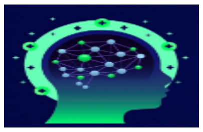
DEBRE BIRHAN UNIVERSITY

INSTITUTE OF TECHNOLOGY

COLLEGE OF COMPUTING

DEPARTMENT OF INFORMATION SYSTEMS



Bi-Directional English to Afaan Oromo Neural Machine Translation using a Deep Learning Approach

By: *Waldeyes Adere Gari*

A Thesis Submitted to Department of Information Systems of Debre Berhan University in Partial Fulfillment for the Masters of Science in Information Systems

Advisor: ***Gaddisa Olani Ganfure (PhD)***

Dire-Dawa, Ethiopia

Debre Birhan, Ethiopia

June -20- 2022

DEBRE BERHAN UNIVERSITY

INSTITUTE OF TECHNOLOGY

COLLEGE OF COMPUTING

Department of Information Systems

By: *Waldeyes Adere Gari*

Advisor: *Gaddisa Olani Ganfure (Ph.D.)*

This is to certify that the thesis prepared by **Waldeyes Adere Gari**, titled: *Bi-Directional English to Afaan Oromo Neural Machine Translation using a Deep Learning Approach* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Information Systems complies with the regulations of the University and meets the accepted standards concerning originality and quality.

Approved by Board of Examiners

Name		Signature	Date
Advisor:	Gaddisa Olani (Ph.D.)	[illegible]	05/09/2022
Internal Examiner:	Kinde Biredagn (Ph.D.)		
External Examiner:	Million Meshesha (Ph.D.)		

ACKNOWLEDGEMENTS

"The LORD is my strength and song, and he is become my salvation: he is my God, and I will prepare him an habitation; my father's God, and I will exalt him." Bible (Exodus 15:2)

Above all, I would like to thank Almighty God for giving me strength and courage to step forward for this work. The first person, I would like to thank is my advisor Dr. Gaddisa Olani for guiding me in the right direction and encouraging me throughout the course of this thesis.

I would like to express my deepest gratitude to my friend Mr. Abreham Bekele and Amdework Asefa for their support in preparing parallel corpus and shaping the structure of this thesis. I am also grateful to University of Arsi, College of social and humanities, Bokoji Campus especially to the Library and information service staff and Information Communication Technology staff, Ato Tsigu Gebre Tesema and Tasu Demie Yadete for providing me a suitable working environment. In addition, I would like to say tanks Mr. Yitayew Solomon for giving me an Afaan Oromo-English parallel corpus.

Last but not the least, I am thankful to my mother Kebinesh Bizu and my sister Mesrak wayu who have been of a great help throughout my life.

Waldeyes Adere

____

Date: June 21, 2022

DEDICATION

I would like to dedicate this thesis to my father Adere Gari Negawo and my loving family. Their prayers, support, encouragement and love have sustained me throughout my life.

Abstract

Machine Translation (MT) has gained much attention in recent years. It is a sub-field of computational linguistic which focus on translating text from one language to other language. The same MT approach may not work for European languages as for Afaan Oromo, because of its structure. MT based on Neural Networks (NN) methods has recently become an alternative approach to the statistical MT. In this thesis, we propose a design and implementation of a bi-directional MT between English and Afaan Oromo language pair. We carried out our study by exploring the capabilities of deep learning approach on the algorithms of the Recurrent NN (Bi-encoder and decoder Gated Recurrent Units (GRU) and Bi-encoder and decoder Long Short-Term Memory (LSTM)) with attention and Transformer model. A total of 21323 parallel corpora was used for the experiment, with percentage splits 80% of the entire corpus for training dataset, 10% of the corpus for validation dataset and 10% of the corpus for testing dataset. We have compared the performance of the three models on MT task between Afaan Oromo and English language pair. Two evaluation metrics have been used to compare the efficiency of the three models. These are BLUE and Perplexity. In terms of BLUE score, the BLUE score for Afaan Oromo to English MT increased from 6.48 while using the Bi-encoder and decoder LSTM with Attention model, and it increased to 15.04 while using the Bi-encoder and decoder GRU with Attention model, and it increased to 16.40 while using the Transformer model. In terms of perplexity, perplexity of the model for Afaan Oromo to English MT decreased to 150.377 while using the Bi-encoder and decoder LSTM with Attention model, and it decreased to 50.805 while using the Bi-encoder and decoder GRU with Attention model, and it decreased to 28.188 while using the Transformer model. The result of the comparison shows that Transformer based MT works better than the other two in terms of both metrics. Transformer model is mainly to reduce the number of trainable parameters, memory demand, as well as training time and performed best result as length of sentence increased when to our proposed models.

Key word: Artificial Neural Network, Afaan Oromo Machine Translation, Attention, Bidirectional LSTM, Bidirectional GRU, Transformer, Perplexity, BLUE score

Table of Contents

List of Tables

List of Figures

List of Appendix Figure

List of Acronyms

AI:	Artificial Intelligence	3
ALPAC:	Automatic Language Processing Advisory Committee	12
ANN:	Artificial Neural Network	26
BLEU:	Bilingual Evaluation Understudy	24
CBMT:	Context Based Machine Translation	38
CBMT:	Corpus-Based Machine Translation	13
CNN:	Convolutional Neural Networks	28
DDMT:	Data-Driven Machine Translation	13
DL:	Deep Learning	2
DNN:	Deep Neural Network	3, 28
DSR:	Design Science Research	7
EBMT:	Example Based Machine Translation	17
FFNN:	Feed Forward Neural Network	26
GPU:	Graphics Processing Unit	66
GRU:	Gated Recurrent Units	6
GTM:	General Text Matcher	24
HICATS:	Hitachi Computer-Aided Translation System	12
HMT:	Hybrid Machine Translation	13
IR:	Information Retrieval	11
IS:	Information Systems	7
LSTM:	Long Short-Term Memory	6
METEOR:	Metric for Evaluation of Translation with Explicit Ordering	24
MIT:	Massachusetts Institute of Technology	11
ML:	Machine Learning	65
NLP:	Natural Language Processing	2
NMT:	Neural Machine Translation	1
NN:	Neural Networks	2
OCR:	Optical Character Recognition	11
PP:	Perplexity	26
RBMT:	Rule-Based Machine Translation	13
ReLu:	Rectified Linear Unit	27
RNN:	Recurrent Neural Network	6
RNN:	Recurrent Neural Networks	28
Seq2Seq:	Sequence to Sequence	6
TER:	Translation Edit Rate	24

CHAPTER ONE

INTRODUCTION

This Chapter starts its discussion by giving general information about the overall thesis by stating the background of natural language and Neural Machine Translation (NMT). The discussion is followed by stating the problem behind the initiation of this research paper. Then, it discusses the general and specific objectives and research methodology. Finally, the Chapter is concluded by showing the roadmap of the organization of the whole thesis.

1.1 Background

Natural languages are languages spoken by people It is one of the most powerful tools of any living being to convey their thoughts to the other but it is only possible if the communicating subjects have the same language [1]. Communication is fundamental to the evolution and development of all kinds of living beings and it can be conducted in different ways such as audio (spoken), text written, and sign to exchange ideas, emotions, and information. With no disputes, languages should be recognized as the most amazing artifacts ever developed by mankind to enable communication. The Computer device has also become such a unique machine, due to its capacity to communicate with humans through languages [2]. It is worth mentioning that the languages understood by computers and humans are quite different, yet people can communicate with computers. This has been possible since the computer is fundamentally an artifact that can translate one language to another.

Despite the differences in languages, people still want to communicate with persons who use different languages. Differences in languages have become a barrier for cross-cultural communications. For instance, many nations have not been able to access a huge reservoir of world knowledge written in English, unless they have sound knowledge in English. On the other hand, people who don’t know English will not be able to contribute much to world knowledge. The importance of mother tongue is undisputable for the discovery and creation of new systems of knowledge. Consequently, this has resulted in what is called a language barrier for communication [3]. In fact, this issue is not only between English and other languages, but also between any two languages. Of course, people have been practicing a solution for the issue such as getting acquainted with the knowledge of both languages. However, can we really expect everyone to know every language?

To alleviate this issue, the emergence of digital computer technology in early 1950s had postulated the concept of machine translation [3]. Since then hundreds of research works have been conducted on MT between natural languages. MT is a classic sub-field in Natural Language Processing (NLP) that investigates how to use computer software to translate text or speech from one language to another without human involvement, which comes under the broad area of Artificial Intelligence (AI) [4]. It is commonly cited that machine translation has been one of the least achieved area in AI over the last sixty years. As such, a generic approach to machine translation has been an unrealized dream of researchers. MT has different advantages [5]. The first one is Confidentiality. That means since people use MT systems to translate their private information, people communicate only with the MT system than other individuals, as a result, the privacies of individuals are protected. The second advantage is fast translation. By using MT system it is possible to save time while translating large texts even paragraph or document in short period of time. The third one is universality. Usually a human translator translate the meaning of the text in their own context. This may bias the meaning of the text. But, in case of MT a text will be translated with the same meaning anywhere and everywhere, this makes machine translation universal. The other benefit of using MT in translation is as new data will be added to the model it will be able to adopt the changes independently. Moreover, a machine can handle multi-dimensional data as well as a multi-variety of data.

Recently, with the development of Deep Learning (DL), a new kind of method called NMT has emerged. As the name suggests, this is the new way of focusing on how do our brain and the human nervous system work [6]. NMT is closely observing the neural system of a human being. This helps it to understand the neural system and communication better. Through this, we can get to know how a normal human brain thinks and we can use it to map a new algorithm for it so that we can solve a problem through a machine just as it has been solved by a human brain. DL can also be seen as Neural Networks (NN) which have multi-layer architectures and very huge parameters on which it works [7]. Currently, the survey conducted on languages indicated that around 7000 languages are spoken worldwide [8]. In this era where most of our daily chores are performed by computer machines, why not machines should work as middleman translators for different people. Because for a machine it’s not difficult to learn 7000 languages.

Deep Neural Network (DNN) represents the type of machine learning when the system uses many layers of nodes to derive high-level functions from input information [9]. It means transforming the data into a more creative and abstract component. With strong progress in deep learning technology in speech, vision, and other fields, researchers began to apply deep learning technology to MT. DL is a subset of Artificial Intelligence (AI). This is generally represented using the following figure 1:1:

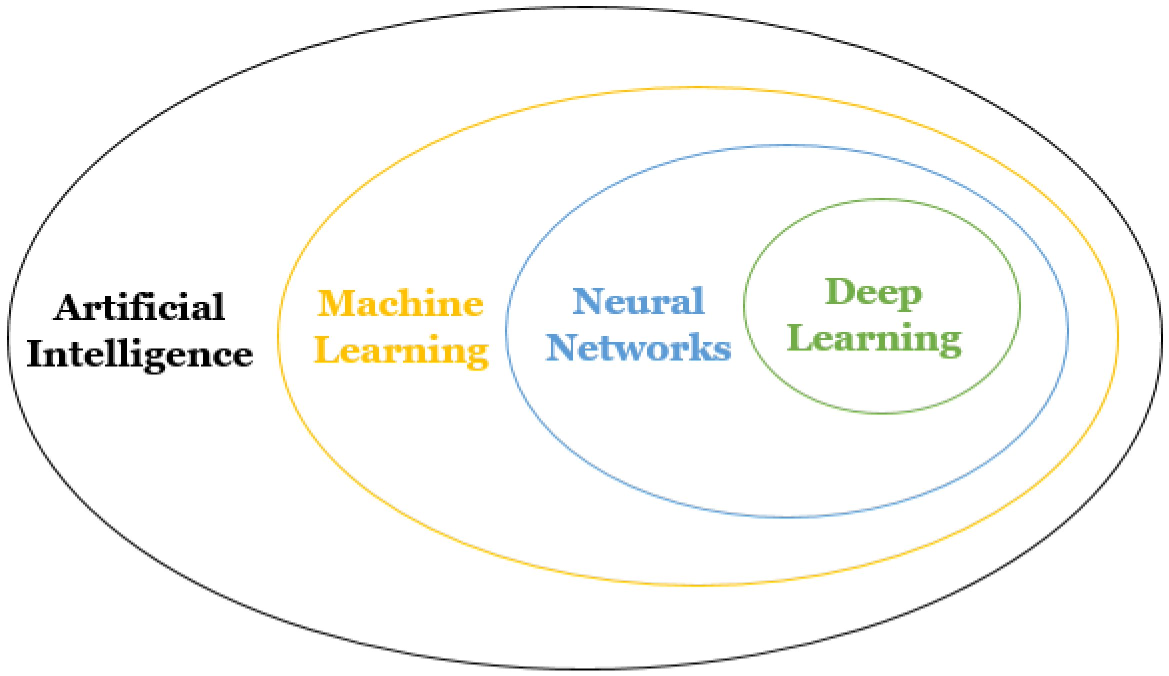


Figure 1:1 for subsets of Artificial Intelligence *[10]*

Most recently, MT tools and methods have achieved good performance in translation [11]. After a primitive model in origin, there are a variety of NMT models being proposed, some of which have achieved great progress with the state-of-the-art result [12].

In this research paper, we will demonstrate the efficiency and effectiveness of different deep learning approach in the task of bi-directional MT for English and Afaan Oromo language pairs. This research paper focuses on sentence to sentence translation. The Oromo language, also known as Afaan Oromo is one of the Cushitic languages, which form a branch of the Afro-Asiatic family. It is one of the major African languages that are widely spoken in Ethiopia. Currently, it is an official language of the Oromia regional state and also serves as an academic language for primary schools in the region. Oromo people are the native speaker of this language; they are the largest ethnic group in Ethiopia. According to the Ethiopia population census commission 2007 the total population of Oromo people which use Afaan Oromo language is 34.5% of the total population of the country [13]. According to, [13], it is also the third most widely spoken language in Africa next to Arabic and Hausa languages. It is widely spoken and used in most parts of Ethiopia (Oromia state) and also some parts of other neighboring countries like Kenya, Tanzania, Djibouti, Sudan, and Somalia. It is written in Latin script. Oromo language, literature, and folklore is being delivered as a field of study in many universities located in Ethiopia and other countries.

1.2 Motivation

The fact that initiated this study is enabling the development of Afaan Oromo to grow with current information technology support. Afaan Oromo is surprisingly resource-scarce from a technological point of view. Studies show that only twenty percent of the world population speaks English and fifty percent of the resource is available in the English language [8]. Creating an inclusive system for less translated languages like Afaan Oromo would create a better platform for the world to see and understand their civilization. Just as a technology needs to be improved, language needs to be studied to become the language of technology. They can write their science, their findings, and their cultural knowledge with their language and share it with the global world. That means Afaan Oromo to English translation system is helps to promote own language and culture to the world. It helps people who speak and conduct research in Afaan Oromo language to share and explain their ideas and research with the world. At the same time English to Afaan Oromo translation systems helps translate texts written in the English language into own (Afaan Oromo) language. But, our experimental system not compete with world-renowned translation system with relative robustness; having limited resources both in corpus and hardware requirement, but it will bring forth a system that which approach is best for low resource language such as Afaan Oromo in order to improve translation of languages with a little amount of parallel corpus and explore the challenge we will face while conducting our experimental study. And also show which approach and algorism is best for low resource language in order to solve the challenge behind to select best method before conduct the experiment.

1.3 Statement of the Problem

MT is a growing technology, much like any other technology. The growth in MT technology inspired different researchers to explore and investigate applications of MT for different languages and also understanding its limitation is key to successful implementation. As our world becomes increasingly connected, language translation provides a critical cultural and economic bridge between people from different countries and ethnic groups [14]. It is not possible to know and grasp all the languages within the world. According to [15], there are more than 7000 languages in the world and most translation research efforts have targeted a few high resource languages. Researches within the field of language translation are exploring the possibilities of message transferring from one language to another language [11]. MT plays an important role to handle language barriers between people and documents who want to access them. The manual translation method is boring and time-consuming. It takes a very long time, editing and evaluation is too costly (in terms of money, time, and raw materials like paper, vellum, and ink), and translation errors cannot be handled easily. It can be very difficult to translate, evaluate and deliver a translation in a short period. Currently, the demand for translation is increasing rapidly but those demands will not be fulfilled using a manual method and also MT systems. The current MT systems also need improvement. Before improved any systems or MT system, we must knowing what is the problem behind the system and which approach and algorisms is best to solve problem. So conducting experiment is important to knowing that. Therefore, MT is rolling a great job by translating to satisfy the demand for translation. Of course, MT has its own challenges but it's still an active research area [16]. The challenges are the translation of low-resource and morphologically rich languages such as Afaan Oromo get limited attention due to the unavailability of standardized parallel corpus which has a great effect on alignment between source and target languages [5]. Several studies and applications have been done for foreign languages using different methodologies and approaches. Most of the works have been done on language pairs of English and other languages, such as Japanese-to-English [17], English-Arabic [18] , English to Hindi[19], Manipuri-English [20], etc until to now. This is because English languages is the dominant language spoken across the world next to Mandarin and it is the language of international communication [21].

However, little work has been done on the MT system among English and Ethiopian languages. Some studies are carried out on English-local language pair [22]–[24], and English to Afaan Oromo language pair [5], [7], [25]. So, to simplify this translation task and to improve the translation accuracy level of MT, a thorough study has to be conducted. When we compare the English to Afaan Oromo translation result with the result of MT between English and other languages like German or Japanese, it is not adequate. Different researcher suggested different approaches in-order to enhance the performance of MT system. So if we want to recognize those suggestion we must conduct the experiment on low resource language and also one language different from the other language in terms theirs language structure. In recent times NMT models that learned from large, unstructured data using a language-independent architecture achieved the state-of-the-art result in MT. The NMT is a better choice for more accurate translation and it also provides better performance [4]. While resourceful languages such as English benefited from this technique, low resourced languages such as Afaan Oromo are barely represented. Comparing the different DL algorisms increase knowledge about their difference. So, comparing the Transformer to the Recurrent Neural Network (RNN) based sequence to sequence (seq2seq) model on the task of MT would increase knowledge on how the removal of recurrence in NN affects translation quality. Since the Transformer is a relatively new model in comparison with recurrent models, there are still areas to explore, such as how the removal of recurrence affects learning and performance in different contexts. So this research paper focuses on how well the Transformer performs on the MT task in comparison with a Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) seq2seq language model architecture by using bidirectional encoder and decoder with attention mechanism. This knowledge would help with model selection for low-resourced languages and provide insights into the readability of the output from encoder-decoder models and also we will try to explore the challenge during conduct the experiment across to our domain.

1.4 Research Question

In general, the consideration of the points discussed in Section 1.3 raise some central questions of the study. So the research paper will answer the following research questions at the end.

1. How does the Transformer algorithms perform on the task of MT in comparison with a GRU and LSTM algorithms by using bidirectional encoder and decoder with attention mechanism for low resource language such as Afaan Oromo?
2. What are the possible main challenges in the Afaan Oromo and English MT system?

1.5 Objectives of the Study

1.5.1 General Objective

The general objective of this study is to design and develop a bi-directional NMT system for English and Afaan Oromo language pairs by using a DL approach.

1.5.2 Specific objectives

To achieve the above stated General Objective, the following specific objects are devised:

- Review Literature to have a conceptual understanding of the problem area of MT, tools, methods and over all syntactic structure of pair's language that are used to solve the problem.
- Prepare and Pre-process a parallel corpus for pair's language.
- Design overall architecture of the proposed systems and develop a models based on designed architecture.
- Train and evaluate the developed models.
- Develop a prototype for demonstration purpose.
- Evaluate developed prototype UI usability and translated sentences quality.
- Discuss on the result and explore the challenge we faced during conduct those experiment.

1.6 Research Methodology

This section describes how this study was exhaustively done. The following subsections discuss the methodologies that are applied in this study

1.6.1 Research design

To conduct this research study, we follow Design Science Research (DSR) methodology. The method was derived from other DSR advocates such as [26]. DSR is suitable as it is a systematic problem-solving method for producing relevant, new, and innovative Information Systems (IS) solutions within a specific domain.

DSR is also relevant for addressing problems that have dynamic or variable requirements and other constraints [27]. DSR is a popular new research approach and paradigm, for which many research methodologies have been developed. DSR is a problem-solving methodology that focuses on the production of innovative and new knowledge to solve the problem and also decreases the gap between theory and practice by producing practical knowledge and it's flexible and modified according to the targets of the researcher [27].

This DSR process includes six activities, which cover the complete study from the start (motivation) to the end (communication) [27]. These phases are problem identification and motivation, the definition of the objectives for a solution, design and development, demonstration, evaluation, and communication. Therefore, the sequence steps for our study are described in Figure 1:2 in which each activity is described by box and the iterative nature of the DSR process is represented by the arrows between the six steps. The methodology was derived from other DSR methodology advocates such as [26].

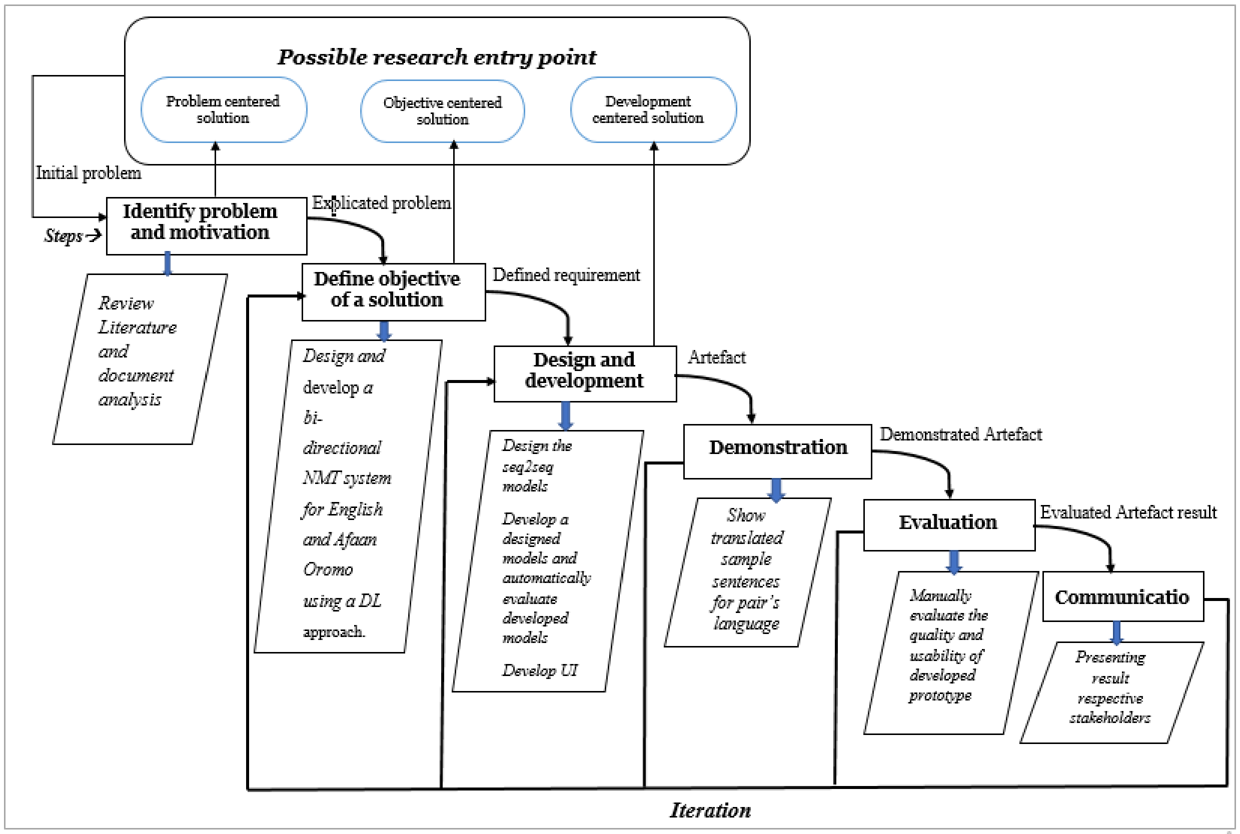

Possible research entry point
Problem centered solution
Objective centered solution
Development centered solution
Initial problem
Explicated problem
Steps
Identify problem and motivation
Review Literature and document analysis
Define objective of a solution
Defined requirement
Design and develop a bi-directional NMT system for English and Afaan Oromo using a DL approach.
Design and development
Artefact
Design the seq2seq models
Develop a designed models and automatically evaluate developed models
Develop UI
Demonstration
Demonstrated Artefact
Show translated sample sentences for pair's language
Evaluation
Evaluated Artefact result
Manually evaluate the quality and usability of developed prototype
Communicatio
Presenting result respective stakeholders
Iteration

Figure 1:2 step in the DSR methodology as applied in this study (Adapted from *[26]*)

This DSR process applied for implementing a bi-directional NMT system between English and Afaan Oromo. Le as describe each step under the following sub section.

1.6.1.1 Problem Identification and Motivation

First, identify the relevant problem within the domain as we try to described under section 1.2 and 1.3 and justify the value of a solution by using an extensive literature review and document analysis. Previous research works will be thoroughly discussed. This provide a basic explanation of the different methodologies proposed earlier. So this activities describes the specific research problem and justifies the value of a solution, by vast document analysis and review of literature are accessed.

1.6.1.2 Define the objectives of a solution

The objective that we described under section 1.5 inferred rationally from the problem specification. It means what is the problem we solve? That means “how the proposed Afaan Oromo and English bi-directional NMT is useful for community? Among proposed algorithm’s, which algorithm best for low resource language such as Afaan Oromo? Can bring the objective of a solution.

1.6.1.3 Design and development.

In this step, creating an artifact and determining the intended functionality of the artifact. According to [27], such artifacts are perhaps concepts, models, methods, and instantiations. This design phase includes the designing of all proposed systems which are in GRU and LSTM by using a bidirectional encoder and decoder with attention mechanism and transformer-based NMT to compare our proposed system with them. At this phase requires the consideration of the required components for training, validation, and testing. Then develop the models based on designed. After that train and evaluate the models by using automatic evaluation methods and finally develop prototype for best performance model.

1.6.1.4 Demonstration

This could involve its use in experimentation, simulation, case study, proof, or other appropriate activity in different fields of study [26], [27]. This activity demonstrates the use of the artifact to solve the identified problem. In this study, we demonstrate our prototype for end users in order to prove the use of the new system.

1.6.1.5 Evaluation

The evaluation measures how well the artifact or system supports a solution to the problem. It should also be noted that DSR is iterative, allowing the artifact to be improved through various iterations as required. Based on the result we decide whether to iterate back to step three for further improvement of the model effectiveness or to continue the next step. In this stud, in order to check the acceptance of the prototype usability and evaluate translated sentences quality we used manual evaluated method.

1.6.1.6 Communication

Appropriate forms of communication are employed depending upon the research goals and the audience, such as practicing professionals [27]. At the end, in case of our domain all parts of the problem and the designed and developed systems prototype are presented to the respective stakeholders. The findings of this research were communicated through research reports academic publications as well as a master thesis.

1.7 Scope and Delimitations of the study

This research focuses on designing and implementing deep learning NMT approaches for translating a sentence written in English to Afaan Oromo sentence and vice versa. It does not include, document-level translation, speech to speech translation, text to speech translation, and speech to text translation. It focuses on sentence to sentence translation from a source language to the target language. For the source of the dataset, we will use an existing domain-specific bilingual corpus having 4,362 parallel sentences prepared by previous research [5]. The author collected the dataset from Federal Democratic Republic of Ethiopia (FDRE) criminal code, FDRE constitution, Megeleta Oromia. We will also collect and prepare additional corpus from the Holy Bible. Because of the unavailability of the standardized corpus.

