E

List of Tigrigna and Geez Stop Words

Tigrigna Stop Words									
ኣብቲ	ማለት	እዩ	ከም	ነቲ	ኣብ	እቲ	ወይ	ድማ	ክሳብ
መን	እዉን	ከምዘለዎ	ኣበይ	እምበር	በቶም	ነዚ	ይኹን	ኣሎ	እዚ
ዝኾነ	ኸዓ	እዛ	እንታይ	እታ	ካብ	ነይሮም	ከምቶም	እቶም	እዉን
ኣብዚ	ነበረ	ኩሉ	ዘለዉ	ብዛዕባ	ከለዋ	ነታ	እታ	ግን	ጥራሕ
እዞም	እኳ	ምኽኖም	በቲ	ብዘይ	ስለ	እሞ	ዘለኩም	ድሕሪ	ቅድሚ
ክሳዕ	በዚ	በዙይ	ናይዞም	ኩሎም	ኢሎም	ናይቲ	እንተድኣ	ናይ	ዘለዎም
ከምኡውን	እዮም	ካብተን	ድኣ	ናይዞም	ናይዘን	ኣብዚአም	እዙይ	ዘለዋ	እዚአም
ስለዝኾነ	ምስቲ	ምኻኑ	የለን	ኮይኑ	ከምኡ	ምኽን	ምስ	የብሉን	እያ
ዝኾነ	ኮይኑ	ኣብዛ	ኣብዚኣ	ኣላ	ከምተን	እንትኸውን	ከምዚኣተን	ኣይኮነን	ከምቶም
ዘለና	ዘለካ	ዘሎ	የለውን	ዘላ	የብላን	ይኹንደኣምበር	ስለዝኾነ	ስለዝኾኑ	ብምኽኑ
ቅድሚ	ከምዘለና	ናትና	ከምእን	እየን	ነቶም	ብምኽኑ	ከከም	ከለዉ	ኣብዝሓ
ካሊእ	ክንዲቲ	ከምታ	ካልኦት	ከምዙይ	ድሕሪ	ከምዚኣ	ብቅድሚት	ንቶም	ንተን
ብምኽኑ	ናይዚ	ዝበሃሉ	መዓዝ	ምስኡ	ምስኡ	ከምዘለዋ	ክብሉ	ብዝኾነ	ምስኣ
ምሳና	ምሳኽን	እንትኾን	ማለትና	ምስአም	ማለትኪ	እንተዝኾን	ማለቶም	ማለትካ	ማለታ
ማለቱ	ንማለት	ምስኡ	እማ	ከለና	ማለቱ	እንተዝኾኑ	ብምኽኖም	ምስኣተን	ንማለት
Geez Stop Words									
እስመ	እለ	ታሕተ	ዉስተ	ኀበ	እም	መትሕተ	ማእከለ	መንገለ	እምነ
ምስለ	ሶበ	እንበለ	ከማኡ	ከማሁ	ኩሉ	ዘእንበለ	በእንተዝ	እምዛ	ድህረ
ለእለ	ባሕቱ	እንተ							