1.8 Significance of the study

The contribution from this research paper can be summarized as follows:

- ✓ The output of this research paper can be used as a base for future research papers which will be done in the area of MT and other NLP tasks.
- ✓ The parallel corpus of this research paper can be used as a good input for future research papers which will be done in the area of English and Afaan Oromo MT and other NLP applications such as cross-language information retrieval for both language pairs.

1.9 Thesis Organization

This thesis paper is organized into Six Chapters including the current one. Chapter Two presents a literature review on machine translation and related works of our domain. Chapter Three presents an overview of Afaan Oromo and the English language. Chapter Four presents the design of the proposed system. The experiments and results are discussed in Chapter Five and finally, conclusion and future works are presented in Chapter Six.

CHAPTER TWO
LITERATURE REVIEW

This Chapter discusses about NLP, approaches to NLP and its applications like MT. It further explains MT, history of MT, development of MT, challenges in MT and evaluation of MT systems and also about ANN. Finally, related work are discussed

2.1 Natural Language Processing (NLP)

NLP is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things [25]. The term NLP surrounds a wideset of techniques for automated generation, manipulation, and analysis of natural or human languages. NLP is also a subfield of AI in the area of computer science. MT, automatic summarization, Information Retrieval (IR), Optical Character Recognition (OCR), speech recognition, and text to speech conversion are among the operations that can be carried out by way of NLP [29]. This paper strives to study one of the most popular applications of NLP like, MT and provides a brief description of common MT approaches that are being used for different NLP tasks. Also, this paper presents a review of various approaches to NLP and some related topics to NLP and MT.

2.2 Machine Translation systems

MT, which is a field of NLP, can be employed for the translation of speech or text in a source language into the target language [12]. Due to various reasons associated with the complexity of languages, for more than the last sixty years, MT has been identified as one of the least achieved areas in computing [29]. MT systems can be described as devices that perform translations for any given pair of languages.

2.2.1 Historical Overview of MT

Investigations on MT began in the year 1949. Warren Weaver, who coined the phrase ‘computer translation’, suggested the use of computers for NLP. Under the guidance of Yehoshua Bar-Hillel, the initial symposium organized for machine translation was held in 1952 at MIT (Massachusetts Institute of Technology). The original mechanical ‘Russian to English translator’ made its appearance in 1954 [30]. World-renowned linguists and computer scientists were present at the initial international conference on machine translation. Organized under the heading ‘Languages and Applied Language Analysis of Teddington’, this conference was held in 1961. A committee known as Automatic Language Processing Advisory Committee (ALPAC) was set up in the year 1964 [31].

During 1970 to 1980, the project named REVERSO was initialized by some Russian researchers (1970), another MT system named SYSTRAN1 (Russian to English) by Peter Toma, was developed. A MT system ATLAS2 (Korean to Japanese) by FUJITSU (a Japanese firm) was developed. FUJITSU was founded on rules (1978) [31], [32].

The period stretching from 1980 to 1990 saw the Japanese make great strides in the field of MT. NEC introduced its MT system in 1983. This system, which is called the 'Honyaku Adaptor II', utilizes the PIVOT algorithm for inter-lingual translations. Not to be left behind, Hitachi developed a Japanese to English language translation scheme known as Hitachi Computer-Aided Translation System (HICATS) [31], [32].

On a basic level, MT performs mechanical substitution of words in one language for words in another, but that alone rarely produces a good translation, because recognition of whole phrases and their closest counterparts in the target language is needed. Not all words in one language have equivalent words in another language, and many words have more than one meaning. In addition, two given languages may have completely different structures. However, due to the complexity of natural languages, development of the MT systems has become a research challenge. In addition, many researchers have also noted that Operational syntax, idioms, and Universal syntactic categories are some completely unsolved linguistic problems in MT [16], [31].

During 1990 to 2000, the use of MT touched new heights as it became a part of the internet. In 2005, the initial website for automatic MT was set up by Google. In 2008, MT took a hype with 23% of Internet users exploring MT features and 40 % considering doing so and in 2009, 30% of the professionals have started utilizing MT systems for their work, 18% perform a proofreading and 50% planned to use MT for translation (2010). As all these research is for the language translation, it already includes ambiguities [33]–[35].

2.2.2 Why Machine Translation Matters?

Communication is fundamental to the existence and survival of humans as well as to an organization. The ability to communicate with one another is a fundamental part of being human.

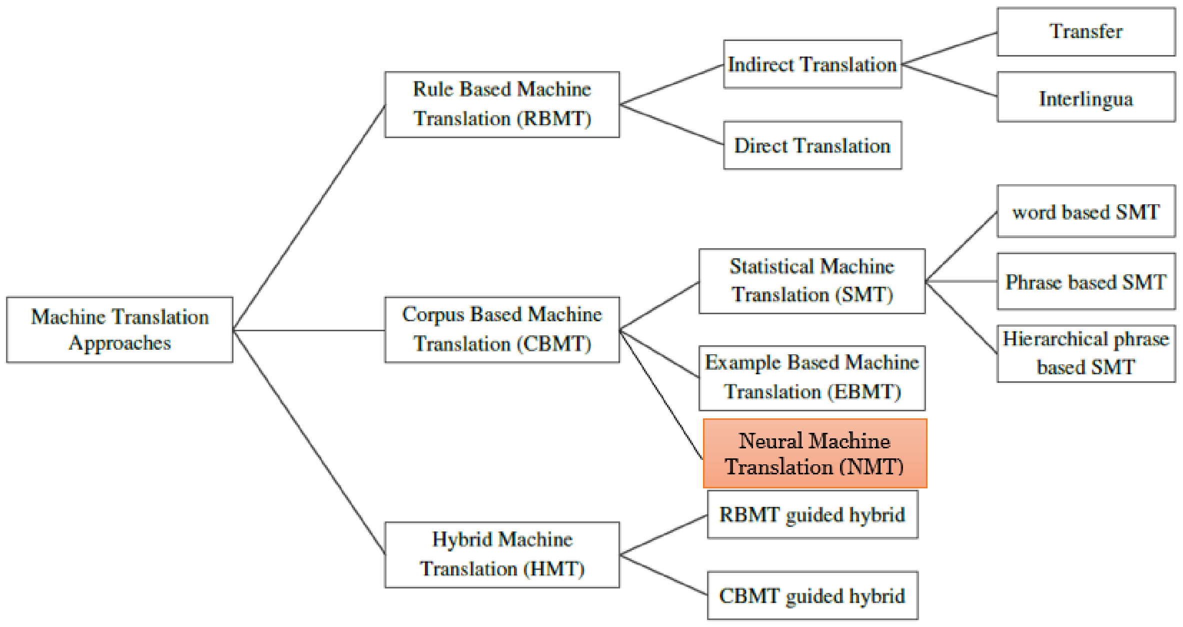
As our world becomes increasingly connected, language translation provides a critical cultural and economic bridge between people from different countries and ethnic groups.

Alongside increasing globalization comes a greater need for readers to understand texts in languages foreign to them. For example, approximately 48% of the pages on the Web are not available in English [36]. The translation aims to support users who need to access content in a language in which they are not fluent. Besides the scientific value, MT also has huge potential of saving labor costs in many practical applications, such as business**,** commerce**,** media**,** education (sharing of ideas, collaboration, translation of research papers), and government (foreign relations, negotiation) [37]. MT is a tool that can help businesses and individuals in many ways. While MT is unlikely to replace human beings in any application where quality is really important, there are a growing number of cases that show how effective and useful it can be.

MT enables global companies to translate content at scale using "machines" such as google translate. It is often found as a feature that is integrated into localization platforms and used by companies looking to lower their translation costs. To meet these needs, technology companies are investing heavily in MT. This investment and recent advancements in deep learning have yielded major improvements in translation quality [38]. However, translation is a difficult task due to the complexity of natural languages and their structure. In addition, manual translation does not scale to the magnitude of the web. One remedy for this problem is MT [36].

2.2.3 Development of Machine Translation Approaches

Over the years, researchers have developed different methods to reduce the amount of manual work and human intervention and increase the amount of automatic work, and machine-dependent translation. The methods in MT are mainly divided into three categories; the first one knowledge-driven procedure or Rule-Based Machine Translation (RBMT), secondly the Data-Driven Machine Translation (DDMT) method or Corpus-Based Machine Translation (CBMT), and the third is Hybrid Machine Translation (HMT) method which merges the plus points of the two previously-described approach. While RBMT is based on the linguistic theory, CBMT is based on the data theory [39]. Overtime, different approaches for MT were defined and gained maturity for practical use today. The MT approach are depicted in Figure 2:1.



Machine Translation Approaches
Rule Based Machine Translation (RBMT)
Indirect Translation
Transfer
Interlingua
Direct Translation
Corpus Based Machine Translation (CBMT)
Statistical Machine Translation (SMT)
word based SMT
Phrase based SMT
Hierarchical phrase based SMT
Example Based Machine Translation (EBMT)
Neural Machine Translation (NMT)
Hybrid Machine Translation (HMT)
RBMT guided hybrid
CBMT guided hybrid

Figure 2:1 Machine translation Approach *[40]*

2.2.3.1 Rule Based Machine Translation (RBMT)

This system came into being during the 1940s. It comprises a set of rules (grammar rules), a bi-lingual or multi-lingual lexicon, and software programs for managing the rules [41], [42]. Rule-based MT uses handwritten linguistic rules for both languages in its translation process. It requires a lot of human effort to define the rules and modifications of rules usually cost very high. It works based on a manually determined set of rules encoded by linguistic experts. The putting together of the rules for this system requires a substantial amount of time and effort as it is very closely aligned to the language theory. This approach gives grammatical correct translation by using a set of rules. It depends on the semantic information of both source and target languages.

The rule-based machine translation methods include dictionary-based MT, transfer-based MT, and Interlingua MT [40]. The Vauquois Triangle illustrates these levels of analysis and shows their dissimilarities as depicted in Figure 2:2.

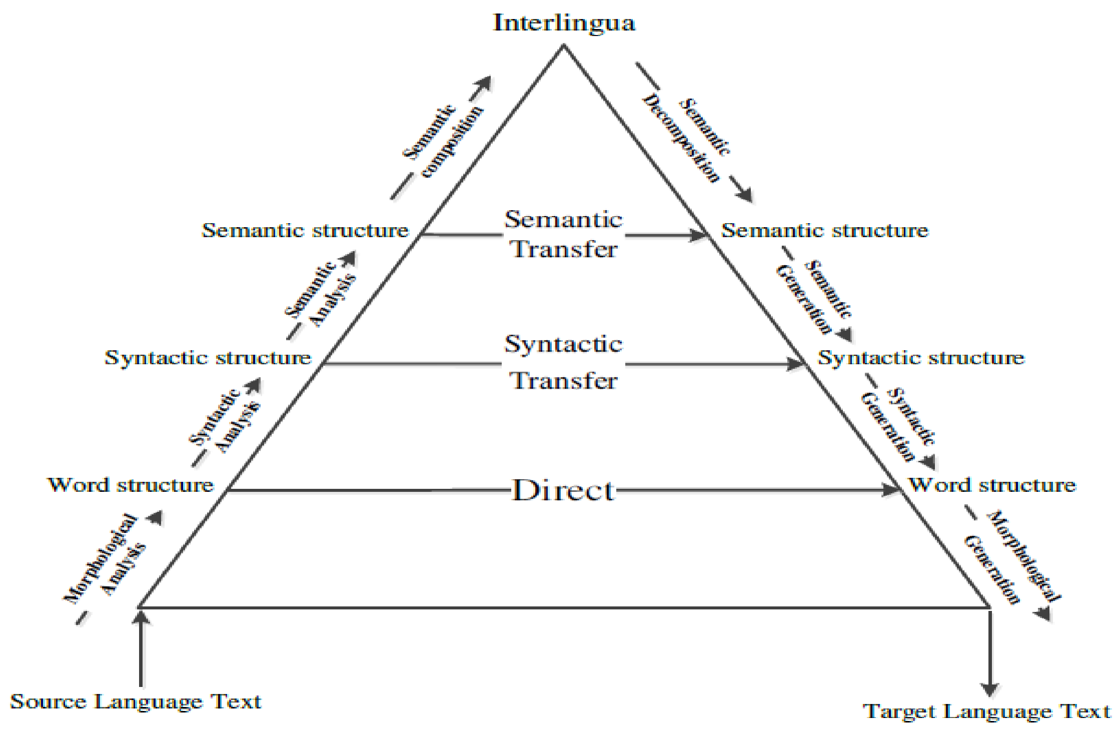


Figure 2:2 The Vauquois triangle *[41]*

RBMT has some disadvantages as in and first one is unavailability of good dictionaries. New dictionary building is truly high-priced task. Another limitation is, it's necessary to set some linguistic information manually. In addition, it's very difficult to manage rule interactions and ambiguity in the large system. RBMT allows building new rules and extends it but these changes are very expensive [11]. To be useful in practice, RBMT systems consist of large collections of rules, developed manually over time by translation experts, mapping structures from the source language to the target language. It requires linguistic experts to apply language rules to the system. The RBMT system is applicable for both direct and indirect translations.

2.2.3.1.1 Direct Machine Translation (Dictionary Based Machine Translation)

In dictionary-based MT, source language is directly converted to target language without using intermediate steps. It uses entries in a language dictionary to find a words equivalent in the target language [39]. Using a dictionary as the sole information source for translation means that the words will be translated as they are translated in a dictionary. This approach needs only a little syntactic and semantic analysis. It is a word by word translation approach with some simple grammatical adjustments. It depends on dictionary look-up. Without the analysis of internal structure and grammatical correlation, the source sentence is morphologically analyzed to derive target sentence [43].

2.2.3.1.2 Indirect translation

Indirect translation begins with the conducting of structural analysis (morphology, semantic and syntactic) for every SL input text [41]. Subsequently, the input text is transformed into intermediate representation mainly in the structure of an abstract parse tree. Based on the specific generator portrayed on the Vauquois triangle in Figure 2:2 [41], the target text is then acquired by way of structural conversion. For the most part, indirect RBMT is employed for multi-lingual translations. Transfer and Interlingua are the two MT approaches that can be applied for indirect translation purpose.

A. **Transfer Based Machine Translation**

The source sentence is transformed into an intermediate, less language-specific structure [42]. This structure is then transferred into a similar structure of the target language and, finally, the sentence is generated in the target language. The transfer uses morphological, syntactic, and semantic information about the source and target languages.

B. Interlingua Machine Translation

In this method, source language is translated into an intermediary representation which does not depends on any languages [42]. The target sentence is then generated out of the Interlingua. The inter-lingua approach is clearly most attractive for multilingual systems [41]. This Interlingua is neutral representation and related to the structure of both languages. This Interlingua based translation needs to drive the inter-mediatory representation between languages of having similar structure and difficult for the languages of having different structures.

2.2.3.2 Corpus Based Machine Translation (CBMT)

This translation procedure is based on the employment of bilingual parallel aligned corpora. Corpus is a very large collection of text produced by real users of the language and used to analyses how words, phrases and language in general are used [43]. It is used by linguists, lexicographers, social scientists, humanities, experts in NLP and in many other fields. A corpus is also be used for generating various language databases used in software development such as predictive keyboards, spell check, grammar correction, text/speech understanding systems, text-to-speech modules , MT and many others [43]. Such a corpus may not be available for under-resourced languages pending its generation by another party. In the area of MT, CBMT's are categorized as follows:

2.2.3.2.1 Example Based Machine Translation

Example Based Machine Translation (EBMT) belongs to corpus-based approaches because examples are extracted from large collections of bilingual corpora. It is characterized by its uses of bilingual corpus with parallel texts as its main knowledge. In this approach, the corpus that is used is one that contains texts that have already been translated. Given the source sentence, sentences with similar sub-sentential components are extracted from the source side of the bilingual corpus, and their translations to the target language are then used to construct the complete translation of the sentence [43]. If the text that is not yet translated or a superior translation is found, then the corpus is updated by the administrator for future translations. The main advantage of this model is it work well with small set of data and possible to generate output more quickly by train the translation program. However, some problems that had already been solved for linguistics based methods reappeared. The majority of these problems are connected to the issue of ambiguity, including syntactic and semantic variations [44].

2.2.3.2.2 Statistical Machine Translation (SMT)

SMT is based on statistical methods. It also belongs to corpus based approaches, as statistical methods are applied on large bilingual corpora. Building a SMT system does not require linguistic knowledge [40]. SMT has been the mainstream technology for the past 20 years. This system translates source text to target text depends on the analytical methods uprooted from the huge volume of alliance bilinguals text corpora. The statistical models help the system to obtain the perfect translation for the text while translation is performed [43]. After examining the parallel corpus, SMT algorithms automatically learn how to translate new sentences. SMT has three components [40]: - translation model, language model and decoder. Where in that a source language sentence *s* may translate into any target language Sentence *t*. It is based on statistical searching of the most likely translation from a huge bilingual corpus.

- **The language model** which computes the probability level of the target language ($P(t)$)
- **The translation model** which computes the conditional probability level of the target language output given source language input ($P(t|s)$)

- **The decoder model** which provides the most excellent translation achievable (*t***)** by elevating the two abovementioned probability levels to their highest points. This is portrayed in the following equation 2:1 which involves the use of the search algorithm. In addition, $\boldsymbol{argmax}_t$ (*search algorithm*) *t* is the Search algorithm in decoder that searches for the best translation from the given all possible translations based on the probability estimates (*P(t*|*s*)) and (*P(t)*) and performs the actual translation.

$$t = argmax\ s\ (P(t|s) * P(t)) \qquad \textit{Equation 2:1}$$

Where (*t*|*s*)**:** The *Translation* model that provides the probabilities of possible translation pairs of the source sentence $\boldsymbol{s}$ given the translated sentence $\boldsymbol{t}$. The $\boldsymbol{P}(s)$ is the *Language* model that provides a probability to each unit of text. In addition, $\boldsymbol{argmax_s}$ is the Search algorithm in decoder that searches for the best translation from the given all possible translations based on the probability estimates $\boldsymbol{P}(s|t)$ and $\boldsymbol{P}(s)$ and performs the actual translation [41].

The important feature of this method is no customization work is required by linguists because the tool learns translation methods through statistical analysis of bilingual corpora. Also, this tool is cheaper than that of rule-based tools. The approach is subdivided into three approaches namely: Word based SMT, Phrase based SMT and hierarchical based SMT [5].

Word-based SMT: Sentences are broken down to the fundamental unit (word) and translation for source language to target language is done word by word. Upon the generation of the target words, they are assembled in a particular sequence by way of a reordering algorithm. This facilitates the creation of the target sentence. However, the inadequacy of this method is exposed when it comes to the management of compound words [5].

Phrase-based SMT: Mainly uses phrases instead of single words as the fundamental unit of translation. It involves the separation of the source and target language sentences in the parallel corpora into phrases. Phrase based translation models are derived from a word aligned parallel corpus. Though Phrased based SMT may result to better performance, long phrases may degrade the performance [5].

Hierarchical phrases-based SMT: The model involves the combination of phrase based SMT and syntax-based translation. Phrase based consists of the unit of block or segment of translation while the syntax translation brings the rules of translation. Statistical word-based translation model is based on the idea that every target sentence is a possible translation of every source sentence. A reasonable assumption is that the probability of a given target sentence being a good choice for translation relies heavily on which source sentence is under consideration for translation [40].

2.2.3.2.3 Neural Machine Translation (NMT)

Nowadays, a novel SMT paradigm has arisen called NMT which relies on NN algorithms. NMT has been achieving impressive results and is now the state of the art in MT approaches [44]. It has made rapid progress in recent years, and it is paving its way into the translation industry as well. NMT is a DL-based approach to MT that uses a large NN based on vector representations of words. It is a recently proposed method utilizing special neural grid framework called Encoder-Decoder architecture. The goal of NMT is to build a model that maximizes the translation performance. The representation of source sentence is a sequence of words and it uses vector representation to store the encoded meaning of the source [43].

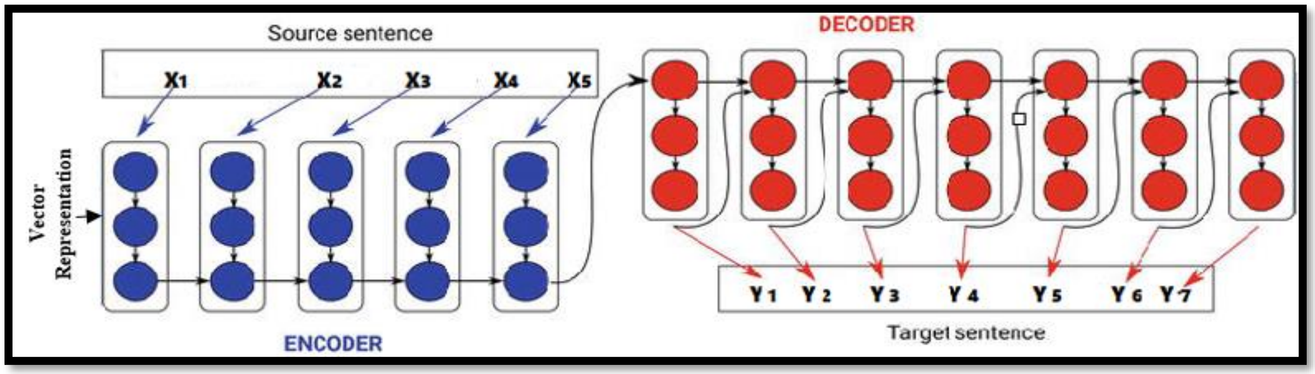


Figure 2:3 Neural machine translation system using the encoder-decoder architecture *[39]*.

The input words are passed through the layers of the encoder (blue circles) to its last layer, the context vector, updating it for every input word. The context layer is then passed through the decoder layers (red circles) to output words, and it is again updated for each output word. The entire encoder-decoder process, consisting; of the encoder and the decoder combine, is mutually prepared to maximize the likelihood of a rectified interpretation given a source sentence [7]. Although effective, the NMT systems still suffer some issues, such as scaling to larger vocabularies of words and the slow speed of training the models [39].

A. Neural Machine Translation Challenges

Different researchers have examined many challenges to NMT and given empirical results on how well the technology currently holds up, compared to traditional SMT. NMT has shown better results in recent research. According to Koehn et.al [16], Until now six types of challenges related to NMT are discussed in the following section.

Domain Mismatch: Different words have different translations in different context. To produce domain specific or context aware translation is a big task in NMT as training data usually available in general perspective. NMT systems have a lower quality out of domain. In order to robust performance of NMT task we must extended training dataset domain. To select appropriate translation according to situation is real problem in MT.

Amount of Data: Amount of training data always plays an important role in NMT model training. In order to enhance NMT system it need more and more data. As the number of training data set increase the machine is more trained. But for luck resource language it is difficult to build more data.

Out of Vocabulary Words: Out of Vocabulary (OOV) is a Linguistic Unit or a token that does not appear in training vocabulary. Although NMT systems outperform SMT systems on the translation of very infrequent words, but still challenging for it too. Reasons of OOV or rare words includes word missing in training data, misspelled words, technical terms and foreign words that usually cannot be translated.

Long Sentences: Another challenge is to maintain translation quality of the system on longer sentences. The authors of Koehn et.al [16] have shown that the recently introduced NN translation systems suffer from a significant drop in translation quality when translating long sentences, unlike existing SMT systems. Different researchers have shown that NMT produces significant results on sentences having length up to 60 words. The performance of NMT decreases on sentences where length increases from 60 words which may be improved by introducing extra layers in network [16], [45].

Word Alignment: Word alignment is another issue in NMT. Usually, the attention based NMT models does not always fulfill the role of a word alignment from source to target sentences, but may in fact dramatically diverge. In parallel corpora single sentences in one language can be found translated into several sentences in the other and vice versa. Sentence alignment can be performed through different alignment algorithm. Corpus creation can be costly for users with limited resources, the results are unexpected, and NMT does not work well between languages that have significantly different word orders. Words may have different alignment that is one-one or one-many or many-one and many-many this makes alignment of words difficult.

Beam Search: The other challenge NMT faces is beam search, closely related to this sequential model. In NMT, search strategy is the major issue for extracting best translation of given word. Optimal beam sizes are between 30 -50 in many cases (sometimes up to 200), however, quality is still drop in larger beams and NMT only improves translation quality for narrow beams and is bad when exposed to larger search space. In NMT, this is done by simple beam search decoder which finds the best suitable translation by translating word by word having fixed size of beam. Having fixed size of beam, it gives good results for narrow and small beams only. As explained in Beam search strategies for NMT, new sentences are translated by a simple beam search decoder that finds a translation that approximately maximizes the probability of a trained NMT model. The beam search strategy generates the translation word by word from left to right while keeping a fixed number (beam) of active candidates at each time step. To address this issue, [16] has discussed different pruning techniques.

2.2.3.2.4 Hybrid Machine Translation (HMT)

The aggregation of two or more translation systems is known as Hybrid Machine Translation (HMT)**.** This approach seeks to extract the benefits of both the systems by utilizing accessible assets [43]. Subsequently, RBMT and CBMT have been combined in order to resolve the drawbacks of these two families of approaches. While the rule-based procedure is acclaimed for its raised precision level, it is also time consuming and expensive to develop. In comparison, data driven systems are more favorable in terms of coverage and costs. However, CBMT is disadvantaged by its requirement for corpora, and this setback is particularly evident in the context of under-resourced languages. HMT poaches and merges the plus points of both RBMT and CBMT [44]. The different machine translation approaches that are used in translating source language texts to target language texts are reviewed individually [43]. From the review the different machine translation systems are compared and differences among them are shown in the below table 2:1.

Table 2:1 Comparison of machine translation Approach’s

No	MT approach	Advantages	Disadvantage
1	Direct Approach	✓ It is implementation is easy. ✓ For language with similar structure and grammar rules, the translation is effective.	✓ For multilingual scenarios, it can be expensive. ✓ Involves only in lexical analysis
2	Interlingua Approach	✓ Well suited for domain approach. ✓ Produce content stationed portrayal and can be applied in IR (Information Retrieval)	✓ Time efficiency is less than direct approach. ✓ Major problem in defining Interlingua.
3	Transfer based approach	✓ It defines the modular structure. ✓ It can easily handle the ambiguities in different languages.	✓ A part of the origin text content may be vanished at the foot of the translation.
4	Statistical Approach	✓ SMT systems are not designed for any particular language pair. ✓ It is less expensive compared to RBMT ✓ Translation used to be natural as it is trained by the real time texts.	✓ The creation of corpus is expensive. ✓ Errors are hard to find and fix. ✓ It requires extensive hardware configuration.
5	Example based Approach	✓ It avoids the use of manually driven rules. ✓ It is adaptable to many languages. ✓ To make the system attractive, minimum prior knowledge is required.	✓ It require large database. ✓ The system is not efficient as it consists of noisy corpora.
6	Neural Approach	✓ NMT output are more fluent. ✓ It performs better in terms of inflection and reordering ✓ Occupies lesser memory than STM. ✓ Table comparison of machine translation Approachs	✓ It performs rather poorly for long sentences. ✓ The vocabulary is limited and source coverage problem. ✓ Emprises translation problem
7	HMT approach	✓ Used to overcome the limitations of each approach and to enhance the quality of translation.	✓ Need for extensive editing. ✓ More Complex work than single machine translation approach’s.

2.3 Application of MT

MT is already widely used in many areas due to its low cost, high efficiency, and high translation quality [46]. Text translation is the most common form of MT application. Below are some typical applications of text translation.

Webpage translation. With the rapid progress of globalization, there is an increasing need for quick acquisition of information in foreign languages. While it is expensive and time-consuming to hire human translators to translate a huge number of webpages, MT provides a convenient way to view webpages in foreign languages.

***Scientific literature translation*.** Users such as researchers, engineers, and graduate students often use MT to read scientific literature such as papers and patents in their own language, or to translate their work into other languages.

E-commerce translation. MT is widely used in transnational online trade. With the help of MT, sellers can effectively translate their website, product information, and manuals into foreign languages, while buyers can easily buy products from all over the world. MT is also used in customer services to improve service quality and efficiency.

Language learning. Current MT systems usually provide rich functions, including translation, high-quality dictionaries, sentence pair examples, and so forth. Users can thus conveniently determine the meaning of a word or phrase and learn how to use it. Student users often input a whole paragraph for comprehension reading and use the sentence pair examples to help in their writing.

2.4 Challenges in Machine Translation

The fast progress of MT has boosted translation quality significantly, but, unfortunately, machine translation approaches are not equally successful for all language pairs [40], [45]. To select appropriate translation according to situation is real problem in MT. Particularly, many languages in the world have different syntactic structure and morphologically complexity. Morphologically rich languages are problematic in MT, especially if the translation is from a morphologically less complex to a morphologically more complex language. Morphological distinctions not present in the source language need to be generated in the target language. Much work on morphology aware approaches relies heavily on language-specific tools, which are not always available. Many morphologically rich languages fall in the category of low-resource languages [47], [48].

Syntactic structure is another challenge of MT. We describe detail under chapter three. A word can have more than one meaning due to Semantic (out of context), Syntactic (in a sentence), and Pragmatic (situations and context) meanings, Technical Verbs, paragraphs with symbols and Equations, and abbreviated word are very difficult to translate. According to Koehn et.al, Since NL are ambiguous, context-dependent, and ever evolving [16]. Finally after conducting our excrements we will explore the challenge regarding to our domain.

2.5 Evaluation of MT systems

It is not just enough to translate text from source language to target language; we also need a way to measure the quality of the translated text. It is necessary to evaluate MT quality before use in practice. Evaluation of the MT system has been received significant attention in the past few years. To evaluate the MT systems, several methods are used. These evaluation methods can be categorized into two groups namely the automated evaluation and the human supported evaluation [12]. Human Evaluation is depends on the knowledge of the language professionals affiliated with source and target languages [49]. From a practical point of view, manual evaluation, performed by translation experts, is expensive and takes time. What is needed are automatic metrics that are quick and cheap to use and approximate human judgements accurately. Automatic metrics, used in the MT community, are Bilingual Evaluation Understudy (BLEU), Metric for Evaluation of Translation with Explicit Ordering (METEOR), General Text Matcher (GTM), Translation Edit Rate (TER), Perplexity (PP) and etc. Automatic metrics need reference translations because they compare the MT output with reference translations and provide comparison scores. If reference translations are available, these metrics can be used to evaluate the output of any number of systems quickly, without the need for human intervention [12], [50]. We use automatic and human evaluation methods. Frist we will use BLEU and PP evaluation measures for checking the effectiveness of our proposed models.

Besides, we will do side-by-side human evaluation by linguistic raters, who evaluate the usability of prototype and compare the quality of translations predicted by the proposed model.

2.5.1 Bilingual Evaluation Understudy (BLEU)

BLEU is an automated, straightforward, low cost and language-independent evaluation method that correlates highly with human evaluation [12], [50]. The primary programing in BLEU implementer is to match the n-grams of candidate with n-grams of reference without considering the position of word. BLEU was one of the metrics to achieve a high correlation with reference translation and remains one of the most popular automated and inexpensive metrics used in different researches for evaluation purpose. BLEU uses a modified form of precision to compare a candidate translation against multiple reference translations. BP will be 1.0 when the candidate translation length is the same as any reference translation length. The closest reference sentence length is the best match length. With the BP, we see that a high-scoring candidate translation will match the reference translations in length, in word choice, and word order. BP is an exponential decay and is calculated as shown below:

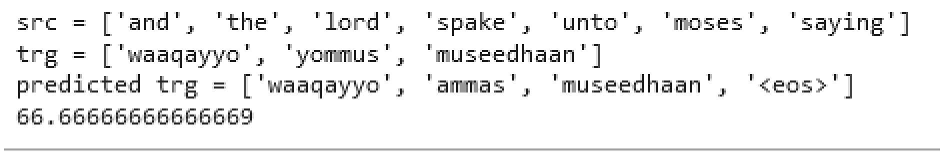
$$BP = \begin{cases} 1 & \text{if } c > \mathrm{r} \\ e^{(1-r/c)} & \text{if } c \leq \mathrm{r} \end{cases}$$

[Brevity penalty] Equation 2:2

As shown in Equation 2.2, r- count of words in a reference translation and c- count of words in a candidate translation. Finally, calculate BLEU. Then, the BLEU score is calculated by first computing the geometric average of the modified n-gram precisions, P_n , using n-grams up to length N and positive weights w_n summing to one. The basic BLEU metric is then determined as shown in Equation 2.3.

$BLEU = BP.exp(\sum_{n=1}^{N} w_n \log p_n)$	[BLEU score] Equation 2:3

For example, where 3 is total length of output sentence (predicted trg) and 2 tokens are matched with reference sentence (target sentence (trg)), then 2/3 = 0.66666, BLEU score is 0.66666*100= 66.666%



```
src = ['and', 'the', 'lord', 'spake', 'unto', 'moses', 'saying']
trg = ['waaqayyo', 'yommus', 'museedhaan']
predicted trg = ['waaqayyo', 'ammas', 'museedhaan', '<eos>']
66.66666666666669
```

Hence, BLEU has been shown as a strong correlation with human evaluation. This is also the reason behind to select BLUE evaluation metric.

2.5.2 Perplexity (PP)

In the context of NLP, Perplexity (PP) is one way to evaluate language models [51], [52]. It is a measurement of how well a probability model predicts a sample. A commonly used quality measure for a given model M is related to the entropy of the underlying source and was introduced under the name of perplexity [52]. Assuming that a language model is a probability matrix between a word and the next word that occurs in the corpus of the training set, PP is the probability of the test dataset, normalized by the number of words [51]. Minimizing perplexity is the same as maximizing probability. A low perplexity indicates the probability distribution is good at predicting the sample. In addition to cross-entropy loss, another good metric we can use is perplexity, which is a measurement of the models ability to predict the target translation. This can be calculated as following equations. The lower the perplexity, the better the model is at predicting the target translation.

$$P(X) = \prod_{i=o}^{t} P\,(x_i|x_{<i})$$

Probability of a sequence $X_i \rightarrow$ i the word in test set Equation 2:4

$$CE(X) = -\frac{1}{t}\log \mathrm{P}(X)$$

$\rightarrow$ Cross-Entropy Equation 2:5

$$PP(X) = e^{CE(X)} = e^{-\frac{1}{t}\sum_{i=0}^{t}\log p(x_i|x_{<i})}$$

Perplexity $X_i \rightarrow$ i the word in test set Equation 2:6

2.6 Artificial Neural Network

An Artificial Neural Network (ANN) is a distributed knowledge treatment system in which performance essentials are alike to the human brain, and is based on a simulated biological neural network [53], [54]. It is also known as a Feed Forward Neural Network (FFNN) because inputs are processed only in the forward direction. Each NN has three layers, namely, input, hidden, and output. The input layer is a layer for providing data provided as inputs to the network. The output layer contains values predicted by the network. The hidden layer is the data analysis location. Usually, the number of selected neurons of the layers is obtained by trial and error. The general architecture of the ANN is displayed in figure 2:6 where X (x_1, x_2, ..., x_n) = inputs vector, W= connecting weights to the next layer, b_k = bias, and Y_k is the ANN final output. Weight is the parameter within a NN that transforms input data within the network's hidden layers. Within each node is a set of inputs, weight, and a bias value. As an input enters the node, it gets multiplied by a weight value and the resulting output is either observed, or passed to the next layer in the NN. Biases make up the difference between the function's output and its intended output.

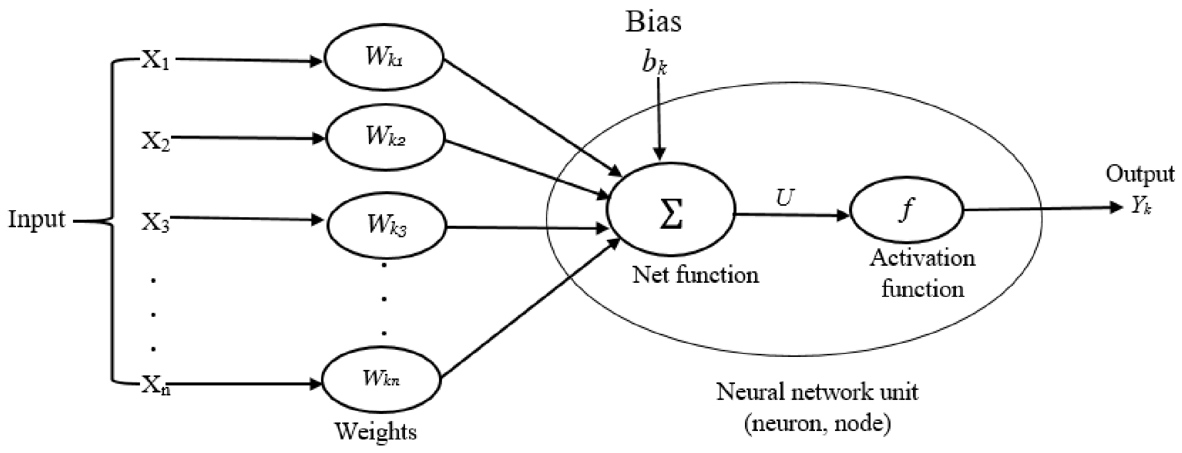


Figure 2:4 General Structure of a neural network

In Figure 2:6, n inputs are given from x_1 to x_n to the counterpart weights W_{k1} to *Wkn*. Initially, the weights are multiplied by their inputs, and then they are summed with the amount of bias to obtain

$U = \sum_{i=1}^{n} w_{kn} x_n + b_k = w_1x_1 + w_2x_2 + \cdots w_{kn}x_n + b_k \rightarrow (b = 1)$ always *Equation 2:7*

The activation function converts input signals into output signals. That mean we apply the activation function on the summation results, the final output value is obtained as Y_k of the neuron.

$Y_k = f(U) = \sum x_i w_i + Bias$ → Equation 2:8

The main role of the activation function is to normalize the input sequence. Activation Function are multiple types like Sigmoid Logistic Activation Function (sigmoid), Hyperbolic Tangent Activation Function (Tanh) and Rectified Linear Unit (ReLu) activation Function. Vanishing gradient problem mostly occurs during the backpropagation when the value of the weights are changed. This causes them to make the learning process very slow, and make them converge to their optimum. We use ReLu activation function in our proposed systems, because it is the best and most advanced activation function right now compared to the sigmoid and Tanh because all the drawbacks like Vanishing Gradient Problem is completely removed in this activation function which makes this activation function more advanced compare to other activation function and also Speed is fast compare to other activation function [55]. Generally according to [55], when to compare ReLu activation function to other activation function, it have the as following table 2:2:

Table 2:2 Comparison of activation functions

Activation function	Range	Vanishing gradient problem	Nature	Equation
sigmoid	0 *to* 1	Yes	Non linear	$f(x) = \frac{1}{1+e^{-x}}$ Equation 2:9
Tanh	-1 *to* 1	Yes	Non linear	$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}}$ Equation 2:10
ReLu	0 *to* infinity	No	Linear	$f(x) = \{o\ for\ x < 0$ $x\ for\ x \geq 0\}$ Equation 2:11

To overcome the limitation of ANN different types of NN architectures are developed under Deep Neural Network (DNN), such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Attention mechanism and Transformers.

2.6.1 Deep neural network (DNN)

DL, is also called DNN, which uses the NN to analyze different factors with a structure that is similar to the human neural system [56]. NN are a specific set of algorithms that has revolutionized the field of ML. These DNN are made of multiple layers. The first layer is called an Input layer, the last layer is called an output layer, and all layers between these two layers are called hidden layers. In the DNN, there are multiple hidden layers, and each layer is composed of neurons. These neurons are connected in each layer. The input layer receives input data, and the neurons propagate the input signal to its above layers. The hidden layers perform mathematical operations on inputs, and the performed data forwarded to the output layer. The output layer returns the output to the user. A network with two or more hidden layers is a DNN or DL. The more hidden layers the more the model learns better and predicts correct translation [12], [57].

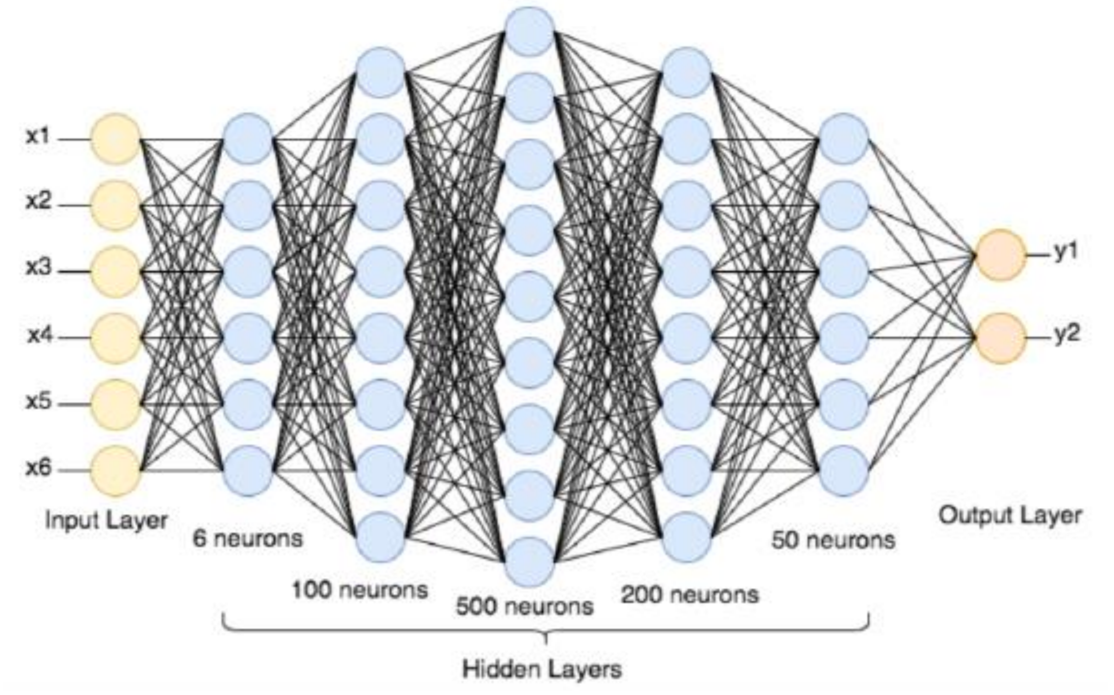


Figure 2:5 Deep Neural Network architecture *[58]*

There are a variety of DL architectures used by researchers and engineers, and each of the different architectures has its own specialty use case. DL methods employ multiple processing layers to learn hierarchical representations of data, and have produced state of the art results in many domains. Recently, a variety of model designs and methods have blossomed in the context of NLP [59]. Let as discus in the following section.

2.6.1.1 Convolution Neural Network (CNN)

One of the most popular DNN is CNN. Since the 1950s, the early days of AI, researchers have struggled to make a system that can understand visual data [59]. In 2012, computer vision took a quantum leap when a group of researchers from the University of Toronto developed an AI model that surpassed the best image recognition algorithms and that too by a large margin [37], [60]. The layers of a CNN consist of an input layer, an output layer and a hidden layer that includes multiple convolutional layers, pooling layers, fully connected layers and normalization layers. The removal of limitations and increase in efficiency for image processing results in a system that is far more effective, simpler to trains limited for image processing and NLP. Though CNN were introduced to solve problems related to image data, they perform impressively on sequential inputs as well. CNNs tend to start with an input "scanner" which is not intended to parse all the training data at once. This input data is then fed through convolutional layers instead of normal layers, where not all nodes are connected to all nodes. Each node only concerns itself with close neighboring cells. These convolutional layers also tend to shrink as they become deeper, mostly by easily divisible factors of the input. The important thing to remember about any deep learning model is that it requires a large amount of data to train and also requires a lot of computing resources. This was a major drawback for CNNs at that period and hence CNNs were only limited to the postal sectors and it failed to enter the world of ML.

2.6.1.2 Recurrent Neural Network (RNN)

In single layer NN, information flows in one direction only. These type of networks are also called feed-forward since information flows in forward direction from input layer to output layer and there is no cycles or loops in them [61]. The term "recurrent" applies as they perform the same task over each instance of the sequence such that the output is dependent on the previous computations and results. This template is naturally suited for many NLP tasks such as language modeling, MT, speech recognition and image captioning [62]. Moreover, RNNs are powerful models for sequential data [62], and they use the previous output to predict the next output.

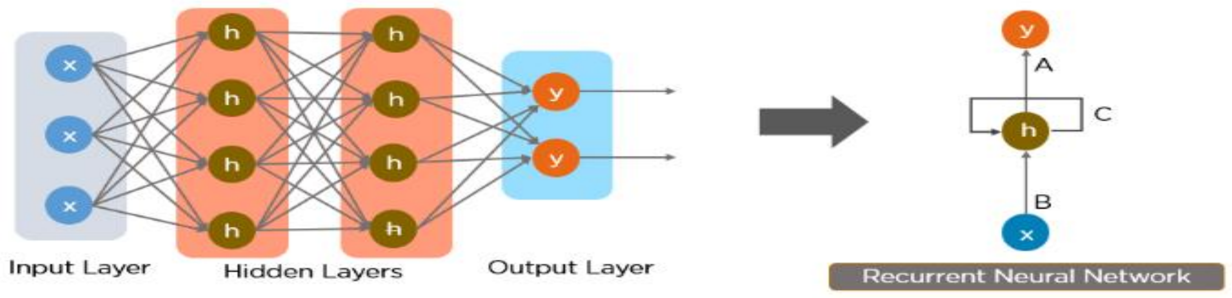


Figure 2:6 Simple Recurrent Neural Network *[64]*

A simple recurrent network has only one internal memory h_t which is computed from Equation 2:12:

$$h_t = g(Wx_t + U_f h_{t-1} + b) \qquad \textit{Equation 2:12}$$

Where *g()* denotes an activation function, *U* and *W* are flexible weight matrices of the *h* layer, *b* is a bias, and *X* is an input vector [64]. Training a RNN seems simple since we have just a set of weight matrices, however, it is extremely hard due to its recurrent connections. We can analyze and understand the reason of this hardness by using the tool of back propagation and chain rule. For instance, as we multiply all the weight matrices in forward propagation we need to do the same in the back propagation. As we go backward, the signal may become too strong or too weak; this is the gradient exploding or vanishing problem, respectively [65]. In a network of *n* hidden layers, *n* derivatives will be multiplied together. If the derivatives are large then the gradient will increase exponentially as we propagate down the model until they eventually explode, and this is what we call the problem of exploding gradient. Alternatively, if the derivatives are small then the gradient will decrease exponentially as we propagate through the model until it eventually vanishes, and this is the vanishing gradient problem. We can overpass the problem of exploding or vanishing gradients by using special RNN architectures with leaky units such as LSTM and GRU [66].

2.6.1.3 Long Short-Term Memory (LSTM)

LSTM networks are a type of RNNs, it was created to fix the vanishing gradient problem [66]. It has internal mechanism called gates, that can regulate the flow of information into and out of the cell [66]. The greatest feature of LSTM is in its capability to learn long-term dependency, which is not possible with simple RNNs. The memory cell is responsible for passing long term dependencies as well as information from earlier inputs in the sentences to predict later ones. To predict the next step, the weight values on the network have to be updated, which requires the maintenance of information from the initial steps. LSTM has three gates: input, forget, and output.

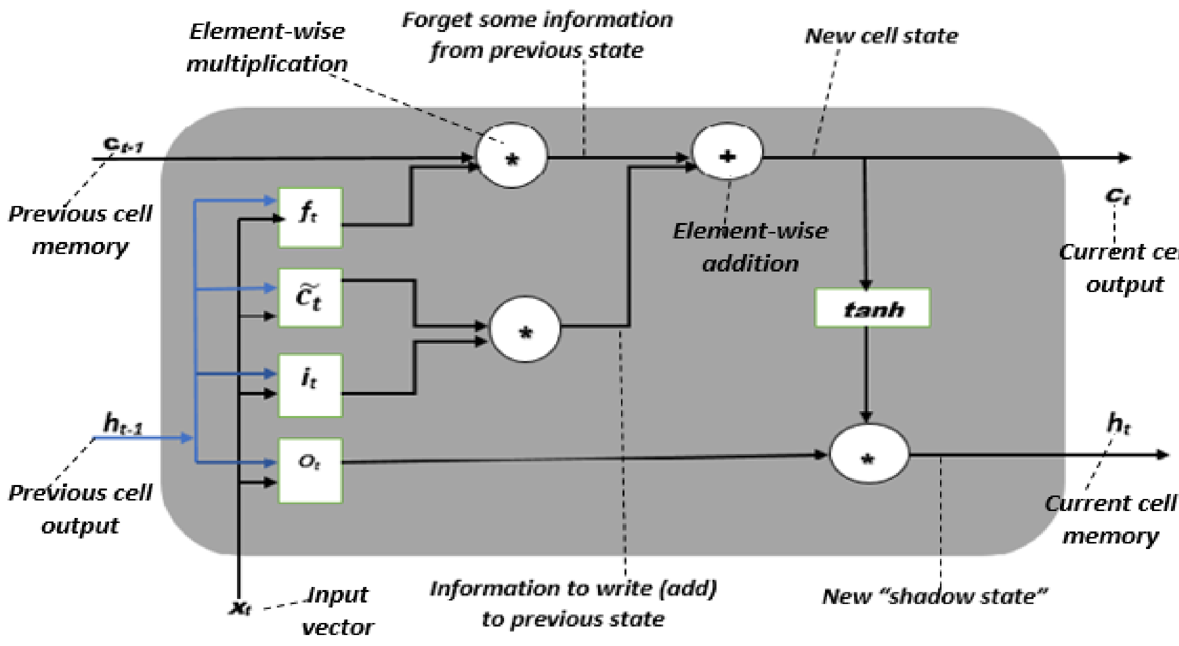


Figure 2:7 The architecture of LSTM cell unit

For LSTM, the hidden state h_t is computed as follows: where i_t, f_t, and O_t are the input, forget, and output gates at time t, respectively; W_i, W_f, W_o, and W_c are weights that map the hidden layer input to the three gates of input, forget, and output while U_i, U_f, U_o, and U_c weights matrices map the hidden layer output to gates; b_i, b_f, b_o, and b_c are vectors. Moreover, C_t and h_t are the outcome of the cell and the outcome of the layer, respectively [66].

Input gate $\rightarrow$	$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$	*Equation 2:13*
Forget gate $\rightarrow$	$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$	*Equation 2:14*
Output gate $\rightarrow$	$o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o)$	*Equation 2:15*
Memory cell candidate $\rightarrow$	$\tilde{c}_t = tanh(W_c h_{t-1} + U_c x_t + b_c)$	*Equation 2:16*
Memory cell $\rightarrow$	$C_t = f_t * c_{t-1} + i_t * \tilde{c}_t$	*Equation 2:17*
hidden state (*Cell output*) $\rightarrow Y_t = h_t$	$h_t = o_t * tanh(c_t)$	*Equation 2:18*

Memory cell candidate is used for computing current gate and last gate. The forget gate is embedded to indicate how much the previous memory remembers and how much it has forgotten. Then, the input gate decodes the data to be added from the present step. Finally, the output gate determines the next hidden state.

2.6.1.4 Gated Recurrent Unit (GRU)

GRU is a somewhat advanced type of RNN suggested by Cho et al. [66]. The variance with LSTM is that GRU merges the input and forget gates and converts them with an update gate. GRU including to two gates that is Reset gates and Update gates. Reset gate is adjusts the incorporation of new input with the previous memory and an update gate that controls the preservation of the precious memory are introduced [66], [67]. The reset gate and the update gate adaptively control how much each hidden unit remembers or forgets while generating a sequence. According to [45], [66] GRU is utilizes an update and a reset gates to avoid the issue of the vanishing gradient that penalizes the standard RNN as well as solving long term dependencies to a great extent.

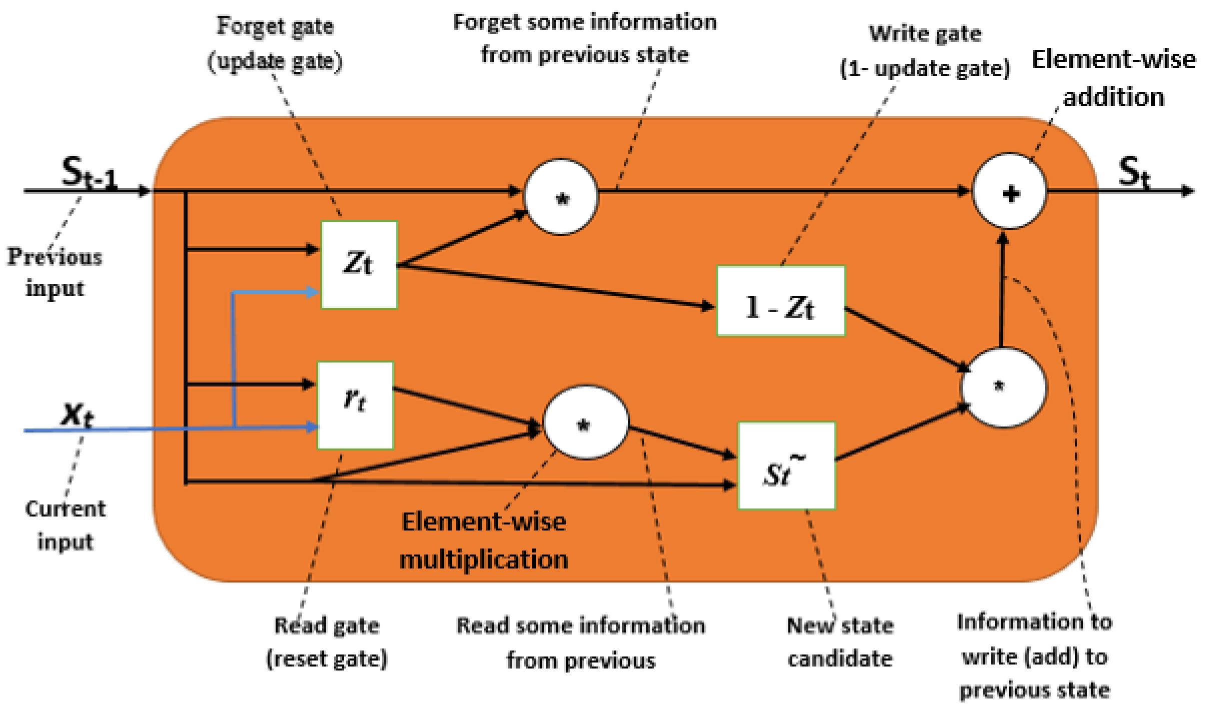


Figure 2:8 The architecture of GRU cell unit

Therefore, GRU has fewer parameters than LSTM which makes training easier. For GRU, the output value of h_t is computed as follows:

$r_t= \sigma(W_rS_{t-1} +U_rx_t + b_r)$ →*Read gate/reset* *Equation 2:19*

$\boldsymbol{z_t} = \sigma(W_zS_{t-1} +U_zx_t + b_z)$ → *Forget gate* Equation 2:20

$\tilde{s}_t = tanh(W_s(r_t *s_{t-1})+ U_sx_t + b_s)$ →*Sate candidate* *Equation 2:21*

$S_t = z_t * S_{t-1} + (1 – z_t) * \tilde{s}_t$ →*Current state* *Equation 2:22*

Where *r* is a reset gate, and *z* denotes an update gate. The reset gate indicates how the new input can be combined with earlier memory. The update gate also indicates how much previous memory is kept. If the update gate is 1, the previous memory is fully preserved, and if it is 0, the previous memory is completely forgotten. There is a forget gate in the LSTM that automatically determines how much of the previous memory is maintained whereas, in GRU, all previous memories are maintained or completely forgotten.

2.6.1.5 Attention Mechanism

Attention mechanism allows a neural network to pay attention only part of an input sentence while it's generating a translation [68], [69]. The goal is to break down complicated tasks into smaller areas of attention that are processed sequentially. Similar to how the human mind solves a new problem by dividing it into simpler tasks and solving them one by one. NN can achieve this same behavior using attention, focusing on part of a subset of the information they are given [69]. This is due to the fact that Attention was introduced to address the problem of long sequences in MT. The attention mechanism was introduced by Bahdanau, Cho, and Bengio in the context of MT [4], [70]. By implementing the attention mechanism, the quality of the translation could be improved, mainly when translating longer sequences of text. Other suggestions are made by Luong, Pham, and Manning, in which the dot product between the decoder hidden state and the decoder states is used as the alignment model [4], [70]. At every decoding step, the decoder will be informed how much attention needs to be paid to each input word using a set of attention weights. These attention weights provide contextual information to the decoder for translation.

Bahdanau et al. [4] uses the concatenation of the forward and backward hidden states in the bi-directional encoder and previous target's hidden states in their non-stacking unidirectional decoder. Loung et al. [69] attention uses hidden states at the top layers in both the encoder and decoder. Luong attention mechanism uses the current decoder's hidden state to compute the alignment vector, whereas Bahdanau uses the output of the previous time step. We use attention mechanism in our proposed models in order to enhance the translation quality within long sentence regarding to our domain.

2.6.1.6 Transformers

Transformers were originally introduced by researchers at Google in the 2017 NIPS paper *Attention is All You Need* [68]. Transformers are powerful deep learning models that can be used for a wide variety of NLP tasks. A transformer has two main segments, the first is an encoder that operates primarily on the input sequence and the second is a decoder that operates on the target output sequence during training and predicts the next item in the sequence. Figure 2:9 demonstrates how a transformer is put together, with the encoder on the left and the decoder on the right.

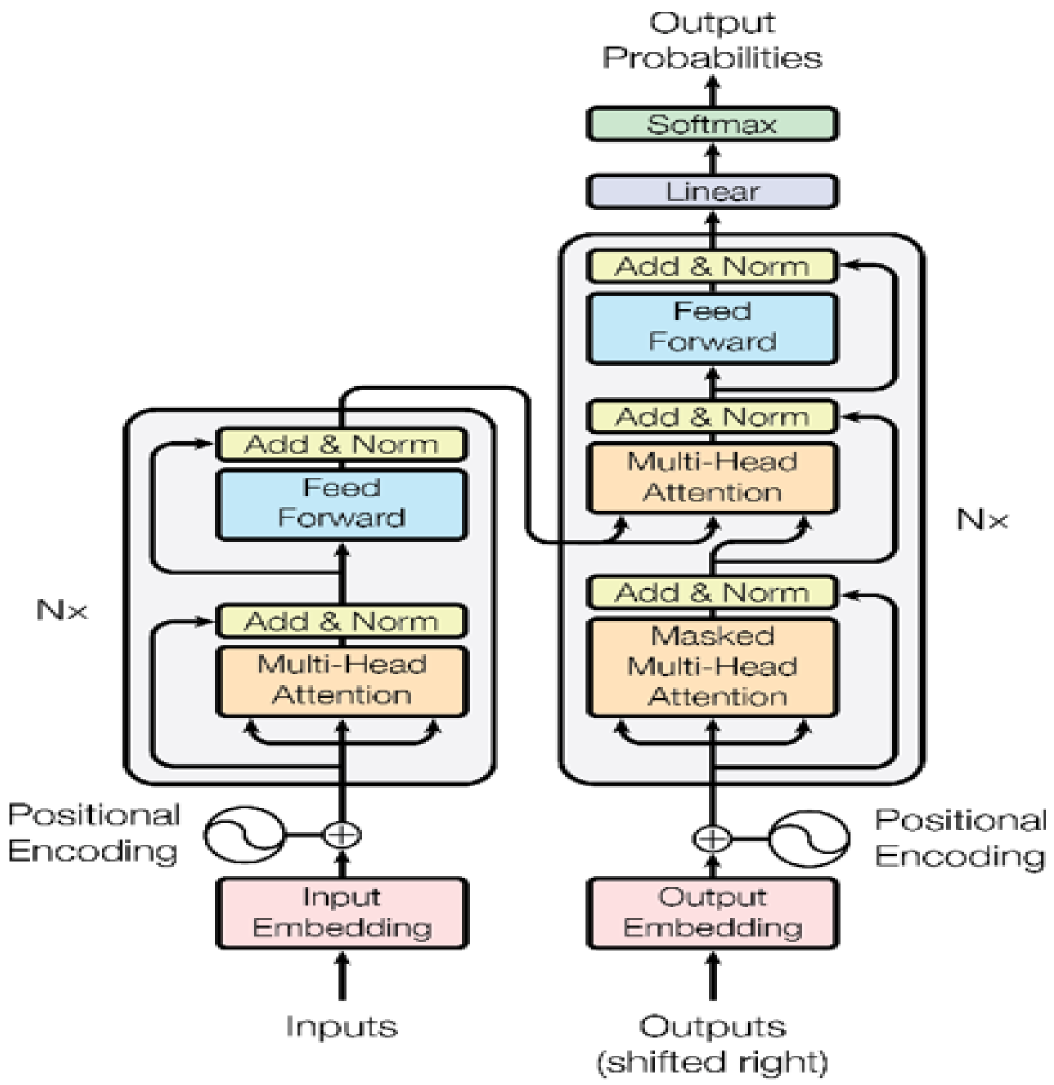


Figure 2:9 Architecture of Transformer Model adopted from *[68]*

We implement transformer, LSTM and GRU models as we proposed and fourthly we discussed how our proposed models work under chapter four.

2.7 Related Work

This section presents researches that have made effort so far concerning machine translation using different techniques. The following researches consider a specific kind of environment as their problem area. Previous works, such as thesis and other online publications are systematically reviewed in-order to understand the domain and the advancements. Let as seen some research paper on MT system for Ethiopian and foreign country language pairs.

Machine Translation Systems for foreign country (Non-Ethiopian) language pairs

In 2017, *.Ashish Vaswani et al*. [68], conducted their study on Attention is all you need area. The authors described that, the dominant sequence transduction models are based on complex recurrent or CNN that include an encoder and a decoder. According to the authors, the best performing models also connect the encoder and decoder through an attention mechanism. The authors proposed a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. As stated by the authors experiments on two MT tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train.

The model developed by the authors achieves 28.4 BLEU on the WMT 2014 English to German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English to French translation task, their model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. The authors stated that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data. Focuses on the WMT 2014 English to German translation and On the WMT 2014 English to French translation.

In 2020, Ariel and Wenhan [6], proposed Russian to English bidirectional MT system. Their review depicts their submission to the WMT20 shared news translation task. WMT is the conference to assess the level of MT capabilities of organizations in the word. The authors participated in one language pair and two language directions, from Russian to English and from English to Russian. They used official training data, 102 million parallel corpora and 10 million monolingual corpora. Their baseline systems are Transformer models trained with the Sockeye sequence modeling toolkit, supplemented by bi-text data filtering schemes, back-translations, reordering and other related processing methods. The BLEU value of their translation result from Russian to English is 35.7, ranking 5th, while from English to Russian is 39.8, ranking 2th.

In 2020, Debajit Datta et al. [73], conducted their research study on NMT using RNN. The authors said that in this era of globalization, it is quite likely to come across people or community who do not share the same language for communication as us. MT are being developed to acknowledge the problems caused by this. Several NMT have been developed in this regard, but RNN, on the other hand, has not grown much in this field. In their work, the authors have tried to bring RNN in the field of MT, in order to acknowledge the benefits of RNN over ANN. The results show how RNN is able to perform MT with proper accuracy. Focuses on designing RNN that translates English to French by using python library on Google Colab.

The system built by the authors, combines the models of RNN giving the highest accuracies within ten epochs. It has been observed that taking multiple models at a time resulted in a better model giving higher accuracy for the system as a whole. It has also been observed that as individual models, RNN with embedding gives the maximum accuracy with further iterations, followed by bidirectional RNN. According to the authors, the model must be chosen smartly, and while combining, it should be taken care that the system must not result to overfitting issues and other overheads, as they will lead to worse accuracy and an inefficient system for MT using neural networks. According to the results observed from the experiments done, bidirectional RNN with encoding algorithm gives a high accuracy compared to that of the given models of RNN. Single RNN models like the basic RNN, only RNN with encoding, only bidirectional RNN and RNN with encoder and decoder gives lesser accuracy comparatively. The authors further described that, the work can be extended for a larger dataset consisting of even bigger text corpus, so that the system becomes ideal for translating every possible sentence instead of just a few sentences. Moreover, the possible combination of models can be broadened into other models of RNN as well, considering all possible combinations within them, so that the systems can be made further accurate.

Machine Translation Systems for Ethiopian language pairs

Yitayew Solomon [5], conducted his research study on optimal Alignment for Bi-directional Afaan Oromo to English SMT. As defined by the author, SMT is an approach that mainly use parallel corpus for translation, in which parallel corpus alignment of the given corpus is crucial point to have better translation performance. The author described that alignment quality is a common problem for SMT because, if sentences are miss aligned the performance of the translation processes becomes poor. The author’s study aims to explore the effect of word level, phrase level and sentence level alignment on bi- Directional Afaan Oromo-English SMT.

To conduct the study, the author collected corpus from different sources such as criminal code, FDRE constitution, Megleta Oromia and Holly Bible. In order to make the corpus suitable for the system, he applied different preprocessing tasks applied such as true casing, sentence splitting and sentence merging has been done. A total of 6400 simple and complex sentences are used in order to train and test the system. The author used 9:1 ratio for training and testing respectively. For language model he used 19300 monolingual sentences for English and 12200 for Afaan Oromo.
For the purpose of the system, he used Mosses for Mere Mortal for translation process, MGIZA++, Anymalign and hunalign tools for alignment and IRSTLM for language model. After he prepared the corpus, he conducted different experiments. Experiment results shows that better performance of 47% and 27% BLUE score was registered using phrase level alignment with max phrase length 16 from Afaan Oromo-English and from English to Afaan Oromo translation, respectively. This depicts an improvement of on the average 37% accuracy registered in this study. The reason for this score is length of phrase level aligned corpus handle word correspondence. This depicts that alignment has a great effect on the accuracy and quality of SMT from Afaan Oromo-English and the reverse. The author further described that, during MT alignment of a text of multiple language have different correspondence, one-one, one-many, many-one and many-many alignments. In his study, the author stated that, many-many alignments are a major challenge at phrase level that needs further investigation.

Yeabsira Asefa [24], conducted his research on Context Based Machine Translation (CBMT) with RNN for English to Amharic Translation. As the author said, capturing context in translating between two human languages using computing machines is challenging. It is more challenging when the languages differ greatly in grammar and have small parallel corpus like the English to Amharic pair. The author said that, the current approaches for English to Amharic machine translation usually require large set of parallel corpora in order to achieve fluency as in the case of SMT and EBMT. The context awareness of PBMT approaches used for the pair so far are also questionable.

The author develops a system that translates English text to Amharic text using a combination of CBMT and a RNNMT. He built a bilingual dictionary for the CBMT system to use along with a target corpus. The RNNMT model has then been provided with the output of the CBMT and a parallel corpus for training. The author evaluated his proposed approach using the New Testament

Bible as a corpus. The result shows that the combinational approach on English to Amharic language pair yields a performance improvement of 2.805 BLEU scores on average over basic NMT.

Arfaso Birhanu, [25] conducted his research study on English-Afan Oromo MT using CNN. The author defined MT as an automatic translation of a source language to a target language. In his work, the author proposed, a bi-directional text-based MT for English and Afaan Oromo language pairs using CNN. The author started his work with the objective of improving the previous work on English to Afaan Oromo MT by making the translation bi-directional through the application of CNN on translations between these language pair. In order to achieve his objective, the author collected parallel corpus from different sources and divided them into training and testing sets. He used 80% of total dataset for training and 20% of total dataset for testing. He implemented three systems where the first system uses a word-based statistical approach that used as a baseline, while the second system with RNN approach is used as a competitive model and lastly, the third system uses CNN for the bi-directional translation between Afaan Oromo and English Languages.

The author stated that, after training and testing these systems on corresponding training and testing datasets, the CNN achieved 3.86 BLEU score improvement on translation from English to Afaan Oromo and 3.32 BLEU score on translation from Afaan Oromo to English translation than baseline system. Besides, CNN approach has shown an improvement of 1.58 BLEU score on translation from English to Afaan Oromo and 1.51 BLEU score on translation from Afaan Oromo to English translation than RNN approach. The CNN approach is faster on training than RNN approach.

Dereje Seifu [7], conducted his research study on hybrid artificial neural machine translation using deep learning techniques English-to-Afaan Oromo. The author described that, artificial neural networks are at the basis of the most state-of-the-art for a variety of activities and have enjoyed great success in a variety of tasks such as MT, image recognition, speech recognition, text summarization, and in another computation. According to the author, NN with recurrent model so-called NMT, have recently been applied to MT and begun to show successive results. However, as a recently appeared method, the approach has some restrictions. A NMT system typically has to apply a vocabulary of some size to prevent the time-usage in training and decoding; consequently, it produces a critical out of vocabulary difficulty. Moreover, the decoder lacks a mechanism to assurance all the source words to be translated and usually resulted short translations, resulting in influent but insufficient translations.

The author said that, the main aim of this research is to design a model that significantly reduces translation problems shown in the NMT system. He proposed a model that is implemented by classifying the model into three layers namely; encoder layer, hybrid layer and decoder layer. His experiment result shows that the proposed method significantly improved the translation quality of the state-of-the-art NMT system on English to Afaan Oromo translation tasks prepared dataset. The problems; out-of-vocabulary lead to unknown word output, lack of translation problem for long sentences, lack of decoding mechanism, which badly hurt the translation quality in the neural machine translation system is reduced by hybrid neural machine translation model proposed by the author. His approach resulted in a gain of up to 2.4 BLEU score on test sets. The BLEU score evaluation of the translations obtained with this combination increased, and this also results in improved translation quality as observed in our experiments. His experiments on English to Afaan Oromo translation tasks show that his system achieves significant improvements over the reference on a small amount of the training dataset collected from the web. The proposed method achieved better performance compared with the previous models (RNN, and SMT).

Summary of Related Works

There are many kinds of research done on MT in different language pairs. As we mentioned above do their part and suggested many further works to be done. The summary of related works on MT systems for Ethiopian language pairs shown in table 2.3 underneath.

Table 2:3. Summary of related works

No	Authors (Year)	Title	Approach	Corpus size used	Focus	Tool	Result	Remark
1	Yitayew Solomon (2017)	Optimal Alignment for Bi-directional Afaan Oromo-English SMT.	SMT	6400 parallel corpus. He used 9:1 ratio for training and testing. For language model he used 19300 monolingual sentence for English and 12200 for Afaan Oromo.	The author's study aims to explore the effect of word level, phrase level and sentence level alignment	Moses tools such as IRSTLM for language model, MGIZA++ for word alignment	Performance of 47% and 27% BLUE score was achieved from Afaan Oromo-English and from English-Afaan Oromo translation result.	Small parallel corpus size and he used phrase level alignment with max phrase length 16
2	Yeabsira Asefa (2020)	CBMT with RNN for English to Amharic Translation.	Combine a RNNMT and CBMT	New Testament of the Bible is used as a corpus providing a total of 8603 phrases and sentences with a random 80% training and 20% test dataset split.	Focuses on building a translation system from English to Amharic based on context of the text with a good accuracy.	Python programming language and its libraries, such as NLTK and Tensorflow library.	The CBMT without being provided the target flooded data has performed better on average by 14.23 BLEU points over the NMT (18.3% for Romans and 10.16% for Mark).	He used small and domain-specific dataset.
3	Arfaso Birhanu (2019)	Bi-Directional English to Afaan Oromo	Deep learning (CNN)	5550 sentences of parallel corpus from religious books, some published conversational books, data of online available conversational	Implementation and performance analyzing of bidirectional machine translation	Python programming language and its libraries	Performance of model with BLEU score shows 24.37 for translation from English to Afaan Oromo and 23.18	Small number of dataset and also does not show the method to

		MT using CNN		sentences, Oromia regional revenue and health sector. he used 8:2 ratio for training and testing	between English and Afaan Oromo		from Afaan Oromo to English translations	displayed the Effect of long sentence
4	Dereje Seifu (2019)	Hybrid ANN using deep learning technique s English to Afaan Oromo	SMT, LSTM and HNMT.	He used 13300 parallel corpus, collected from religious web documents, FDRE constitution, FDRE criminal code, Council of Oromia regional state, and English-Afaan Oromo language education materials, from the prepared, he used 7:3 ratio for training and testing	Focuses on designing a model that significantly reduces translation problems shown in the NMT systems	Python programmin g language with different python library,	The approach resulted in a gain of up to 2.4 BLEU score on test sets English-to-Afaan Oromo translation	Small dataset size and Low BLEU score
5	This work	Bi-Direction al English to Afaan Oromo NMT	Deep learning (LSTM, GRU and Transformer)	We used 21323 parallel corpus, from the prepared dataset (80% for training, validation 10% and 10% for testing).	We focuses on comparison of the models and explore the challenge across our domain	Python programmin g language with different python library	We discussed under result discussion section	We will use small dataset when to compare with foreign language

CHAPTER THREE

OVERVIEW OF ENGLISH AND AFAAN OROMO LANGUAGES

This chapter briefly discusses an overview of English and Afaan Oromo language. We will cover the scripting, language morphology and word formation process in Afaan Oromo language. Finally, we will present the sentence structures, the articles and punctuation marks of both English and Afaan Oromo languages are describe in this chapter.

3.1 The English Language

English is the international communication language [74]. It worldwide, there are over 400 million native speakers of English, and over one billion people speak it as a second language. English is probably the third language in terms of number of native speakers (after Mandarin and Spanish); and probably the most widely spoken language on the planet taking into account native and non-native speakers [74]. It is the world's most widely-used language in international business and telecommunications, newspaper and book publishing, scientific publishing, mass entertainment and diplomacy. English language uses a writing system based on the classical Latin or Roman alphabet and contains 26 letters with 5 vowels. The English language has eight parts of speech such as Noun, Adjective, Pronoun, Verb, Adverb, Preposition, conjunction and Interjection [5].

3.2 A Brief Overview of Afaan Oromo

Afaan Oromo (literally Oromo mouth) or the Oromo language, belongs to the Cushitic branch of Afro-Asiatic language phylum. It is the language of the ancient Cush that was spoken all over East Africa [25], [75]. It is the third most widely spoken language in Africa after Arabic and Hausa [5]. The language is widely spoken in Ethiopia and neighboring countries like Kenya, Somalia Sudan and etc. Afaan Oromo is an official language of Oromia Regional State and also in Oromiya Zone of the Amhara Region and using the Afaan Oromo language in teaching and learning processes for primary, junior secondary schools of the region, teacher training institutions and colleges and it is a field of study in higher education institutions. It is also used in courts, media, religious organization and for administration [5]. Afan Oromo has its own phonetic language, which means that it is spoken in the way it is written. It uses simple Latin script, which makes it straightforward in its written system. It is a common mother tongue for Oromo people, who are the largest ethnic group in Ethiopia, at 32.1% of the population according to the 1994 census[25], [76]. Although it is difficult to identify the actual number of Afaan Oromo speakers (as a mother tongue) due to lack appropriate and current information sources. Even if the language is spoken by large number of the population, the number of literature works, newspapers, magazines, educational resources, official documents and religious writings written and published in this language are few in number.

Geographically, Afan Oromo is found in Northern, Eastern, Southern, Central and Western Cushitic Language families by retaining its homogeneity. The Oromo of all these areas could communicate in this language without dialectical barriers [77]–[79]. Afaan Oromo has five major dialects: Rayya (Northern), Borana (Southern), Tulama (Central), Harar (Eastern), and Mecha (Western). For instance, the Mecha dialect predominantly uses ***Koo*** to mean 'my, mine', whereas other dialects use ***kiyya***. The Mecha dialect uses ***dhara*** to mean '*lie*', whereas Borana ***kijiba, soba***. Despite these occurrences of word variations and some differences in pronunciation, it needs to be underlined that all Oromo varieties are mutually intelligible, and all Oromo people understand one another without great difficulty.

3.2.1 Afaan Oromo Scripting (Qubee Afaan Oromoo)

Before 1991, Ge'ez script was used for writing Afaan Oromo documents. A Latin-based alphabet called Qubee has been adopted and became the official script of Afan Oromo since 1991. It has its own consonants and vowels in addition it has double consonants. Afaan Oromo has 34 letters called 'qubee'. five of them known as vowels (dubbachiiftuu), the other seven letters are known as double consonants (Qubee dachaa), and the rest are known as consonants (22) (dubbifamaa) [7], [80]. It typified by capital and small letters that are familiars in the English writing system. Double consonant letters derived from a combination sequence of two consonant letters. Vowels in Afaan Oromo formulated as long and short vowels. The five letters (three from consonant letters and two from double consonants) to write the borrowed foreign words. The first letters P, TS, V, Z, fi ZY in Afaan Oromo there is no root words formulated from those letters. But, in writing the language, they are scripted to refer to foreign words such as "Paappaayyaa", Zeeytuuna (guava), 'Poolisii or foolisii' (police), 'Tsahaay or Xahaay' (noun), ''Telefishiina or Telebishiinii' (Television), 'Zayiita' (oil) etc.

Table 3:1. This is the complete list of the Qubee letters (Afaan Oromo alphabet)

No	Capital Letter (Qubee Gurguddaa)	Small Letter (Qubee Xixiqqaa)	Type	Long	Short	Sound relative to English sound
1	A	a	**Vowel**	Aa	A	*as in apple* in long *as in far* in short
2	B	b	Consonant			*as in bad, **b**ody, a**b**out*
3	C	c	Consonant			*tch sound*
4	D	d	Consonant			*as in dad*
5	E	e	**Vowel**	Ee	E	*as in eight* in long *as in p**e**n or **e**mpty in short*
6	F	f	Consonant			*as in **f**ive or a**f**ter*
7	G	g	Consonant			as in **g**ame or a**g**o
8	H	h	Consonant			as in **h**ammer
9	I	i	**Vowel**	Ii	I	*as in hip* in long as in h**i**t or **i**n in short
10	J	j	Consonant			as in **j**ump or a**g**ency
11	K	k	Consonant			as in **c**o**c**o
12	L	l	Consonant			as in **l**itt**l**e
13	M	m	Consonant			as in **mem**ber or mam
14	N	n	Consonant			*as in narrow*
15	O	o	**Vowel**	Oo	O	as in s**o**re or **o**pen
16	P	p	Consonant			*as in past*
17	Q	q	Consonant			*Glottal sound k*
18	R	r	Consonant			*as in rat*
19	S	s	Consonant			*as in sun*
20	T	t	Consonant			*as in task*
21	U	u	**Vowel**	Uu	U	*As in book* **in long** *As in bull* in short
22	V	v	Consonant			*as in value*
23	W	w	Consonant			*as in what*
24	X	x	Consonant			*Glottal sound t*
25	Y	y	Consonant			*As in year*
26	Z	z	Consonant			*as in zero*
27	Hudhaa (')		Consonant			**Apostrophe**
28	CH	ch	Double Consonant			*as in chase*

29	DH	dh	Double Consonant			*glottal sound d*
30	NY	ny	Double Consonant			*Like in co**gn**ac*
31	PH	ph	Double Consonant			*Pope*
32	SH	sh	Double Consonant			*As in shall*
33	TS	ts	Double Consonant			*Like in*
34	ZY	zy	Double Consonant			

3.2.2 Afaan Oromo Language Morphology

Morphology is the branch of linguistics that studies patterns of word formation within and across languages, and attempts to formulate rules that model the knowledge of the speakers of those languages [5]. The rules understood by the speaker reflect specific patterns (or regularities) in the way words are formed from smaller units and how those smaller units interact in speech. Word is the most basic unit of linguistic structure. As stated in [5] Like other Ethiopian languages, Afaan Oromo has complex and rich morphology. Therefore, the grammatical information of the language is described in relation to its morphology. This makes it very hard to create grammar checker, spelling checker and any other NLP task like MT.

The Afaan Oromo morphemes discussed here is based on the analysis and discussion from [7], [25], [76], [78] works. Afaan Oromo morphemes are divided into two: ***Dhamjecha walabaa*** 'free morphemes' and ***Dhamjecha hirkataa*** 'bound morphemes'. Free morphemes are those morphemes that can stand alone, that is, not attached to some other morpheme to give a meaning. By contrast, bound morphemes have to be attached to some other morpheme to give a meaning. For instance, in ***bishaaniin*** 'by water', /-**iin/** is a bound morpheme and ***bishaan*** 'water' is a free morpheme.

3.2.2.1 Free morphemes

Afaan Oromo free morphemes are classified into two: ***Dhamjecha hiikaa*** 'lexical morpheme' and ***Dhamjecha tajaajilaa*** 'functional morpheme'. Lexical morphemes are free morphemes that have their own content and stand for one meaning. For example, if we use the word ***Bishaan*** **'**water' as a subject or as the object of a sentence its meaning never changes. In contrast, functional morphemes generally perform some kind of grammatical role, carrying little meaning of their own. Functional morphemes specify a relationship between other morphemes. For instance, words like ***Kun***, ***Sun***, ***and Kana*** are all functional morphemes.

3.2.2.2 *Bound morphemes*

Based on their content, bound morphemes are classified into two: ***hundee hirkataa*** 'bound root' and ***fufii*** 'affix'.

1. **Bound root**

Bound roots are morphemes that make the most precise and concrete contribution to the words meaning. For instance, words like ***beeka***, ***beeki***, ***beekte***, ***beeku***, ***beekti***, ***beekumsa***, *and* ***beekan*** have a bound root ***beek-***. Most of the root words in Afaan Oromo are bound root.

2. **Affix**

Affix is a bound morpheme that attaches to bases. The classification of Afaan Oromo affix can be done based on the position of the affix and in terms of the shift of word class. Based on their location, Afaan Oromo affixes are classified into four: ***fufii duree*** 'prefix', ***fufii giddee*** 'infix', ***fufii naannee*** 'Circumfix' and ***fufii duubee*** 'Suffix'. Table 3:2 summarizes these affixes.

Table 3:2. Affix classification based on their position

Affix	**Definition**
Prefixes	Bound morphemes which occur only before other morphemes. Examples: **Hin- (<u>Hin</u>**deemne**)**➞he/you can't go **Ni-(<u>Ni</u>**deemna**)** ➞we can go
Suffixes	Bound morphemes which occur following other morphemes. Examples : **-e** (ijaar**<u>e</u>**) ➞he built **-ame (**qal**<u>ame</u>)** ➞slughtered
Infixes	Bound morphemes which are inserted in to other morphemes. Examples: **-a (**ag**<u>a</u>**rre**)** ➞we saw it.
circumfixes	Bound morphemes that are attached to a root or stem morpheme both at the beginning and end. Example: **dhe- (<u>dhedhe</u>**eroo) ➞ long **di-** (in **<u>di</u>**d**<u>di</u>**imo) ➞ red

3.2.3 Word Formation Process in Afaan Oromo

A word, in Afaan Oromo "jecha", is a part of language containing one or more sounds that can stand independently and make sense [76]. It can be defined as a sequence of characters delimited by spaces, punctuation marks, etc, in case of a written text. Every language around the world has its own word classes or parts of speech. Most linguists of Afaan Oromo classify the word classes into Noun *(Maqaa),* Affix ***(Durduubee),*** Verb (***Gochibsa***), Adjectives (***Maqibsa***), Conjunction ***(Qabsiistuu)*** and Pronoun ***(Bamaqaa)*** [76]. The words grouped into these word classes are formed through either *derivation* or *inflection.*

3.2.3.1 Derivation

Derivation is a process of word formation in which one or more affixes is attached to a root word to produce a new word known as derived word. Usually, derivation will change the part of speech of the root word to which a suffix is added. Afaan Oromo is derivationally rich language. The word formation discussed here is based on the analysis and discussion from [5], [7], [76] research works.

A. Noun Derivation

Nouns are words that are used to name or identify any of categories of things, people, places or ideas or a particular of one of these entities. In Afaan Oromo, there is a large stock of nominal derived from adjectival, verbal and nominal bases. Suffixes involved in the derivation of nouns in Afaan Oromo are classified into different groups based on the type of word class they change into nouns.

Group 1 */-eenya/ and /-ina/*

These suffixes are used to derive nouns from adjectives as the following set of examples illustrate.

Adjective	**Gloss**	**Affix**	**Derived noun**	**Gloss**
Adii	White	-eenya	Addeenya	Whiteness
Dhiyoo	Near	-eenya	Dhiyeenya	Closeness
Bareedaa	Beautiful	-ina	Bareedina	Beauty

Group 2 */-a/, /-aa/, /-ee/, /-ii/, /-sa/, /-sa/, /-aatii/, /-cha, /-choo/, /-maata/, /-nsa/, /-noo/*

These suffixes are used to derive nouns from verb. The following examples indicate the derivation of such nouns.

Verb	**Gloss**	**Affix**	**Derived noun**	**Gloss**
Ibse	Make it clear	-aa	Ibsaa	Light
Lole	He fought	-a	Lola	War
Dhuge	He drank	-aatii	Dhugatii	Drink

Group 3 */-ummaa/ and /-ooma/*

These suffixes are used to derive nouns from other nouns. It changes the concrete noun (maqaa waan qabatamaa) to abstract noun (maqaa waan yaadaan jiruu). The following example shows the derivation of nouns from other nouns

Noun	Gloss	Affix	Derived noun	Gloss
Bilisa	Free	-ummaa	Bilisummaa	Freedom
Garba	Slave	-ummaa	Garbumma	Slavery
Fira	Relative	-ummaa/ -ooma	Firooma	Relationship

B. Verb Derivation

The verbs are words or compound of words that express action, a state of being and/or relationship between two things. Verbs are doing words or action words (for instance, ***figi*** 'run', ***rukuti*** 'hit', ***konkolachisi*** 'drive', and ***nyaadhu*** 'eat'), but some verbs show a 'state of being' (for instance, ***mullate*** 'appear' and ***fedhe*** 'like'). Like that of nouns, Afaan Oromo verbs are derivational. Similarly suffixes involved in the derivation of verbs (verbalizer) in Afaan Oromo are classified into different groups based on the type of word class they change into verbs.

Group 1 */-oom-/, /-aa'-/, /-a'-/*

These verbalizer are used to derive verbs from nouns and adjectives. The following examples indicate the derivation of such verbs.

Word	**Gloss**	**Verbalizer (Affix)**	**Derived verb**	**Gloss**
Arjaa	Donator	-oom-	Arjoome	Donated
Gurracha	Black	-a'-	Gurracha'e	Blacked
Qulqulluu	Blessing	-aa'-	Qulqullaa'e	Blessed

Group 2 */-at-/, /-am-/, /-sis-/, /-siis/, /-s-/*

These verbalizer are used to derive verbs from adjectives and another verb. Examples include:

Word	**Gloss**	**Verbalizer (Affix)**	**Derived verb**	**Gloss**
Mare	Rolled	-at-	Marate	Have rolled
Diima	Red	-at-	diimate	Become red
Dhuge	He drunk	-siis-	Dhugsiise	Cause to drink

C. Adjectives Derivation

Adjectives are words that describe or modify nouns (and pronouns). Forming adjectives from other lexical categories is termed as adjectival. From stative verb like **/diim-at**/ 'become red' one can derive the adjective **/diim-at-aa (-tuu)/** 'reddened'. In Afaan Oromo adjectives can be formed from verbs by taking adjectival like **/-aa/, /-tuu/, /-eessa/, and /-eettii/.** Afaan Oromo adjectives are not derivatives as nouns and verbs, this is because there are a few numbers of adjectivizers in the language [88]. The following examples show the derivation of such adjectives.

Word	**Gloss**	**Affix**	**Derived Adjectives**	**Gloss**
Sodaate	Feared	-aa/-tuu	Soodaataa/sodatuu	Fearful
Iyee	Shouted	-eesa/ -etti	Iyyeessa/iyyeetti	Poor

3.2.3.2 Inflection

The words in a given class vary their forms for grammatical reasons. Inflection does not form a new word. The word formation discussed here is based on the analysis and discussion from [7], [76], [78] this research works.

A. Noun Inflection (Maqaa)

Almost all Afaan Oromo nouns in a given text have person, number, gender and possession markers which are concatenated and affixed to a stem or singular noun form. Afaan Oromo nouns are inflected to indicate different grammatical functions like gender, number and definiteness [76]. Inflectional suffixes are combined with stem usually resulting in a word of the same class as the original stem. Both Afaan Oromo nouns and adjectives are highly inflected for number and gender. The principles of noun inflection here apply to verbs and adjectives. Number and gender markers are considered as inherent categories.

Number (Lakkofsa)

Afaan Oromoo distinguishes between singular and plural nouns as for other languages. In contrast to the English plural noun markers /-(e) s/, there are different types of suffixes that can be attached to nouns to indicate plurality. Plural nouns are formed through the addition of suffixes.

The most common plural maker suffixes in Afaan Oromo are: **/*-oota*/**, **/*-oolii*/**, /*-een*/, **/*-lee*/**, **/-*wwan*/**, **/*-yyii*/**, **/*-eetii*/**, **/*-ii*/**, **/*-oo*/**, **/*-n*/**. The following examples indicate the inflection of nouns for number.

Non	**Gloss**	**Plural**	**Gloss**
Saawwa	Cow	Saawwan	Cow’s
Nama	Man	Nmoota	Men
Meesha	Instrument	Meeshota	Instruments
Waraabessa	Hyena	Waraabeyyii	Hyenas

Gender (Koorniyaa)

Oromo has two grammatical genders, masculine and feminine in similar way to other Afro Asiatic language. There is no neutral gender as in other language like English. Small number of nouns end with ***/-eessa/*** and ***/-eettii/*** to indicate a gender as in ***obboleessa*** ‘brother’ and ***obboleettii*** ‘sister’. Grammatical gender normally agrees with biological gender for people and animals; thus nouns such as ***abbaa*** ‘father’, ***ilma*** ‘son’, and ***sangaa*** ‘ox’ are masculine, while nouns such as ***haadha*** ‘mother’ and ***intala*** **‘**girl’ are feminine. However, most names for animals do not specify biological gender. Names of astronomical bodies are feminine as in ***aduu*** ‘sun’and ***urjii*** ‘star’. The gender of other inanimate nouns varies somewhat among dialects. Suffixes like ***/-ch/*** and **/-tti/** are used to show gender in some cases. The following examples indicate the inflection of such nouns for gender.

Noun	**Affix**	**Masculine**	**Feminine**
Garba	-ch/ -ttii	Garbicha	Garbittii
Saree	-ch/-ttii	Saricha	sarittii

Words like ***korma/ korbeessa, dhaltuu/ goromtii*** can also be used with noun to indicate gender. For instance, ***leenca kormaa*** means lion and ***leenca dhaltuu*** means lioness.

Definiteness (Beekamtummaa)

There is no indefinite article (such as *a, an, some)* in Afaan Oromo like in English. The definiteness article ‘*the’* in English is /***-icha***/ for masculine nouns and /***-ittii***/ for feminine nouns in Afaan Oromo. Vowel endings of nouns are dropped before appending these suffixes. For instance, ***laga*** ‘river’ becomes ***lagicha*** ‘the river’, ***haroo*** ‘lake’ becomes ***harittii*** ‘the lake’. In Afaan Oromo definite suffixes appear to be used less often than in English, and they seem not to co-occur with the plural suffixes. Afaan Oromo verbs are also highly inflected for gender, person, number and tenses. Examples:

- ✓ Inni kaleessa dhuf**e**. → (He came yesterday)
- ✓ Isheen kaleessa dhuf**te**. → (She came yesterday)

- ✓ Isheen dhufaa jir**ti**. → (She is coming)

In the above four sentences the gender and tense of the sentences are described through suffix which is attached to the verb stems /***dhuf-***/ and /***jir-***/.

B. **Adjective Inflection**

In Afaan Oromo, there are uninflected adjectives that are morphologically simple and do not take inflectional affixes for any of the grammatical categories as there are inflected adjectives. The following examples show uninflected adjectives:

Uninflected word	**Glossary**
Haaraa	New
Doofaa	Fool

Like nouns, the inflectional categories of adjectives are inflected for different grammatical categories such as number, gender, definiteness and case. But there is a different way of affixation that is used for inflectional purpose of adjectives only. For instance, unlike nouns, adjectives are inflected by reduplication of stem to mark the plurality. The */-(o)ota/* suffixes are also used to show plurality. Adjectives are inflected for gender class. The marker */-aa/* is used to show masculine, while the markers */-tuu/* and */-oo/* show feminine gender. Both /*-tuu/ and /-oo/* morphemes are suffixed to most adjectives that end with long /*-aa/* by removing them. The case and definiteness grammatical categories of adjectives are identical with noun. The following Example show inflected word (Adjective)

inflected word	**Glossary**
Dursaan ***gabaabaa*** ***(***masculine***)*** dha.	Dursa is short
Hawwiin ***furdoo*** ***(***feminine***)*** dha	Hawi is fat.

C. **Verb Inflection**

Afaan Oromo verb inflection occurs for inherent, the basic members of a word class triggering inflection on that word class such as aspect, mood, and voice and agreement properties, inflection of a word class for properties out of its members such as person, number and gender [76]. There are some studies that include tense to inflectional categories of a verb independently but from the three major tenses present, past and future, Afaan Oromo mainly identifies between past and non-past in its morphology because the morphological markers do not differentiate each tense categories. Because of this, Afaan Oromo conflates tense with aspectual categories.

D. **Pronoun**

In Afaan Oromo, like in other languages, a pronoun is a word that is used instead of a noun or noun phrase. They are characterized based on number and gender. The Pronoun in Afaan Oromo can be independent or hidden with the verb based on their existence in a sentence. Independent pronouns are pronouns exist in a sentence as a separate word in the sentence. In the following example, "*Inni*" is an independent personal pronoun.

Example: Yohaannis yeroo dhufe homaa hin nyaatu homaas hin dhugu ture; Isaanis ***Inni*** dhukuba qaba jedhu.

However, hidden pronouns are pronouns attached to the word in the sentence. In the following example, the word "*deemte*" indicates the pronoun "*Ishee*" hidden in it.

Example: Toltuun kaleessa magaala Finfinnee ***deemte*** ture.

Personal Pronouns

Personal pronouns are one type of pronouns that we may use to express particular person. In Afan Oromo language, personal pronouns are formal with everyone except close friends and other relative persons [80]. Concerning the form and usage of a personal pronoun, in Afan Oromo as a subject, it comes at the beginning of the sentence and as an object, it comes before the verb, unlike English, it comes after the verb.

English object	**Afaan Oromo object**	**English subject**	**Afaan Oromo subject**
Me	Ana	I	Ani
Us	Nu / Nuu	We	Nutii / Nuyi
You	Si	You	Ati
You	Isin	You	Isin
Him	Isa	He	Inni
Her	Ishee / Isii	She	Isheen / Isiin
Them	Isaani	They	Isaan

There is no special word for '**it**' in Afan Oromo, instead '*inni*' /him/ used formally or informally. In addition, gender markers (verb) for other things that are not human being.

Example: That Book is New. It was bought last week

Kitaabni sun haaraadha. Torban darbe bitame.

The verb '*bitame*' shows that the subject is the third person '*he'*.

Demonstrative Pronouns

A demonstrative pronoun is a pronoun that is used to point something specific with in a sentence. They can be singular or plural and points out items in time or space. Some these pronouns are: *kun(i)/this, tun(i)/this, sun/that ,kana/this, tana/this, sana/that, kunniin/these,kanneen/those.*

Possessive pronouns

Possessive pronouns are pronouns that designate possession. These pronouns include *koo, kiyya, isaa, ishee, keenya, isaani, keessan.*

Reflexive pronouns

A reflexive pronoun indicates the person who realize the verb action is same with receipt of the action. These includes *of/self, uf/self-p*ronouns.

Example: Isheen **of** laalti [she looks at **herself**]

Interrogative pronoun

Interrogative pronoun is used to make asking question. Some of these pronouns include *eenyu/who, kan eenyuu/whose, maal(i)/what, maaliif/why, eessa/where, meeqa/how much, akkam/how, yoom/when.*

E. **Conjunction**

A conjunction is a word that is used to connect words, phrases, clauses and sentences or coordinate two or more words, phrases, clauses and sentences. The following table are a list of Afaan oromo conjunction and their English equivalent.

Conjunction in Afaan Oromo	**English equivalent**
Fi	And
Yookaan / Yookiin	Or
Waan ta'eef	So, therefore
Yoo	Unless
Kan	For
Kanaafi	As a result, consequently
Haa ta'u malee	But
Siachi	Yet

3.2.3.3 Compounding

Compounding is the process of putting words together to build a new one that does not denote two things, but one and that is pronounced as one unit. The difficulty with compounds is to work out which words are more heavily pronounced in their first and which ones in their second part. Another problem, also for native speakers, may be to detect which compounds are written how, because some compounds are hyphenated, others are written separately and some are written as one word. This section describes Afaan Oromo lexical compounds and the characteristics which distinguish them from others

1. **Compound Nouns**

In Afaan Oromo, compound nouns are productively formed by a combination of different POS.

Noun Compounds: Two nouns can combine to form various kinds of compound nouns. For example, the noun **/abbaa/** 'father' or **/haadha/** 'mother' can occur as a first member as given in the following examples to give a compound noun:

Noun	Gloss	Noun	Gloss	Compound noun	Gloss
Abbaa	Father	Buddeena	Food	Abbaa Buddeenaa	Step father
Haadha	Mother	Mana	House	Haadha Manaa	Wife

A noun referring to a container and another one referring to a thing contained in it combine to form a locative compound noun as given in the following examples:

Noun	Gloss	Noun	Gloss	Compound noun	Gloss
Okkotee	Jar	Bishaanii	Water	Okkote-bishanii	Water jar
Saanduqa	Box	Wayyaa	Cloth	Saanduqa-wayyaa	Box of clothes

Instrumental compounds are formed by combining two nouns of which the first member is instrumental for the realization of the thing designated by the second member.

Noun	Gloss	Noun	Gloss	Compound Noun	Gloss
Dhagaa	Stone	Daakuu	Flour	Dhagaa-daakuu	Millstone

Names of certain parts of the body may also be combined with other nouns to form compounds designating names of diseases.

Noun	Gloss	Noun	Gloss	Compound Noun	Gloss
Mataa	Head	Cabsaa	Breaker	Mata-cabsaa	Headache
Dugda	Back	Kuta	Cutting	Dugda-kutaa	Backache / pain

Other compounds may be formed by combining nouns referring to locations where activities take place. Such compounds can be treated as locatives. They refer to locations. Examples include:

Noun	**Gloss**	**Noun**	**Gloss**	**Compound Noun**	**Gloss**
Mana	House	Barumsaa	Learning	Mana barumsaa	School
Bakkee	Field	waraanaa	Fighting	Bakkee waraanaa	Battlefield
Mana	House	Sagadaa	Praying	Mana sagadaa	Church / mosque

Compound nouns can also be formed by combining noun and adjectives. The process is not productive. Examples include:

Noun	**Gloss**	**Adjective**	**Gloss**	**Compound Noun**	**Gloss**
Sanbata	Sunday	Guddaa	Big	Sanbata Guddaa	Sunday
Muka	Tree	Gurraacha	Black	Mukaa-gurraacha	Black tree

Compound nouns can also be formed by combining verbs with noun. Examples include:

Verb	**Gloss**	**Noun**	**Gloss**	**Compound Noun**	**Gloss**
Qotee	Plough	Bulaa	Live	Qotee-bulaa	Farmer

2. **Compound Adjective**

Compound adjectives are not as productive as compound noun. But we have instances of them followed by a combination of different categories. Compound adjectives can also be formed by combining noun and noun. It can be formed by combining two nominal of which the first is **abbaa** 'father' or **haadha** 'mother' and the second a nominal designating some attributes. The following examples can be given.

Nominal	**Nominal**	**Compound Adjective**	**Gloss**
Abbaa/Haadha	Boyichaa	Abbaa/Haadha boyichaa	Crying
Abbaa/Haadha	Dubbii	Abbaa/Haadha dubbii	Talkative
Abbaa/Haadha	Hirribaa	Abbaa/Haadha hirribaa	Sleepy

The other one is Noun-Adjective compounding. In this type of compounding the second member is a modifier of the first. Examples include:

Word	**Gloss**	**Word**	**Gloss**	**Adjective**	**Gloss**
Bifa	Color	Badii	Useless	Bifa-badii	Ugly
Garaa	Stomach	Qulqulluu	Clean	Garaa qulqulluu	Clean hearted
Ija	Eye	Jabeessa	Strong	Ija-jabeessa	Shameless

A combination of two adjectives can form a compound adjective in which the second qualifies the first in terms of degree. Examples include:

Adjective	**Gloss**	**Noun**	**Gloss**	**Compound Adjective**	**Gloss**
Adii	White	Qulqulluu	Clean	Adii qulqulluu	Pretty-white
Aja’aa	Stinky	Tortoraa	Rotten	Aja’aa tortoraa	Rotten-stinky

1.1 Afaan Oromo and English Sentence Structure

The Sentence is a group of words come together to express completes thoughts. Afaan Oromo and English have differences in their syntactic structure [25], [76], [78]. In Afaan Oromoo follows Subject-Object-Verb (*Matima-Aantima-Gochima*) word order pattern, where the subject comes first, then the object and the verb next to the object. For example, if we take Afaan Oromo sentence “Tolaan midhaan nyaate”, “Tolaan” is the subject, “midhaan” is the object and “nyaate” is the verb of the sentence. In case of English, the sentence structure is subject-verb-object. For example, let us see the translation of the above Afaan Oromo sentence is translated into English it will be “Tolaan ate food” where “Tolaan” is the subject, “ate” is the verb and “food” is the object. There is also a difference in the formation of adjectives in Afaan Oromo and English. In Afaan Oromo adjectives follow a noun or pronoun; their normal position is close to the noun they modify while in English adjectives usually precede the noun, i.e. Afaan Oromo adjective agree with its head noun, this is not the case in English. For instance, miicaayyoo bareeduu (beautiful girl), bareeduu (adj.) follows miicaayyoo (noun).

1.2 Afaan Oromo and English Language Articles

In Afaan Oromo language there is no special word class of articles that appeared before a noun, like they exist in English. In English there are three main semantic choices for article insertion: *definite article* (the), *indefinite article* (a, an, some, any). In case of Afaan Oromo, there are no articles that are inserted before nouns unlike that of English rather the last vowel of the noun is dropped and suffixes (-icha, -ittii, -attii, -utti) are added to show definiteness instead of using definite articles. In some Afaan Oromo dialects the suffix **-icha** for male and –**ittii (n)** for female which usually has a singularize function is used where other languages would use a definite article as shown in the example below:

- ✓ Jaarsa → 'old man' →Jaarsichi →'the old man'
- ✓ Jaarti → 'old women' →Jaartittin →'the old women'

- ✓ karaa → 'road' → karicha →'the road'
- ✓ Nama →'man' → namicha/namticha → 'the man'
- ✓ Haroo → 'lake' → harittii →'the lake'

1.3 Punctuation Marks

Punctuation marks are placed in text to make meaning clear and reading easier. It is used in both Afaan Oromo and English languages are the same and used for the same purpose with the exception of apostrophe mark. Apostrophe mark (') in English shows possession, but in Afaan Oromo it is used in writing to represent a glitch sound known as '*hudhaa*'. It plays an important role in Afaan Oromo reading and writing system.

For example, it is used to write the word in which most of the time two vowels are appeared together like "**ba'e**" to mean "get out", **"du'a"** to mean "death" and **"boba'aa"** to mean "fuel" with the exception of some words like **"bal'ina"** to mean "wide" which is identified from the sound created.

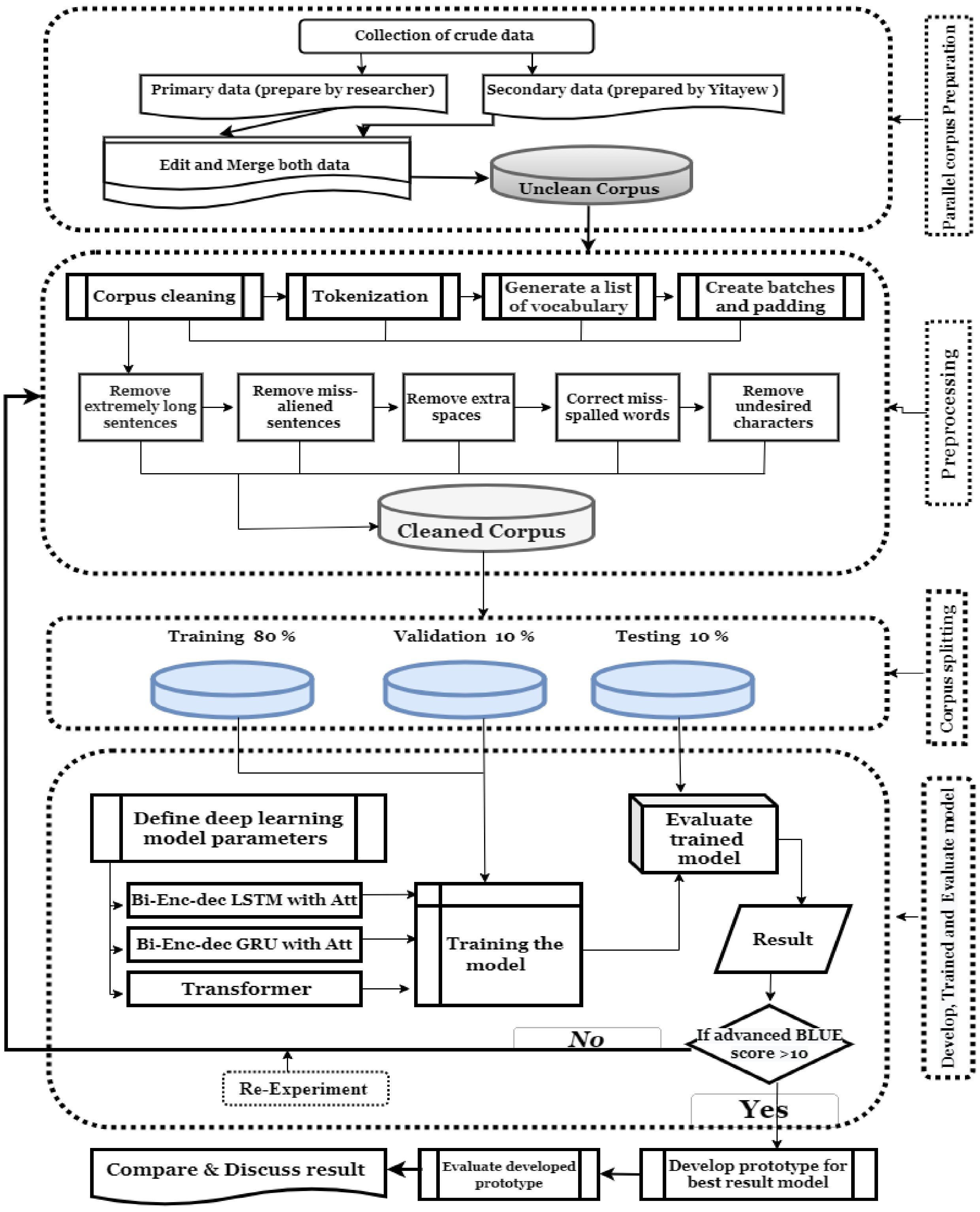
CHAPTER FOUR

PROPOSED SYSTEM

Under this chapter discusses the overall frame work and designing of bidirectional English to Afaan Oromo MT system based on bidirectional encoder and decoder language modeling by using LSTM and GRU with attention mechanisms and also transformer based architecture. Then, we describes the components of each proposed system in details. Lastly develop prototype for best result model and manually evaluate the developed model usability and translation quality. In general, this chapter briefly explains how this study was systematically done.

4.1 The Proposed System

DL approach is represented by a spectrum of architecture that can build solutions for a range of problem areas. In this research work, we focused on the architecture of a Bi-directional English-Afaan Oromo MT system by using different DL algorithms like, bidirectional encoder and decoder LSTM and GRU with attention mechanisms and transformer algorithms. Therefore, encoder-decoder based MT became the better choice for the simplicity of the steps of modeling for MT task. In order to improve the readability of this study, a research framework is given in figure 4:1 and require the consideration of the required components for training, validation and testing for the English and Afaan Oromo languages pairs. This framework works through four major different stages, namely the Parallel Corpus preparation, preprocessing corpus, define and designing seq2seq mode, developing and training seq2seq model, evaluation of trained performance and finally discuss the model result. In order to compare the result of all proposed model, we need to use balance hyper-parameters in all proposed solution.



Collection of crude data
Primary data (prepare by researcher)
Secondary data (prepared by Yitayew)
Edit and Merge both data
Unclean Corpus
Parallel corpus Preparation
Corpus cleaning
Tokenization
Generate a list of vocabulary
Create batches and padding
Remove extremely long sentences
Remove miss-aliened sentences
Remove extra spaces
Correct miss-spalled words
Remove undesired characters
Cleaned Corpus
Preprocessing
Training 80 %
Validation 10 %
Testing 10 %
Corpus splitting
Define deep learning model parameters
Bi-Enc-dec LSTM with Att
Bi-Enc-dec GRU with Att
Transformer
Training the model
Evaluate trained model
Result
If advanced BLUE score >10
No
Yes
Re-Experiment
Develop, Trained and Evaluate model
Compare & Discuss result
Evaluate developed prototype
Develop prototype for best result model

Figure 4:1 Overall framework of the proposed system.

** Bi-Enc-dec LSTM with Att: Bidirectional encoder and decoder long short-term memory with attention, Bi-Enc-dec GRU with Att: Bidirectional encoder and decoder Gated recurrent unit with attention*

Each stages have sub phase. A brief explanation of each phase, shown in Figure 4:1 is discussed as follows:

4.1.1 Parallel Corpora preparation for the language pairs

We preparer a parallel corpus that contains a collection of original texts in language English and their translations into a set of languages Afaan Oromo. In this phase we discuss step by step the tasks we have accomplished to collect our bilingual parallel corpora.

A. Collection of Crude Data

The amount of parallel corpus and its quality plays significant role in quality of translation output [31], [40]. But For low resource languages like Afaan Oromo, it is extremely hard to find sufficient parallel corpus for training, validation and testing of translation engine. So, the option we have is to create this corpus by researcher or used from previous work as secondary source. To do that we should first identify domains with abundant amount of information in a text format. As there are only few materials available in these language pair. After identifying the domains we collects the raw data from source in a text format. To perform this research experiments, the datasets were collects from Holy Bible and we use the previous collected parallel corpus, prepared by Yitayew Solomon [5]. This also use for extend more coverage of the domain. The reason why we select more corpus from Holy Bible is because of there is no well translated and availability of large amount of textual data. We uses the new international version Bible of all other versions because both the Afaan Oromo and English versions are translated from the same Dead Sea scrolls. This makes it more accurately parallel than other versions of the Bible translation.

In other words, whenever the data is gathered from different sources it was collected in raw format which is not feasible for the analysis. So, before cleaning raw data we need to collect crude data for alignment in both language pairs. The data is sentence-aligned, that means that source and target file contain one sentence per line which correspond to each other. For this research, we used two tools to collect the parallel corpus. As the main tools *Notepad++* and *Microsoft excel* are used for alignment purpose of crude data. After collected crude data, the English and the Afaan Oromo sentences have been saved in different text files separately in my local computer, named *eng.txt* and *orm.txt.*

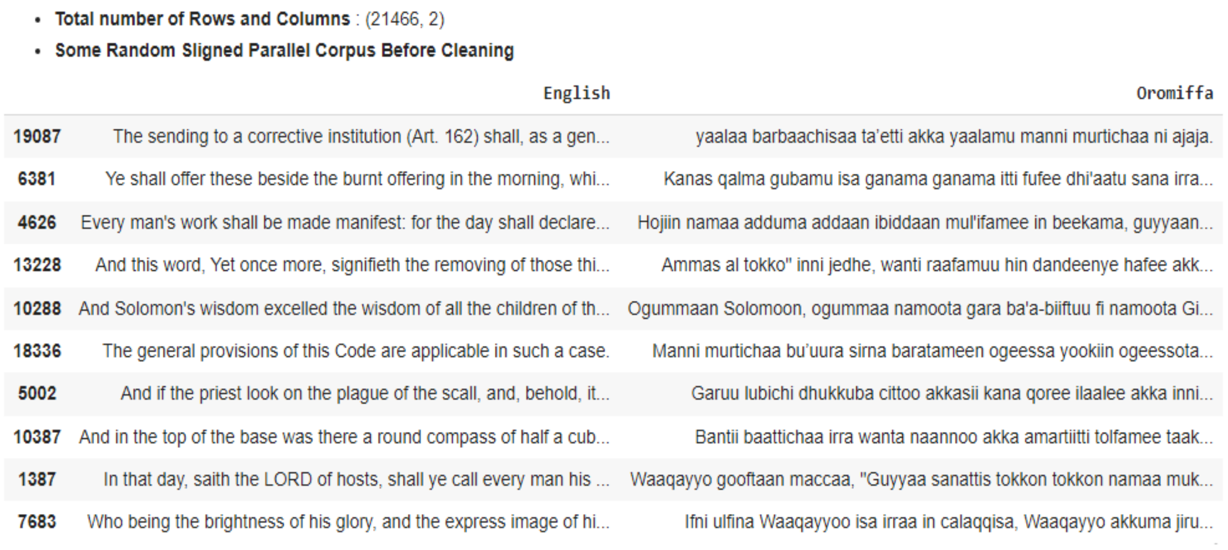
B. Edit and merge corpus

After prepared and collected crude parallel corpus from primary (prepared by researcher) and secondary source (Yitayew Solomon) we merge both crude parallel corpus manually. What would make things easier is being able to merge a number of text files into one single file. It makes to check the contents of two files. When to merge both corpus we try to drop the redundant sentence and checking the grammar and spelling of the aligned sentence by using *Notepad++* and *Microsoft excel* in order to open both file in two columns. We try to drop mistranslated and redundant sentence. Both software making your own text file is simple and requires no coding or programming knowledge. Table 4:1 Show detailed in information about parallel corpus source and size of merged parallel corpus.

Table 4:1 Detailed information about the parallel corpus

Prepared by	**Corpus Source**	**Corpus size**	**Total**
Researcher	Holly Bible and religious document	17105	21461
Yitayew Solomon [5]	Collected from Ethiopian criminal code, Ethiopian constitution and Oromia Regional State Duties and Responsibilities	4361	

After merged both file, the file contains Afaan Oromo and English sentences have been saved in different text files separately, named *oromiffa.txt* and *english.txt*. Finally, we load both file in *Dataset* folder of my Google Drive storage. Then via to Google Colab notebook editor, we access my text file from Google Drive for further preprocessing those parallel corpus. One of these unclean parallel corpus types is shown in figure 4:2



- **Total number of Rows and Columns** : (21466, 2)
- **Some Random Sligned Parallel Corpus Before Cleaning**

	English	Oromiffa
19087	The sending to a corrective institution (Art. 162) shall, as a gen...	yaalaa barbaachisaa ta'etti akka yaalamu manni murtichaa ni ajaja.
6381	Ye shall offer these beside the burnt offering in the morning, whi...	Kanas qalma gubamu isa ganama ganama itti fufee dhi'aatu sana irra...
4626	Every man's work shall be made manifest: for the day shall declare...	Hojiin namaa adduma addaan ibiddaan mul'ifamee in beekama, guyyaan...
13228	And this word, Yet once more, signifieth the removing of those thi...	Ammas al tokko" inni jedhe, wanti raafamuu hin dandeenye hafee akk...
10288	And Solomon's wisdom excelled the wisdom of all the children of th...	Ogummaan Solomoon, ogummaa namoota gara ba'a-biiftuu fi namoota Gi...
18336	The general provisions of this Code are applicable in such a case.	Manni murtichaa bu'uura sirna baratameen ogeessa yookiin ogeessota...
5002	And if the priest look on the plague of the scall, and, behold, it...	Garuu lubichi dhukkuba cittoo akkasii kana qoree ilaalee akka inni...
10387	And in the top of the base was there a round compass of half a cub...	Bantii baattichaa irra wanta naannoo akka amartiitti tolfamee taak...
1387	In that day, saith the LORD of hosts, shall ye call every man his ...	Waaqayyo gooftaan maccaa, "Guyyaa sanattis tokkon tokkon namaa muk...
7683	Who being the brightness of his glory, and the express image of hi...	Ifni ulfina Waaqayyoo isa irraa in calaqqisa, Waaqayyo akkuma jiru...

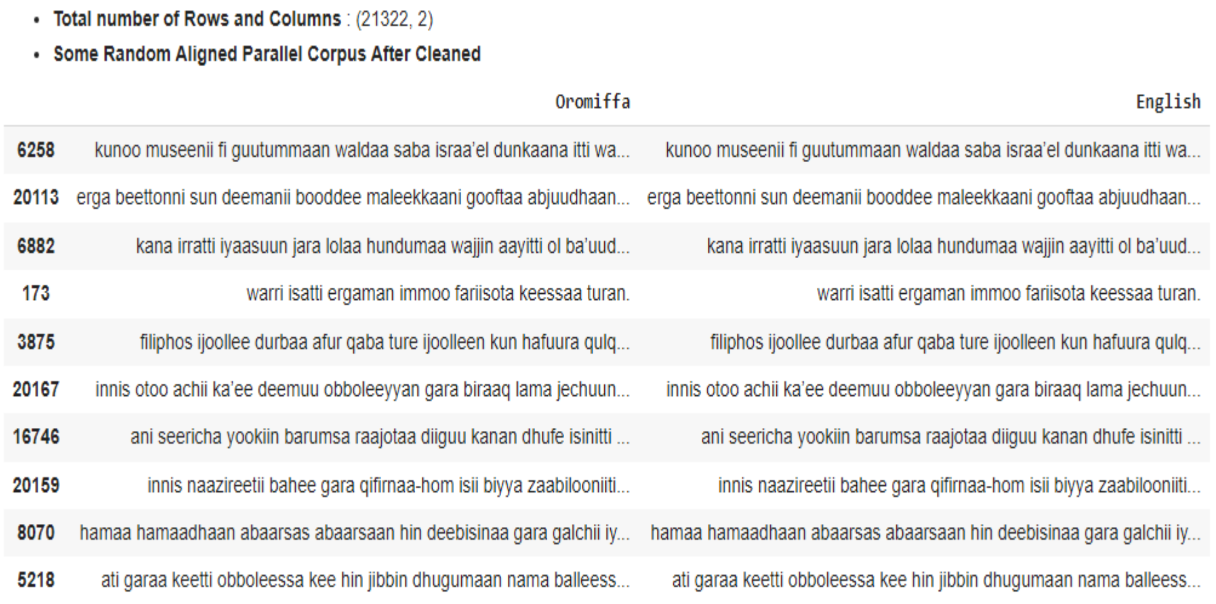
Figure 4:2 Random Sample parallel corpus before cleaning

4.1.2 Preprocessing Parallel Corpus

Preprocessing parallel corpus is a method to clean the text dataset and make it ready to feed corpus to the model. So Pre-processing is important to remove undesired characters and words from parallel corpus. Parallel corpus contains noise in various forms like punctuation, text in a different case as we seen under figure 4:2. We see some of the pre-process it can do for clean those corpus.

A. Clean Parallel corpus

To perform the experiments, we tried to collect manually about 17105 parallel sentences and additionally the previous corpus, with 4479 parallel sentences, was prepared by Yitayew Solomon [5] collected from different source. Before using python code we try to clean on *notepad*$^{++}$ in order to avoided miss-aliened sentences, remove mistranslated sentences and miss-spalled words as well as we remove duplicate sentences from the file. Specially the quality of the previous collected corpus, that prepared by Yitayew Solomon very low, most of the corpus have miss-spalled and miss-aliened. In order to avoid those error we try to clean on *notepad*$^{++}$. After we cleaned the whole corpus on *notepad*$^{++}$, we use different python codes for parallel corpus cleaning purposes. The task of parallel corpus cleaning includes several activities like, remove the word with digit, remove digit, extra space, unwanted punctuation mark and convert into lower case. Remove digit and the words with digits which are combined *like* ***49(3), dabalaa32, 65/ 1995*** *or* ***galme5ts7, ta\u2019a, u\u2019aa remove*** *etc*, this type of text creates a problem for machines to understand. The other one is remove extra spaces that means more than one space is left between the texts, so we need to control this problem. Also we removing unwanted punctuation marks do not have major relevance for translation except Question Marks (?), Apostrophe mark ('), Full stop (.) and Over line (-). Finally convert all sentence into lower case. Then we can see in the following figure 4:3 output after preprocessing are done.



- **Total number of Rows and Columns** : (21322, 2)
- **Some Random Aligned Parallel Corpus After Cleaned**

	Oromiffa	English
6258	kunoo museenii fi guutummaan waldaa saba israa'el dunkaana itti wa...	kunoo museenii fi guutummaan waldaa saba israa'el dunkaana itti wa...
20113	erga beettonni sun deemanii booddee maleekkaani gooftaa abjuudhaan...	erga beettonni sun deemanii booddee maleekkaani gooftaa abjuudhaan...
6882	kana irratti iyaasuun jara lolaa hundumaa wajjin aayitti ol ba'uud...	kana irratti iyaasuun jara lolaa hundumaa wajjin aayitti ol ba'uud...
173	warri isatti ergaman immoo fariisota keessaa turan.	warri isatti ergaman immoo fariisota keessaa turan.
3875	filiphos ijoollee durbaa afur qaba ture ijoolleen kun hafuura qulq...	filiphos ijoollee durbaa afur qaba ture ijoolleen kun hafuura qulq...
20167	innis otoo achii ka'ee deemuu obboleeyyan gara biraaq lama jechuun...	innis otoo achii ka'ee deemuu obboleeyyan gara biraaq lama jechuun...
16746	ani seericha yookiin barumsa raajotaa diiguu kanan dhufe isinitti ...	ani seericha yookiin barumsa raajotaa diiguu kanan dhufe isinitti ...
20159	innis naazireetii bahee gara qifirnaa-hom isii biyya zaabilooniiti...	innis naazireetii bahee gara qifirnaa-hom isii biyya zaabilooniiti...
8070	hamaa hamaadhaan abaarsas abaarsaan hin deebisinaa gara galchii iy...	hamaa hamaadhaan abaarsas abaarsaan hin deebisinaa gara galchii iy...
5218	ati garaa keetti obboleessa kee hin jibbin dhugumaan nama balleess...	ati garaa keetti obboleessa kee hin jibbin dhugumaan nama balleess...

Figure 4:3 Random Sample parallel corpus after preprocessing are done

Finally, now that the dataset has been cleaned, we can save the list of sentences pairs to a file ready for use. The cleaned dataset contains a 21323 sentence pairs. After running, the cleaned sentences are saved in *english.txt* and *oromiffa.txt* files respectively. This is a good number of examples when camper with [5], [7], [25] previous work those discussed under related work for Ethiopian language for developing a MT systems.

B. Tokenization

It is a process of breaking the sentence into a list of individual tokens or words. BY creating the tokenizer functions, we set the tokenize argument to the correct tokenization function for each, with English/Afaan Oromo being the SRC (source) field and Afaan Oromo/English being the TRG (target) field, because we conduct the experiment in both direction. The field also add two extra tokens, the "start of sequence" (sos) and "end of sequence" (eos) tokens via the *init_token* and *eos_token* arguments to the sentences for effective model training.

C. Parallel corpus splitting

After overcome massive amounts of corpus preparation problems, how do we know that we got good results? When training and testing our model, we split the available data into three separate and distinct datasets [83] training dataset, validation dataset, and testing dataset. The model learns by extracting patterns from the training dataset and generalizing the training examples. The model sees and learns from training dataset. Validation dataset is to check the performance of the model during training**.** Hence the model occasionally sees Validation dataset, but never does it learn from this dataset. At the end, when the training is finished, we use the test dataset to check the final performance of the model. Test dataset provides a final, real-world check of an unseen dataset to confirm that the model was trained effectively.

After saved the cleaned pairs of sentence, the next step randomly split this corpus into three categories training dataset, validation dataset and testing dataset, and saving the split portions of dataset to new files namely train_data, valid_data and test_data with percentage splits 80% of the entire corpus for training dataset, 10% of the corpus for validation dataset and 10% of the corpus for testing data with *random-state = 32*. After splitting the corpus in to three districted file we create the *Dataset* folder and saved splinted datase*t* in the *Dataset* folder that we created on google drive. Because we use the same dataset for all proposed model in order compare with each other. The above data splitting methods have been suggested in [86] in the field of Machine Learning (ML). Next, we load the train, validation and test data, by using *TabularDataset* class, we can actually define the dataset of columns stored in JSON format and also map them into integers.

D. **Build the vocabulary**

Build the *vocabulary* for the source and target languages. The vocabulary is used to associate each unique token with an index. The vocabularies of the source and target languages are distinct. By using the minimum frequency (*min_freq*) argument, we only allow tokens that appear at least two times to appear in our vocabulary. Tokens that appear only once are converted into an <unk> (unknown) token. It is important that our vocabulary should only be built from the training set and not the validation or test dataset. This prevents data leakage into our model. Data leakage is when information from outside the training dataset is used to create the model. This additional information can allow the model to learn or know something that it otherwise would not know and in turn invalidate the estimated performance of the mode being constructed [87]. This is done using the build vocab function, which looks through all of the given sentences in the training dataset.

E. Create Batches and Padding

It to create batches of training, testing and validation dataset using iterators. Creating batches is an exhaustive process, luckily we can make use of TorchText's iterator libraries. The original text format must be converted into numericalized form representation. It means converted from a sequence of readable tokens to a sequence of corresponding indexes, using the vocabulary. These can be iterated on to return a batch of data which will have a *src* attribute (the PyTorch tensors containing a batch of numericalized source sentences) and a *trg* attribute (the PyTorch tensors containing a batch of numericalized target sentences). We also need to define a torch device. This is used to tell torchText to put the tensors on the Graphics Processing Unit (GPU) or not. We use the *torch.cuda.is_available()* function, which return true if a GPU is detected on our computer.

When we get a batch of examples using an iterator we need to make sure that all of the source sentences are padded to the same length, the same with the target sentences. The read of sentence starts from start token <*sos*> and continues up to end token <*eos*>. In order to use similar length, one padding is used after end of sentence indicator. So, this representation style uses a fixed length of sentence. Therefore, to set the length of sentences to the fixed length, ***one*** is added to short sentences. We have the maximum sentence length to 66 in English and 55 in Afaan Oromo sentences. So, adding this at the end of each short sentence is known as padding.

We use a BucketIterator instead of the standard Iterator as it creates batches in such a way that it minimizes the amount of padding in both the source and target sentences. BucketIterator provides some additional benefits like sorting the data according to length of the text and group together similar length text in a batch. This helps reduce the amount of padding required. BucketIterator pads the batch according the maximum length sample. We experiment with a batch size of 128. The sentences are tokenized into a list of words and indexed according to the vocabulary. The *"pad"* token gets an index of *1* and *"sos"* token gets index of *2.* Each column corresponds to a sentence indexed into numbers and we have 128 such sentences in a single target batch and the number of rows corresponds to the maximum length of that sentence. Short sentences are padded with 1 to compensate.

4.1.3 Designing Seq2Seq Model

After preprocessing and preparing dataset, next step design the seq2seq model for proposed systems. Therefore, first we design a special kind of RNN based seq2seq model architecture by using bidirectional encoder and decoder LSTM and GRU with attention mechanism algorithms. Lastly we design the most popular transformer based MT architecture is design in order to compare the results of both RNN based and Transformer algorithm to see the improvement result with changing of architecture. Since we need to compare performance in terms of translation quality and time efficiency for low resource language. In the below sections we explain the components of the proposed models and illustrate the architecture of the all models.

4.1.3.1 RNN Seq2Seq Model Architecture

LSTMs and GRUs still prove to be very efficient in MT [61]. Our model follows the common seq2seq learning framework with attention. When to conduct our experimental study we use different RNNs based Encoder-Decoder model with attention mechanism, namely:

1. Bi-LSTM as an encoder and LSTM as a decoder with Attention mechanism.
2. Bi-GRU as an encoder and GRU as a decoder with Attention mechanism.

The architecture of a both RNNs based Afaan Oromo to English and vice versa MT system depicted in figure 4:4. It has three components for each RNN based cell unit: an encoder module, a decoder module, and an attention network. The output of a node goes as an input of another node.

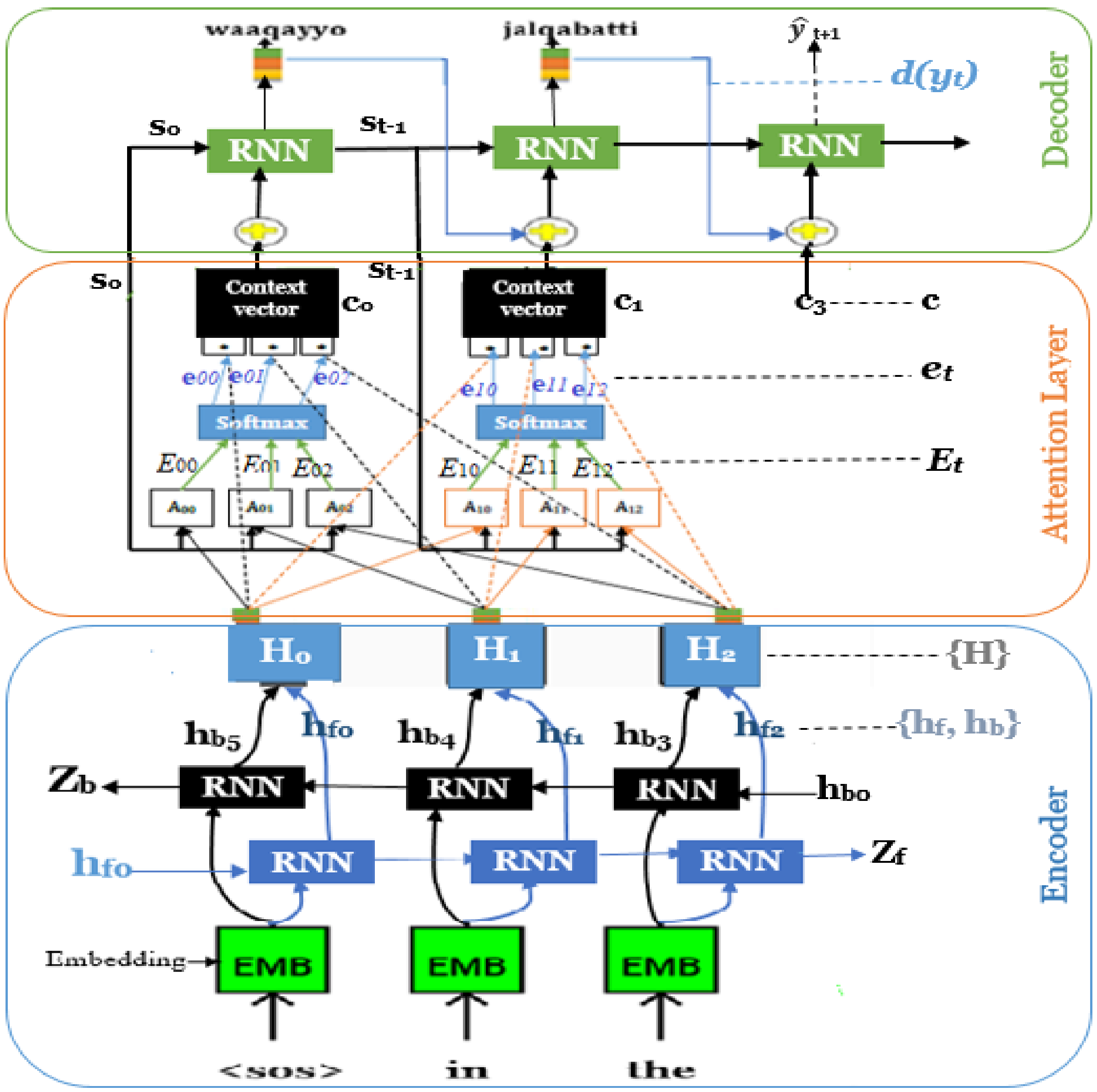


Figure 4:4 Overall architecture of RNNs based for the proposed models

Let's explain the process happening in each components of RNN based proposed system architecture, shown in Figure 4:4 is discussed as follows:

4.1.3.1.1 Word Embedding

The machine cannot directly work on a sequence of sentences. Sentences must be split into a sequence of words, and then these sequences of words must be changed into integer form representation. The first block in the Encoder architecture is the word embedding layer [shown in green block], which converts the input indexed word into a dense vector representation called word embedding (sizes-256). A larger-dimensional embedding can capture more relationships between words but takes more data to learn.

($X_1, X_2, \dots X_n$) → input sequence of symbol representation like sentence *("in the beginning god").* Also, we append the start of sequence "*sos*" and the end of sentence "*eos*" tokens in the starting and in the ending of the input sentence.

($C_1, C_2, \dots\dots\dots C_n$) → sequence of symbol representation (context vector)

($y_1, y_2, \dots y_m$) → output sequence of symbol representation like sentence*("waaqayyoo jalqabatii")*

Encoder: ($X_1, X_2, \dots\dots\dots X_n$) →($C_1, C_2, \dots\dots\dots C_n$)

Decoder: ($y_0, y_1, \dots\dots y_{t-1}; C_1, C_2, \dots\dots\dots C_n$) →($p(y_t| y_0, y_1, \dots\dots y_{t-1}; C_1, C_2, \dots\dots\dots C_n)$)

$X_t \in \{1, 2, \dots, |V|\}$ V →input vocabulary size

$E_t \in R^{|v| \times d_{model}}$ → input embedding matrix

For any time step t, given a minibatch input $X_t \in R^{|v| \times d}$ (|v| (sentence length), d (batch size)).

$e_t = E(x_t) \in R^{d_{model}}$ → *input embedding matrix* *Equation 4:1*

Encoder of the Seq2Seq model takes one input at a time. Our input English word sequence is "*in the beginning god* ". Also, we append the start of sequence *"<sos>"* and the end of sentence *"<eos>"* tokens in the starting and in the ending of the input sentence. Therefore at, At time step-0, the *"<sos>"* token is sent, At time step-1 the token *"in"* is sent, At time step-2 the token *"the"* is sent, At time step-3 the token *"beginning"* is sent, At time step-4 the token *"god"* is sent, At time step-5 the token *"<eos>"* is sent. At each time-step, the input to the encoder RNN is both the embedding, *e*, of the current word, $e(x_t)$, as well as the hidden state from the previous time-step, h_{t-1}, and the encoder RNN outputs a new hidden state h_t. We can think of the hidden state as a vector representation of the sentence so far. The RNN can be represented as a function of both of $e(x_t)$ and h_{t-1}. Each of them follows a simple recursive formula as flows:

$h_t = \text{EncoderRNN}(e(x_t), h_{t-1})$ *Equation 4:2*

We are using the term RNN generally here, it could be recurrent algorithms, such as an LSTM or GRU.

4.1.3.1.2 RNN based Bidirectional Encoder

As we describe, we use bidirectional encoder. The bidirectional encoder consists of two independent encoders, a *forward RNN* going over the embedded sentence from left to right and a *backward RNN* going over the embedded sentence from right to left. The output and final states are concatenate feed to the attention layer. Because we want to process multiple sentences at the same time for speed reasons, it is more efficient on GPU, we need to support mini-batches**.** According to [61], bidirectional RNNs can deliver higher accuracy comparing to single layer of RNNs based. As well as the GRU and LSTM algorithm are motivated by their ability to capture the meaning of a word in its context of the given sentence and able to exploit information both from the past and the future.

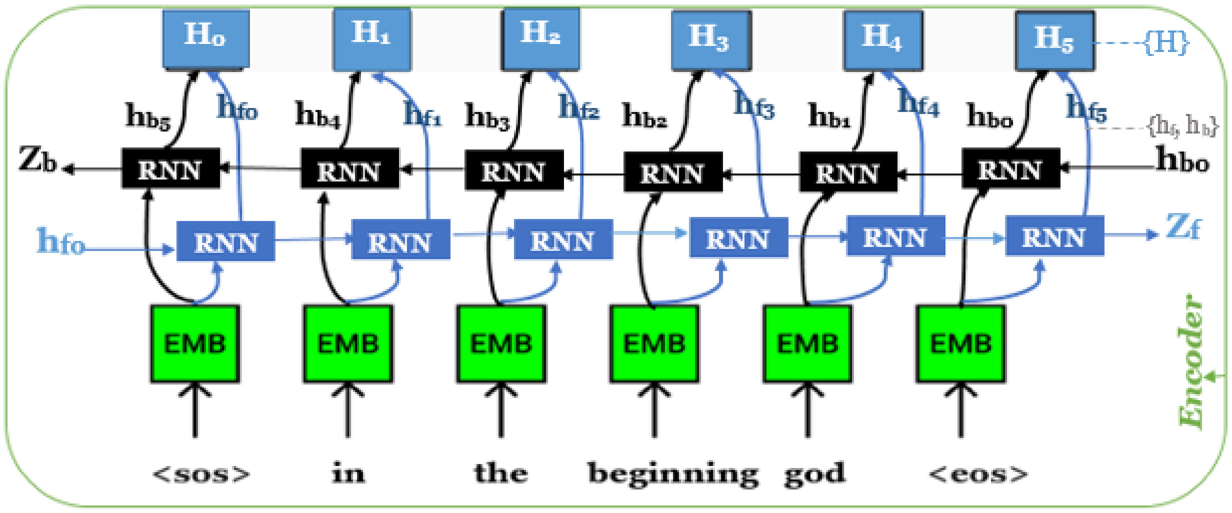


Figure 4:5 Bidirectional Encoder for RNNs based architecture of the proposed models

An architectural overview that visualized in Figure 4.5, their respective calculations are shown in Equation 4:3 and Equation 4:4.

$h_{ft} = \text{EncoderRNN}_f\,(e(x_{ft}), h_{ft-1})$ f→indicate forward *Equation 4:3*

$h_{bt} = \text{EncoderRNN}_b\,(e(x_{bt}), h_{bt-1})$ b→indicate backward *Equation 4:4*

t→At time step for both f and b

As before, we only pass an input (embedded) to the RNN, which tells PyTorch to initialize both the forward and backward initial hidden states (h_{f0} and h_{b0}, respectively) to a tensor of all zeros. We also get two context vectors, one from the forward RNN after it has seen the final word in the sentence (<eos>), Z_f= H_{Tf}, and one from the backward RNN after it has seen the first word in the sentence (<sos>), Z_b= H_{Tb}. The RNN returns outputs and hidden. The forward and backward hidden states concatenated together other, Here, we use element-wise sum to combine the forward and backward pass outputs. i.e. H_0=[hf_0; hb_5], H_1=[hf_1; hb_4], H_2=[hf_2; hb_3], H_3=[hf_3; hb_2], H_4 =[hf_4;

hb_1], H_5=[hf_5; hb_0] and we can denote all encoder hidden states (forward and backwards concatenated together) as H={H_0, H_1,...,H_T}, that could exploit long range context in both input direction. As the decoder is not bidirectional, it only needs a single context vector, z, to use as its initial hidden state, s_0, and we currently have two, a forward and a backward one (z_f=H_fT and z_b= H_{bT} , respectively). We solve this by concatenating the two context vectors together, passing them through a linear layer, *g*, and applying the tanh activation function is defined using the following equation 4:5.

$$Z = H = \tanh(g(H_fT, H_{bT})) = \tanh(g(z_f , z_b)) = s0 \quad \text{concatenating the two context} \quad \rightarrow \text{Equation 4:5}$$

As we want our model to look back over the whole of the source sentence we return outputs, the stacked forward and backward hidden states for every token in the source sentence. We also return hidden, which acts as our initial hidden state in the decoder.

4.1.1.1.1 Attention Layer

In broad terms, Attention is one component of a network's architecture. This is due to the fact that Attention was introduced to address the problem of long sequences in MT [4]. At every decoding step, the decoder will be informed how much attention needs to be paid to each input word using a set of attention weights. These attention weights provide contextual information to the decoder for translation**.** This allows the decoder network to focus on a different part of the encoder's outputs.

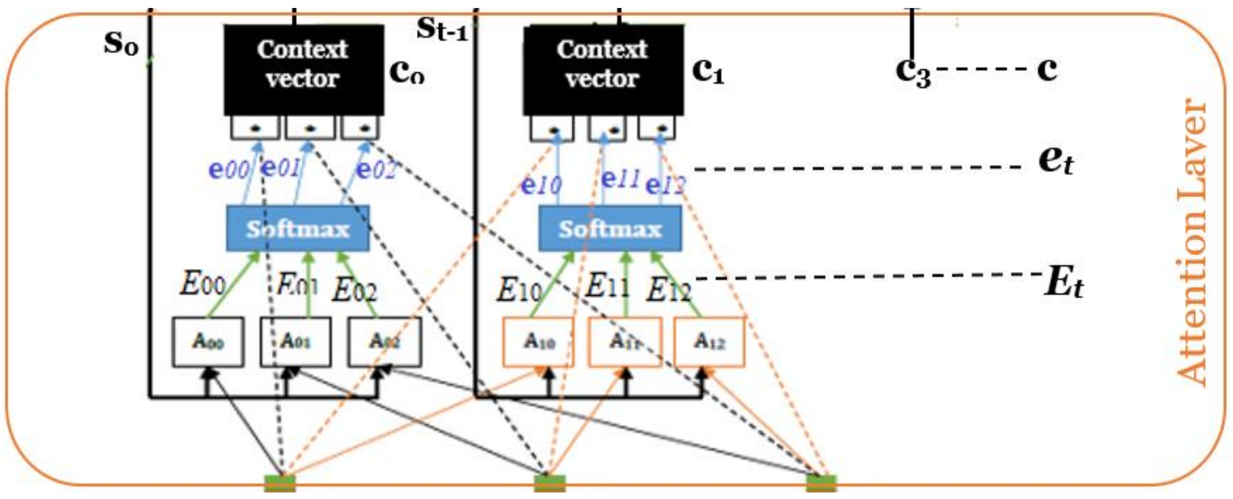


Figure 4:6 Attention layer Architecture for the RNN based of the proposed mnodels

As we seen on the figure 4:6, the attention layers contains three different process namely, calculating alignment score first part, softmaxing the alignment scores and calculating the context vector. Let as discuss as the follows:

4.1.1.1.1.1 Calculating Alignment Scores

First, we calculate the alignment score between the previous decoder hidden state and the encoder hidden states. It also simple single feed forward neural network. This we take in the previous hidden state of the decoder (s_{t-1}) and all of the stacked forward and backward hidden states from the encoder (Z). Thereafter, they added together before being passed through a *ReLu* activation function. We use *ReLu* activation function, because of the advantage of Relu as we compared under table 2.2. We then calculate the energy, ***E_t***, between them by concatenating them together and passing them through a linear layer (attn) and a Relu activation function.

$$\text{Score}_{\text{alignment}} = E_t = w_{alignment} . ReLu(w_{decoder_{t-1}} . s_{t-1} + w_{encodr_t} . H_{encoder})) \rightarrow \text{Equation 4:6}$$

The decoder hidden state and encoder outputs will be passed through their individual linear layer and have their own individual trainable weights. This can be thought of as calculating how well each encoder hidden state "matches" the previous decoder hidden state.

We can think of $w_{alignment}$ as the weights for a weighted sum of the energy across all encoder hidden states. These weights tell us how much we should attend to each token in the source sequence. The parameters of $w_{alignment}$ are initialized randomly, but learned with the rest of the model via backpropagation. $w_{alignment}$ is not dependent on time, and the same $w_{alignment}$ is used for each time-step of the decoding. We implement *v* as a linear layer without a bias. Lastly, the resultant vector from the previous few steps will undergo matrix multiplication with a trainable vector, obtaining a final alignment score vector which holds a score for each encoder output.

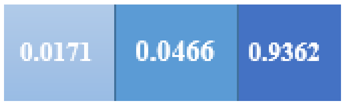
4.1.1.1.1.2 Softmaxing the Alignment Scores

After generating the alignment scores vector in the previous step, we can then apply a softmax on this vector to obtain the attention weights. The softmax function will cause the values in the vector to sum up to 1 and each individual value will lie between 0 and 1, therefore representing the weightage each input holds at that time step. The weights E_t are computed by a softmax function given by the following equation:

$$\text{Alignment scores} = \text{softmax}(E_t) = e_{ij} = \frac{exp^{e_{ij}}}{\sum_{K=1}^{T_x} exp^{e_{ij}}} \rightarrow \text{Equation 4:7}$$

The term on the above of the formula is the normalization term which ensures that all the output values of the function will sum to 1, thus constituting a valid probability distribution. If the input value is negative then also *softmax* returns positive value because exponent of any negative number is always gives positive values. Example let as take the aliment scores $e_{00} = 8$, $e_{01} = 9$, $e_{02} = 12$, Alignment score are softmaxed (8, 9, 12) =

$$\frac{exp^{8}}{exp^{8}+exp^{9}+exp^{12}} + \frac{exp^{9}}{exp^{8}+exp^{9}+exp^{12}} + \frac{exp^{12}}{exp^{8}+exp^{9}+exp^{12}} = 1$$

Attention weights = 

0.0171	0.0466	0.9362

So e_{00} =0.0171, e_{01} =0.0466, e_{02} =0.9362

4.1.1.1.1.3 Calculating the Context Vector

After computing the attention weights in the previous step, we can now generate the context vector by doing an element-wise multiplication of the attention weights with the encoder outputs. Due to the softmax function in the previous step, if the score of a specific input element is closer to 1 its effect and influence on the decoder output is amplified, whereas if the score is close to 0, its influence is drowned out and nullified.

$$context\ vector = c_i = \sum_{j=1}^{T_x} e_{ij} H_j$$ Equation 4:8

The context vector c_i for the output word y_i is generated. The process repeat**s** itself for each time step of the decoder until an token is produced or output is past the specified maximum length.

Eg. $C_0 = (e_{00} \text{ x } H_0 + e_{01} \text{ x } H_1 + e_{02} \text{ x } H_2 + \ldots\ldots + E_{05} + H_5)$ this step is repeats until to end of sequence. This concept is similar to Key, Query and Value, we explain later under transformer section.

Key is the hidden state of the encoder like, H_0, H_1, H_2, H_3, ……H.

Query is the decoder hidden state from previous time step, like S_0, S_1, S_2, …., S_{t-1} and

Value is normalized weight representation, like how much attention *key* can get.

4.1.1.1.2 Decoder for RNN based models

The decoder contains the attention layer, the context vector is concatenated with the previous decoder output and fed into the Decoder RNN for that time step along with the previous decoder hidden state to produce a new output**.** The Figure 4:7 shows decoding the first and second word in an example translation.

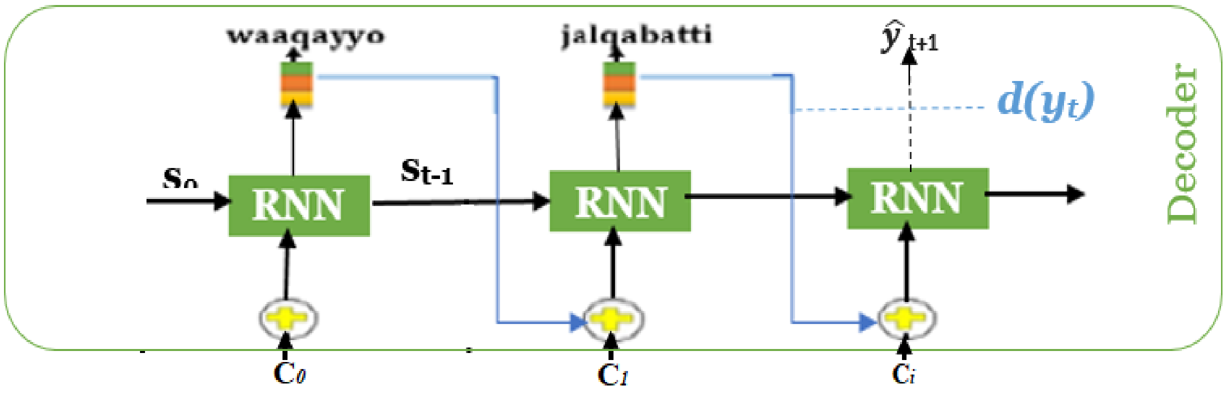


Figure 4:7 Architecture for the Decoder RNN based of the proposed models

The embedded input word, $d(y_t)$, the weighted source vector, C_i, and the previous decoder hidden state, s_{t-1}, are then all passed into the decoder RNN, with $d(y_t)$ and C_i being concatenated together.

s_t = DecoderRNN $(d(y_t),C_t, s_{t-1})$ → Equation 4:9

We then pass $d(y_t)$, C_t and s_t through the linear layer, f, to make a prediction of the next word in the target sentence, $\hat{Y}_{t+1}$. This is done by concatenating them all together.

$\hat{Y}_{t+1} = f(d(y_t),C_t,s_t)$ → Equation 4:10

Generally the decoder also does a single step at a time. The Context Vector from the Encoder block is provided as the hidden state (hs) and cell state (cs) for the decoder's first RNN block. The start of the sentence "<sos>" token is passed to the embedding NN, then passed to the first RNN cell of the decoder, and finally, it is passed through a linear layer, which provides an output Afaan Oromo token prediction probabilities, hidden state (hs), cell state (cs).The output word with the highest probability is chosen, hidden state (hs), cell state (cs) is passed as the inputs to the next RNN cell and this process is executed until it reaches the end of sentences "<eos>".The subsequent layers will use the hidden and cell state from the previous time steps. The above visualization is applicable for a single sentence from a batch. Say we have a batch size of 128 (Experimental), then pass 128 sentences at a time to the Encoder, which provide 128 sets of context vectors, and they all are passed into the Decoder. This is all about our RNN based proposed system architecture, next we can explain Transformer based architecture for proposed model.

4.1.4 Transformers Model Architecture

Transformer has two main segments, the first is an encoder that operates primarily on the input sequence and the second is a decoder that operates on the target output sequence during training and predicts the next item in the sequence. Figure 4:8 demonstrates how a transformer is put together, with the encoder on the left and the decoder on the right.

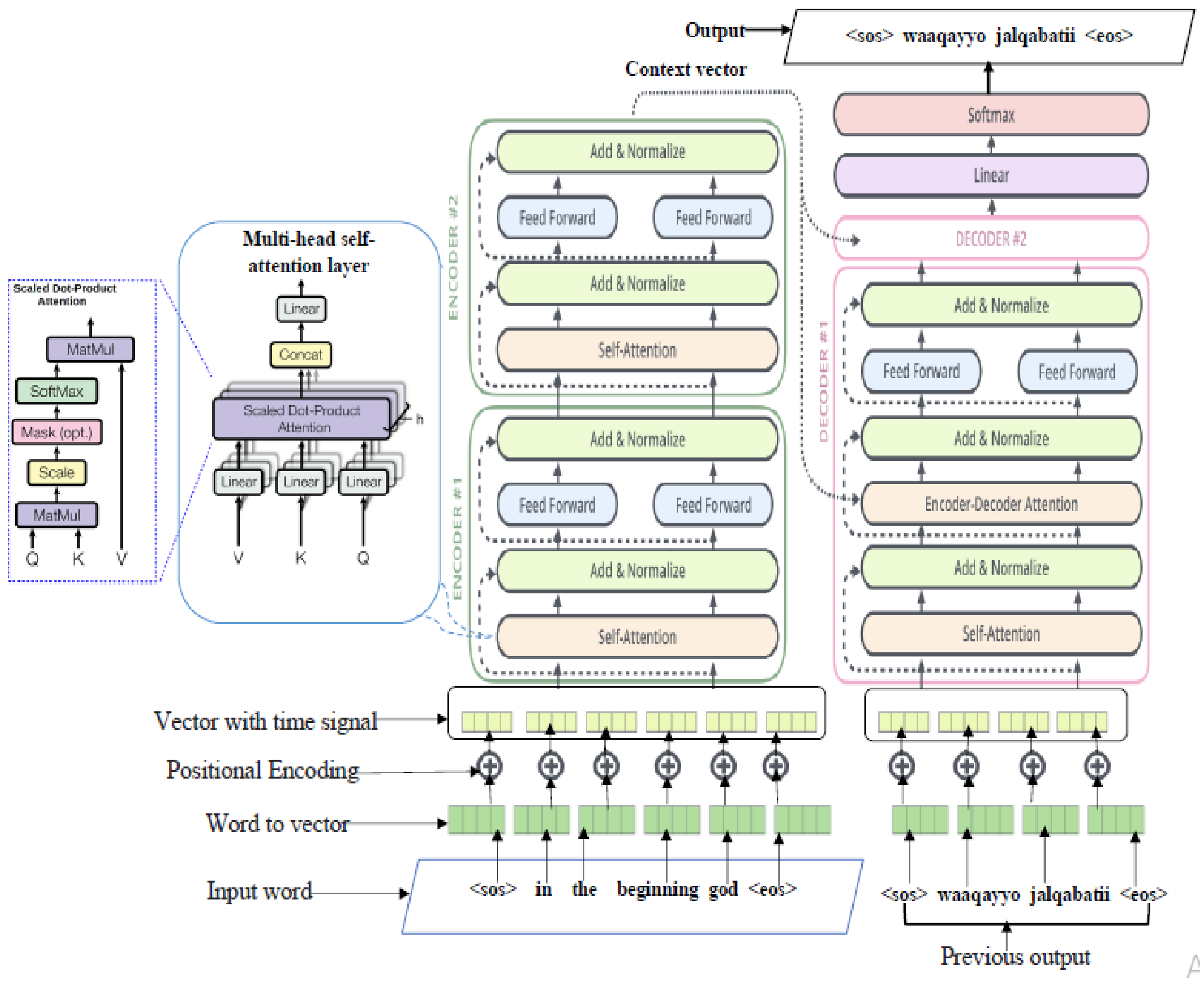


Figure 4:8 Transformer based architecture of the proposed model

The encoder and decoder blocks are actually multiple identical encoders and decoders stacked on top of each other. In addition to the self-attention and feed-forward layers. Attention layer helps the decoder focus on the appropriate parts of the input sequence. Let's see and discuss how this setup of the encoder and the decoder stack works.

4.1.4.1 Input and Output Embedding

Feeding input is the same to RNN embedding layers. But the Transformer handles the entire input sequence in at once and doesn't iterate word by word. The embedding layer takes a sequence of words and learns a vector representation for each word like as we show in figure 4:8.

4.1.4.2 Positional Encoding

First, the tokens are passed through a standard embedding layer. Next, as the model has no recurrent it has no idea about the order of the tokens within the sequence. We solve this by using a second embedding layer called a positional embedding layer. Since our model contains no recurrence, in order for the model to make use of the order of the sequence, we must inject some information about the relative or absolute position of the tokens in the sequence.

Equation 4:11

$P_t \in R^{d_{model}}$ → positional encoding

Equation 4:12

$P_{os(t),\ 2i} = sin(\frac{pos(t)}{10000^{\frac{2i}{d_{model}}}})$

→ For every even index, create a vector using the *sin* function.

Equation 4:13

$\mathrm{P_{os(t),\ 2i+1}} = \cos(\frac{pos(t)}{10000^{\frac{2i}{d_{model}}}})$

→ For every odd index on the input vector, create a vector using the *cos* function.

Equation 4:14

$PE = e_t + P_{os(t)} \in R^{d_{model}}$ → PE indicate positional embedding

→Then add those vectors to their corresponding input embedding.

→ e_t (input embedding matrix) explained under equitation 4:2

This allows us to mathematically represent the relative position of word vectors such that a NN can learn to recognize differences in position. The positional encoding matrix is a constant whose values are defined by a function *(pos(t), i),* where *pos(t)* is the position of the word in the sentences, and *i* is the dimension. When these position specific values are added to our embedding values, each word embedding is altered in a way specific to its position in the sentence. The network is hence given information about structure, and it can use this to build understanding of the languages. The sin and cosine functions were chosen in tandem because they have linear properties the model can easily learn to attend to. The value of PE range between -1 and 1.

4.1.4.3 The Encoder

The encoder is the part of the transformer algorithm that chooses what parts of the input to focus on. The Transformer's encoder does not attempt to compress the entire source sentence, X=(x_1,...,x_n), into a single context vector, Z. Instead it produces a sequence of context vectors, Z=(z_1,...,z_n). So, if our input sequence was 6 tokens long as we seen in the above figure 4:9 we would have Z=(z_1,z_2,z_3,z_4,z_5, z_6). The token and positional embedding are elementwise summed together to get a vector which contains information about the token and also its position within the sequence. First pass the input tokens and its mask into the multi-head attention layer, then perform dropout on it, apply a residual connection and pass it through a layer normalization layer. We then pass it through a position-wise feedforward layer and then, again, apply dropout, a residual connection and then layer normalization to get the output of this layer which is fed into the next layer. The multi-head attention layer is used by the encoder layer to attend to the source sentence. It is calculating and applying attention over itself instead of another sequence, hence we call it self-attention. Let us seen each part of encoder as follows:

4.1.4.3.1 Multi-Head Attention layer

The multi-head attention block is the main innovation behind transformers [68]. The question that the attention block aims to answer is what parts of the text should the model focus on? We must understand how we can calculate self-attention. First, we need to create three vectors from each of the encoder’s input vectors, it is name Query Vector, Key Vector and Value Vector. These vectors are trained and updated during the training process. What are the “query”, “key”, and “value” vectors?

Query vector: Represented by a word vector in the sequence and defines the hidden state of the decoder.

Key vector: Represented by all the words in the sequence and defines the hidden state of the encoder.

Value vector: Represented by all the words in the sequence and defines the attention weights of the encoder hidden states.

As you can see in figure 4:8, the encoder and the decoder apply self-attention separately to the source and the target sequences respectively. The Transformer uses scaled dot-product attention, where the query and key are combined by taking the dot product between them, then applying the softmax operation and scaling by d_k before finally then multiplying by the value. d_k is the head dimension. The final output of the attention block is defined using the following equation.

$$Attention\ (Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \qquad \rightarrow \textit{Equation 4:15}$$

One thing that looks strange at first is that dropout is applied directly to the attention. This means that our attention vector will most probably not sum to 1 and we may pay full attention to a token but the attention over that token is set to 0 by dropout. Let we explain what actually is happening in self-attention mechanism.

If ($X_1, X_2, \ldots X_n$) → input sequence like sentence *("<sos> in the beginning god <eos>")*

The first step is to calculate the Query, Key, and Value matrices. We do that by packing our embeddings into a matrix X, and multiplying it by the weight matrices we've trained (W_Q for weight of query, W_K, for weight of key and W_V for weight of value).

e.g X_0 = *<sos>* X_1 = *in* , X_2 =*the* , X_3 =*beginning* , X_4 = *god* ... X_5 = *<eos>*

$$X_n \text{ x } W_Q = q_n,\ X_n \text{ x } W_k = k_n,\ X_n \text{ x } Wv = v_n$$

The score for the first word is calculated by taking the dot product of the Query vector (q1) with the keys vectors (k1, k2, k3) of all the words

Multiplying X_1 by the WQ weight matrix produces q1, the "query" vector associated with that word. We end up creating a "query", a "key", and a "value" projection of each word in the input sentence. So for each word, we create a Query vector (Q), a Key vector (K), and a Value vector (V). These vectors are created by multiplying the embedding by three matrices that we trained during the training process. The new vectors are smaller in dimension than the embedding vector. Their dimensionality is 64, while the embedding and encoder input/output vectors have dimensionality of 512.

The **second step** in calculating self-attention is to calculate a score. Say we're calculating the self-attention for the first word in this example, *"<sos>"*. We need to score each word of the input sentence against this word. The score determines how much focus to place on other parts of the input sentence as we encode a word at a certain position. The score is calculated by taking the dot product of the query vector with the key vector of the respective word we're scoring. So if we're processing the self-attention for the word in position 1, the first score would be the dot product of q1 and k1. The second score would be the dot product of q1 and k2. The third score would be the dot product of q1 and k2 until to q1 and kn, on figure 4:8, n = 6, because we have only six input words on. Let as we say the output score X_1 = 112, X_2 = 96.

The **third and fourth steps** are to divide the scores by $\sqrt{d_k}$ (the square root of the dimension of the key vectors. This is leads to having more stable gradients.

For example, let as we say d_k= 64, X_1 = 112, X_2 = 96.

$X_1 = \frac{112}{\sqrt{64}}, \frac{112}{8} = 14$, $X_2 = \frac{96}{\sqrt{64}}, \frac{96}{8} = 12$

Scaled Scores pass them through the **softmax** layer in order to normalize the scores and to get the attention weights, which gives you probability values between 0 and 1. By doing a softmax the higher scores get heighten, and lower scores are depressed. This allows the model to be more confident about which words to attend too. We use equation 4:5 in order to do this step:

Softmax = $\frac{exp^{14}}{exp^{14}+exp^{12}} = 0.88, \frac{exp^{12}}{exp^{14}+exp^{12}} = 0.12$

This softmax score determines how much each word will be expressed at this position. Clearly the word at this position will have the highest softmax score, but sometimes it’s useful to attend to another word that is relevant to the current word.

The **fifth step** is to multiply each value vector by the softmax score. The intuition here is to keep intact the values of the word we want to focus on, and drown-out irrelevant words. That means as example => 0.88 x v_1 and 0.12 x v_2

The **sixth step** is to sum up the weighted value vectors. This produces the output (z_1) of the self-attention layer at this position (for the first word). This is the output of the self-attention layer. The higher softmax scores will keep the value of words the model learns is more important. The lower scores will drown out the irrelevant words. Then you feed the output of that into a linear layer to process. We can get the vectors for the rest of each tokens in the input sequence in the same fashion.

However, the scaled dot-product attention isn't simply applied to the queries, keys and values. Instead of doing a single attention application the queries, keys and values have their hidden dimension split into h heads and the scaled dot-product attention is calculated over all heads in parallel. This means instead of paying attention to one concept per attention application, we pay attention to h. We then re-combine the heads into their hidden dimension shape, thus each hidden dimension potentially paying attention to h different concepts.

$MultiHead(Q, K, V) = Concat(head_0, head_2, \dots head_h)W^0$ →Equation 4:16

For example out head =8 so, z_0 , z_1, …….z_7 . So we need a way to condense these eight down into a single matrix. Then concatenate all the attention heads and multiply with a weight matrix (W^0) that trained jointly with the model. The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN. That’s pretty much all there is to multi-headed self-attention. Finally the multi-headed attention output vector is added to the original positional input embedding.

4.1.4.3.2 Add & Normalization

This layer simply takes the outputs from the multi-head attention block, adds them together, and normalizes the result with layer normalization. If you have heard of batch normalization, layer normalization is similar but instead of normalizing the input features across the batch dimensions, it normalizes the inputs to a layer across all features.

4.1.4.3.3 Feed-Forward Layer

The other main block inside the encoder layer is the position-wise feedforward layer. This is relatively simple compared to the multi-head attention layer. The input is transformed from hidden dimension (hid_dim) to position-wise feedforward dimension (pf_dim), where pf_dim is usually a lot larger than hid_dim. The ReLU activation function and dropout are applied before it is transformed back into a hidden dimension representation. This feed-forward layer receives attention vectors from the Multi-Head Attention. We apply normalization to transform it into a single Attention vector. Thus, we get a single vector is digestible by the next encoder block or decoder block.

4.1.4.4 The Decoder

As we describe under RNN, decoder’s is to generate text sequences. These sub-layers behave similarly to the layers in the encoder but each multi-headed attention layer has a different job. It creates its Queries matrix from the layer below it, and takes the Keys and Values matrix from the output of the encoder stack. The encoder start by processing the input sequence. 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. The output of each step is fed to the bottom decoder in the next time step, and the decoders bubble up their decoding results just like the encoders did. And just like we did with the encoder inputs, we embed and add positional encoding to those decoder inputs to indicate the position of each word. The self-attention layers in the decoder operate in a slightly different way than the one in the encoder. In the decoder, the self-attention layer is only allowed to attend to earlier positions in the output sequence. This is done by masking future positions before the softmax step in the self-attention calculation. The above steps repeat the process until a special symbol is reached indicating the transformer decoder has completed its output. The decoder stops decoding when it generates a token as an output.

4.1.4.4.1 Decoder Multi-Head Attention

Unlike the Attention block in the Encoder that receiving every word in the English sentence, only the previous words of the Afaan Oromo sentence are fed into this Decoder’s Attention block.

4.1.4.4.2 The Decoder Final Linear and Softmax Layer

As the purpose of the decoder is to predict the output word, as we seen on the figure 4:8, the output size of this feed-forward layer is the number of Afaan Oromo words in the vocabulary. Softmax transforms the output into a probability distribution, which outputs a word corresponding to the highest probability of the next word. That means, the output of the final pointwise feedforward layer goes through a final linear layer that acts as a classifier. The classifier is as big as the number of vocabulary in target language that you have. For example, if you have 1000 classes for 1000 words, the output of that classier will be of size 1000. The output of the classifier then gets fed into a softmax layer, which will produce probability scores between 0 and 1. The decoder then takes the output, add it to the list of decoder inputs, and continues decoding again until a token is predicted or until to end of sentence token (<eos>) and that’s is all about the mechanics our transformer encoder-decoder seq2seq model architecture. Transformers leverage the power of the attention mechanism to make better predictions. Recurrent Neural networks try to achieve similar things.

CHAPTER FIVE
EXPERIMENT AND RESULT ANALYSIS

Under this chapter we perform the experiment and discusses the experimental results of this thesis by showing the experimental setups and show testing results of the experimental systems. Also, it gives comparison of the results among all proposed models on both directions of translation between the language pair and discuss the results. The experiments were carried out on the NMT task for the English and Afaan Oromo language and vice versa. Finally we develop a prototype for best model result and demonstrate the prototype for user in order to evaluate the translated sentences quality and UI usability.

5.1 Systems Environment

To implement the various proposed models, first we have to prepare and setup the required environment. To conduct this research, models training and evaluation has been done using Google Cloud GPUs. That is freely available when using Google Colab Notebooks, which helps to keep costs down. These GPUs can be used to accelerate specific workloads on virtual machine. They reduce training time compared to CPUs and local machine (PC). For storage of the dataset and models checkpoints Google Cloud Storage was used as this was required for training on GPUs and as Google Colab does not provide persistent storage. Google Colab is a cloud based data science work space provided by Google for free. Colab Notebooks is an open source web application that you can use to create, save and share documents that contain live code, equations, visualizations, and text, all for free from your browser [88]. It also runs on Google servers and you don't need to install anything. Moreover, the notebooks are saved to your google drive storage. All proposed systems were run and tested as well as data preprocessing has been done on Google Colab 12 GB RAM online resource. After setting up the systems environment, the next step is to start coding to investigate corpus.

And also used Toshiba laptop with:

- ✓ Processor: Intel® Core™ i3-2348M CPU @ 2.30GHz 2.30 GHz
- ✓ Installed memory (RAM): 4.00GB
- ✓ System type: 64 –bit operating system, x64-based processor

We used the laptop for corpus preparation and preprocessing and for writing the thesis document and accessing colab notebook. In addition, MS Word and PPT are used for writing a report and presentation slides, and also, we used MS excel and notepad^{++} for text processing and parallel corpus preparation , and Google Chrome browser is used for running Google's Co-laboratory scripts for this study.

5.2 Programming Language Used for the Study

All the required libraries have to be installed and imported before starting to write code. We have chosen Python 3.7.13 programming language to implement the proposed systems. Python programming language supports different set of freely available library. The model of the system is built by using torch tensor. Torch is freely available library for DL algorithm [89].

5.3 Dataset

The initial requirement for setting up a MT system is the availability of parallel corpus for source and target languages. In this step we try to investigate the parallel corpus that we used in training the models as well as evaluate the models. The experiment described in this thesis are carried out using the following corpus statics for English to Afaan Oromo and vice versa. When preparing our model, we need to know the statistics of the dataset. While applying cleaning code, total sentences are reduced from 21466 to 21323. The size and distribution details of dataset are given in the table 5:1 shown below.

Table 5:1 Details of Dataset Statistics.

Descriptions	**English sentences**	**Afaan Oromo sentences**
Total numbers of sentences	21323	21323
Max-sentences length	66	55
Average sentences length	23	20
Total numbers of Tokens (words)	498430	414899
Number of unique Tokens (words)	14625	32474
Totals numbers of characters	2561439	2083410
Min-character length	1	2
Max-character length of sentences	251	324
Average characters length of sentences	98	120

After details of dataset statistics are showed, the next step can be to visualize some features of dataset and plot word cloud. Visualizing data is one of the best ways to humanize data to make it easy to understand and get the relevant trends from it. This activity can be crucial when the user is still trying to optimize the model and make it production ready. Sentence length refers to the number of words in a sentence. Most readability formulas use the number of words in a sentence to measure its difficulty. Figure 5:1 plot the length of sentence and density in the language pairs.

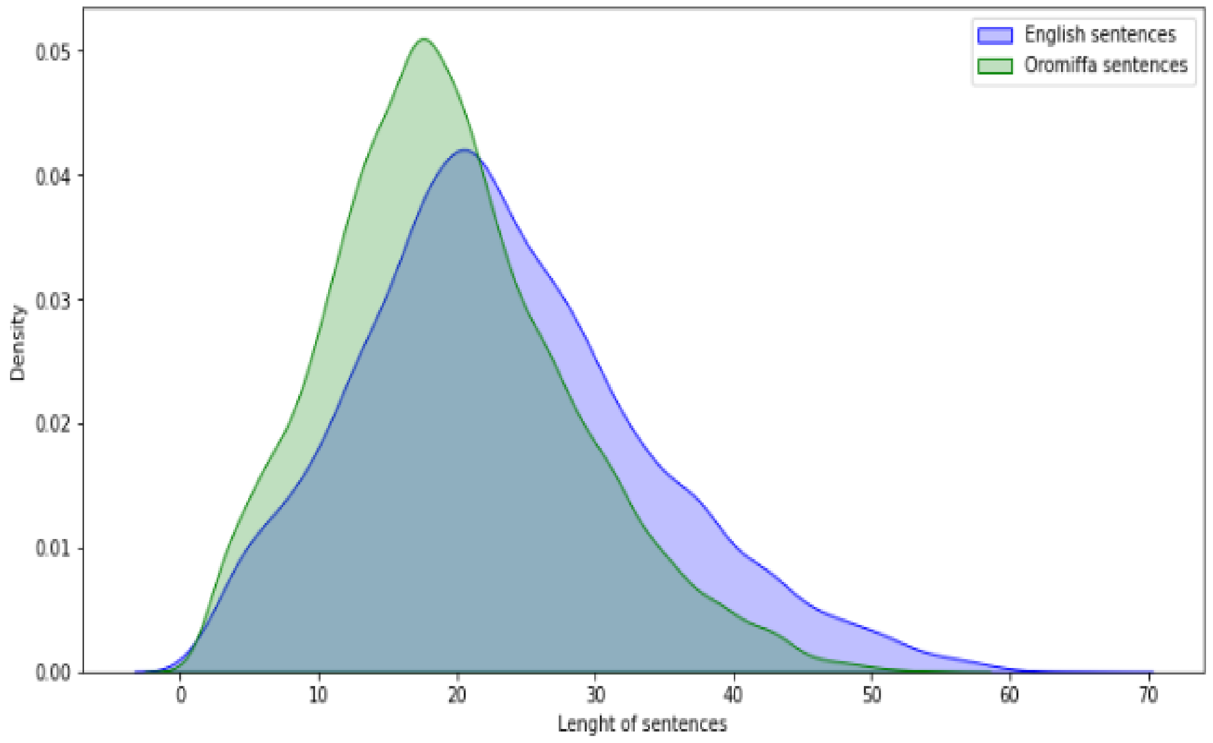


Figure 5:1 Plot the Length of Sentence and Density for Language Pairs

Note lot of sentences are of 8 to 28 length in Afaan Oromo and 10 to 33 in English.

In addition, Word clouds are graphical representations of word frequency that give greater prominence to words that appear more frequently in a source text. The larger the word in the visual the more common the word was in the dataset as described in figure 5:2.

English Words
Afaan Oromo Words

Figure 5:2 Graphical representations of most frequent words in both Language.

5.4 Training Details

This section describes the training regime for our models. After saving the cleaned pairs of sentences, the next step will be to randomly split this data into three categories. These are, training dataset, validation dataset and testing dataset, and saving the split portions of dataset to new files as we described under section 4.1.2 and then, build the vocabulary for the source and target languages. The vocabularies of the source and target languages are distinct. Our vocabulary is small when it's compared against vocabularies of other European language pairs which have large datasets. The size of our training and evaluation dataset are given in the table 5:2.

Table 5:2 Training details for datasets

Datasets	Numbers of datasets	Remark
Total dataset	21322	Total cleaned dataset
Training dataset	17057	80% of total dataset (random-state=32)
Validation dataset	2132	10% of total dataset (random-state=32)
Test dataset	2133	10% of total dataset (random-state=32)
English vocabulary	8625	Using the minimum frequency argument, we only allow tokens that appear at least 2 times to appear in our vocabulary.
Afaan Oromo vocabulary	15253	

After setting the language dataset, the next step is to create batches of training, testing and validation dataset using iterators as we discussed under 4.1.2 section. Sample codes for creating the iterators are included under *Appendix G*. When we get a batch of examples using an iterator we need to make sure that all of the source sentences are padded to the same length, the same with the target sentences.

In the next step, we define the models, with the following hyper-parameters. Since we are working with less dataset, we can make our models smaller as well. As a training of NN needs the adjustment of model's hyper-parameters, it can make a significant impact on the performance of the models. There are no efficient algorithms to select optimal values of hyper-parameters. So optimal values of hyper-parameters are determined using a trial and error process by adjusting or adapting their values for a particular ML task. Different hyper-parameters and methods have been suggested in several literatures in the field of NMT task. But in order to select the best hyper-parameters for our domain of study we follow the suggestion of those model using the hyper-parameters suggested by Joey NMT and OpenNM. Those hyper-parameters have been confirmed by [90], [91] using different datasets. Dropout is a regularization method used to prevent over-fitting by deactivating units in a NN and Adam optimizer requires less memory [68], [69]. All of our proposed model's hyper-parameters are shown in Table 5:3

Table 5:3 Hyper-parameters and values of the proposed MT models

No	Hyper-parameters	Model type		
		GRU	**LSTM**	**Transformer**
1	Encoder direction	Bidirectional	Bidirectional	Unidirectional
2	Decoder direction	Unidirectional	Unidirectional	Unidirectional
3	Encoder layers	2	2	3
4	Decoder layers	1	1	3
5	Encoder embedding dimension	256	256	256
6	Decoder embedding dimension	256	256	256
7	Encoder Hidden dimension	512	512	512
9	Decoder Hidden dimension	512	512	512
10	Encoder position forward dimension	_	_	512
11	Decoder position forward dimension	_	_	512
12	Encoder heads	_	_	8
13	Decoder heads	_	_	8
14	Encoder dropout	0.1	0.1	0.1
15	Decoder dropout	0.1	0.1	0.1
16	Learning rate	0.0003	0.0003	0.0003
17	Optimization	Adam optimizer	Adam optimizer	Adam optimizer
18	The choice of loss function	Cross-entropy loss	Cross-entropy loss	Cross-entropy loss
19	Number of iterations (epochs)	30	30	30
20	Batch size	128	128	128
21	Maximum prediction length	70	70	70

We also define a function that calculate the number of trainable parameters and it print out for all proposed models. The ML model parameters determine how input data is transformed into the desired output [92]. This is required by the model when making predictions, whereas the hyper-parameters control the model's shape. Almost all standard learning methods contain hyper-parameter attributes that must be initialized before the model can be trained.

It is also adjustable parameters that must be tuned in order to obtain a model with optimal performance. A model parameter depends on the selected model, which is chosen for the training.

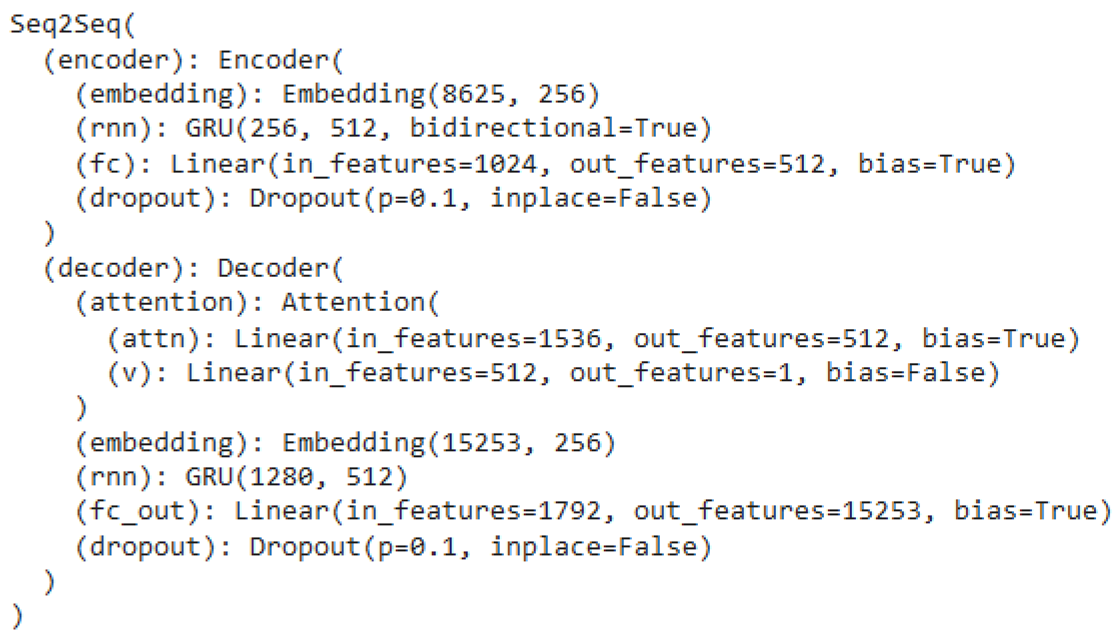
It is really important matrix for understanding the whole model complexity. As training progresses the initial values are updated using an optimization algorithm. The learning algorithm is continuously updating the parameter values as learning progress.

Table 5:4 The number of trainable parameters in the proposed models.

Models	English to Afaan Oromo	Afaan Oromo to English
	Trainable Parameters	**Trainable Parameters**
Bi-encoder and decoder LSTM with Attention model	21,816,726	18,416,562
Bi-encoder and decoder GRU with Attention model	39,894,677	28,010,678
Transformer	14,022,293	12,318,897

Note that all models have small number of parameters, because we don't have enough corpus.

It is possible to increase the number of parameters, if you have more corpus. Summary of GRU model configuration of the proposed model is depicted as sample in figure 5:3.



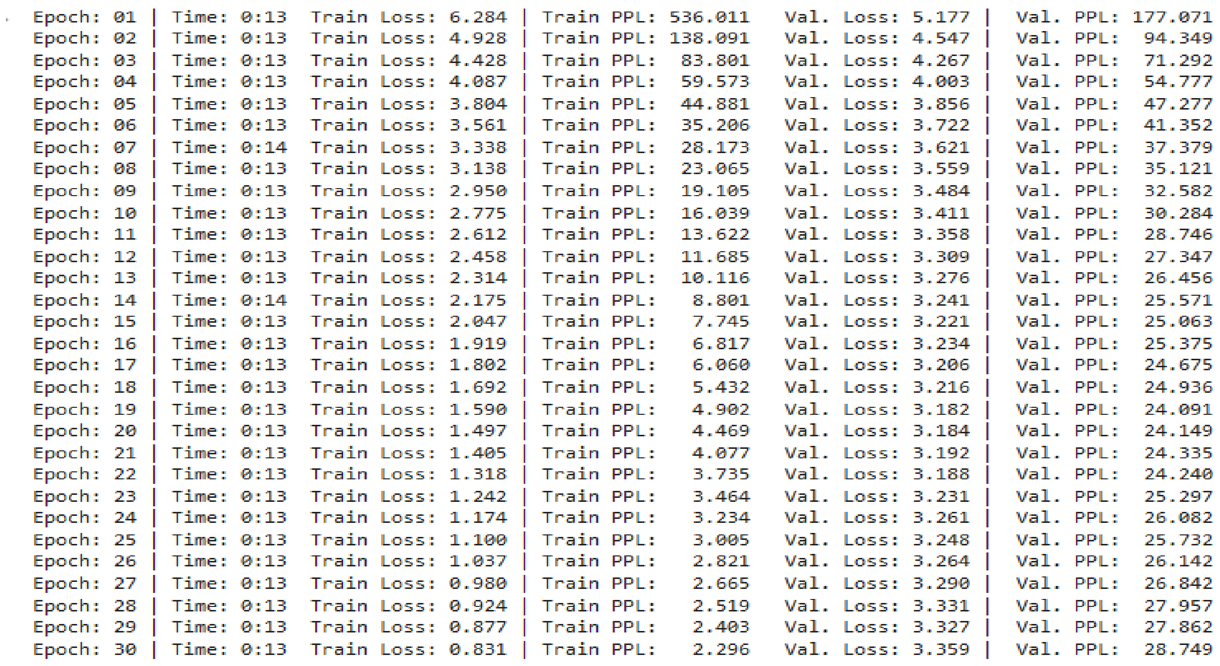
```
Seq2Seq(
  (encoder): Encoder(
    (embedding): Embedding(8625, 256)
    (rnn): GRU(256, 512, bidirectional=True)
    (fc): Linear(in_features=1024, out_features=512, bias=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
  (decoder): Decoder(
    (attention): Attention(
      (attn): Linear(in_features=1536, out_features=512, bias=True)
      (v): Linear(in_features=512, out_features=1, bias=False)
    )
    (embedding): Embedding(15253, 256)
    (rnn): GRU(1280, 512)
    (fc_out): Linear(in_features=1792, out_features=15253, bias=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
```

Figure 5:3 Sample snapshot for summary Bi-encoder and decoder GRU with Attention model

5.5 Experimental Results

At this phase, we train each proposed models by using training dataset and testing the models by using testing dataset. In every iteration of training, the models learns by extracting patterns from the training dataset. That means, the models sees and learns from this dataset. After training, it use the validation dataset to check the performance of the models during training**.** Hence the model occasionally sees this dataset, but never does it learn from this dataset and also it provides the first test against unseen data, allowing how well the model makes predictions based on the new data. At the end, when the training is finished, we use the test dataset to check the final performance of the models. If our highest result is less than ten (< 10), we modify the hyper-parameters and continue training using the training dataset again. In order to make the comparison of all model we try to set balanced hyper-parameters as we define in the Table 5:3 and we use the same dataset for all experiment as we describe in Table 5:2, which we place on the Google Colab GPU.

As described in Figure 5:4, during training we calculated training Loss, validation loss, training perplexity and validation perplexity score and also, we have created a function that tells us how long an epoch takes for time visualization. It is easier to see a change in perplexity than a change in loss as the numbers are much bigger. The scores on the validation set express how well your model is generalizing to unseen data.



```
Epoch: 01 | Time: 0:13  Train Loss: 6.284 | Train PPL: 536.011   Val. Loss: 5.177 |  Val. PPL: 177.071
Epoch: 02 | Time: 0:13  Train Loss: 4.928 | Train PPL: 138.091   Val. Loss: 4.547 |  Val. PPL:  94.349
Epoch: 03 | Time: 0:13  Train Loss: 4.428 | Train PPL:  83.801   Val. Loss: 4.267 |  Val. PPL:  71.292
Epoch: 04 | Time: 0:13  Train Loss: 4.087 | Train PPL:  59.573   Val. Loss: 4.003 |  Val. PPL:  54.777
Epoch: 05 | Time: 0:13  Train Loss: 3.804 | Train PPL:  44.881   Val. Loss: 3.856 |  Val. PPL:  47.277
Epoch: 06 | Time: 0:13  Train Loss: 3.561 | Train PPL:  35.206   Val. Loss: 3.722 |  Val. PPL:  41.352
Epoch: 07 | Time: 0:14  Train Loss: 3.338 | Train PPL:  28.173   Val. Loss: 3.621 |  Val. PPL:  37.379
Epoch: 08 | Time: 0:13  Train Loss: 3.138 | Train PPL:  23.065   Val. Loss: 3.559 |  Val. PPL:  35.121
Epoch: 09 | Time: 0:13  Train Loss: 2.950 | Train PPL:  19.105   Val. Loss: 3.484 |  Val. PPL:  32.582
Epoch: 10 | Time: 0:13  Train Loss: 2.775 | Train PPL:  16.039   Val. Loss: 3.411 |  Val. PPL:  30.284
Epoch: 11 | Time: 0:13  Train Loss: 2.612 | Train PPL:  13.622   Val. Loss: 3.358 |  Val. PPL:  28.746
Epoch: 12 | Time: 0:13  Train Loss: 2.458 | Train PPL:  11.685   Val. Loss: 3.309 |  Val. PPL:  27.347
Epoch: 13 | Time: 0:13  Train Loss: 2.314 | Train PPL:  10.116   Val. Loss: 3.276 |  Val. PPL:  26.456
Epoch: 14 | Time: 0:14  Train Loss: 2.175 | Train PPL:   8.801   Val. Loss: 3.241 |  Val. PPL:  25.571
Epoch: 15 | Time: 0:13  Train Loss: 2.047 | Train PPL:   7.745   Val. Loss: 3.221 |  Val. PPL:  25.063
Epoch: 16 | Time: 0:13  Train Loss: 1.919 | Train PPL:   6.817   Val. Loss: 3.234 |  Val. PPL:  25.375
Epoch: 17 | Time: 0:13  Train Loss: 1.802 | Train PPL:   6.060   Val. Loss: 3.206 |  Val. PPL:  24.675
Epoch: 18 | Time: 0:13  Train Loss: 1.692 | Train PPL:   5.432   Val. Loss: 3.216 |  Val. PPL:  24.936
Epoch: 19 | Time: 0:13  Train Loss: 1.590 | Train PPL:   4.902   Val. Loss: 3.182 |  Val. PPL:  24.091
Epoch: 20 | Time: 0:13  Train Loss: 1.497 | Train PPL:   4.469   Val. Loss: 3.184 |  Val. PPL:  24.149
Epoch: 21 | Time: 0:13  Train Loss: 1.405 | Train PPL:   4.077   Val. Loss: 3.192 |  Val. PPL:  24.335
Epoch: 22 | Time: 0:13  Train Loss: 1.318 | Train PPL:   3.735   Val. Loss: 3.188 |  Val. PPL:  24.240
Epoch: 23 | Time: 0:13  Train Loss: 1.242 | Train PPL:   3.464   Val. Loss: 3.231 |  Val. PPL:  25.297
Epoch: 24 | Time: 0:13  Train Loss: 1.174 | Train PPL:   3.234   Val. Loss: 3.261 |  Val. PPL:  26.082
Epoch: 25 | Time: 0:13  Train Loss: 1.100 | Train PPL:   3.005   Val. Loss: 3.248 |  Val. PPL:  25.732
Epoch: 26 | Time: 0:13  Train Loss: 1.037 | Train PPL:   2.821   Val. Loss: 3.264 |  Val. PPL:  26.142
Epoch: 27 | Time: 0:13  Train Loss: 0.980 | Train PPL:   2.665   Val. Loss: 3.290 |  Val. PPL:  26.842
Epoch: 28 | Time: 0:13  Train Loss: 0.924 | Train PPL:   2.519   Val. Loss: 3.331 |  Val. PPL:  27.957
Epoch: 29 | Time: 0:13  Train Loss: 0.877 | Train PPL:   2.403   Val. Loss: 3.327 |  Val. PPL:  27.862
Epoch: 30 | Time: 0:13  Train Loss: 0.831 | Train PPL:   2.296   Val. Loss: 3.359 |  Val. PPL:  28.749
```

Figure 5:4 Snapshot for sample report during model training

This report printing over a number of epochs is a good way to determine if the model has been sufficiently trained and to optimize these results in-order to build a better model. For every time of training dataset, it helps to increase the performance. The performance of the model depends on the validation perplexity results. That means if the result of validation perplexity decreases, the models performance increases. This is important to identify that the model is not either undertrained or over trained. At each epoch, we check if proposed models has achieved the best validation loss so far. If it has, we update best validation loss and save the parameters of our model. The validation loss of these experimental results are depicted furthermore in graph figure 5:5.

At times, the efficacy of a models is judged by its ability to perform well on an unseen dataset and not by its performance on the training dataset fed to it.

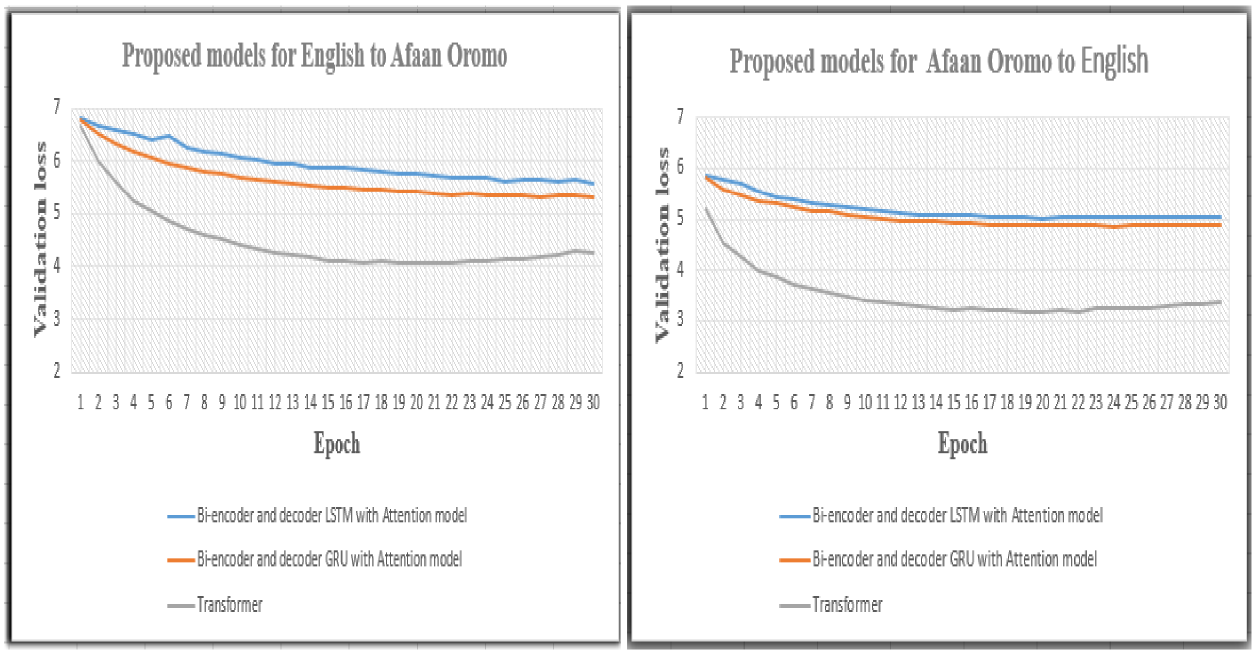


Figure 5:5 Validation loss of proposed models

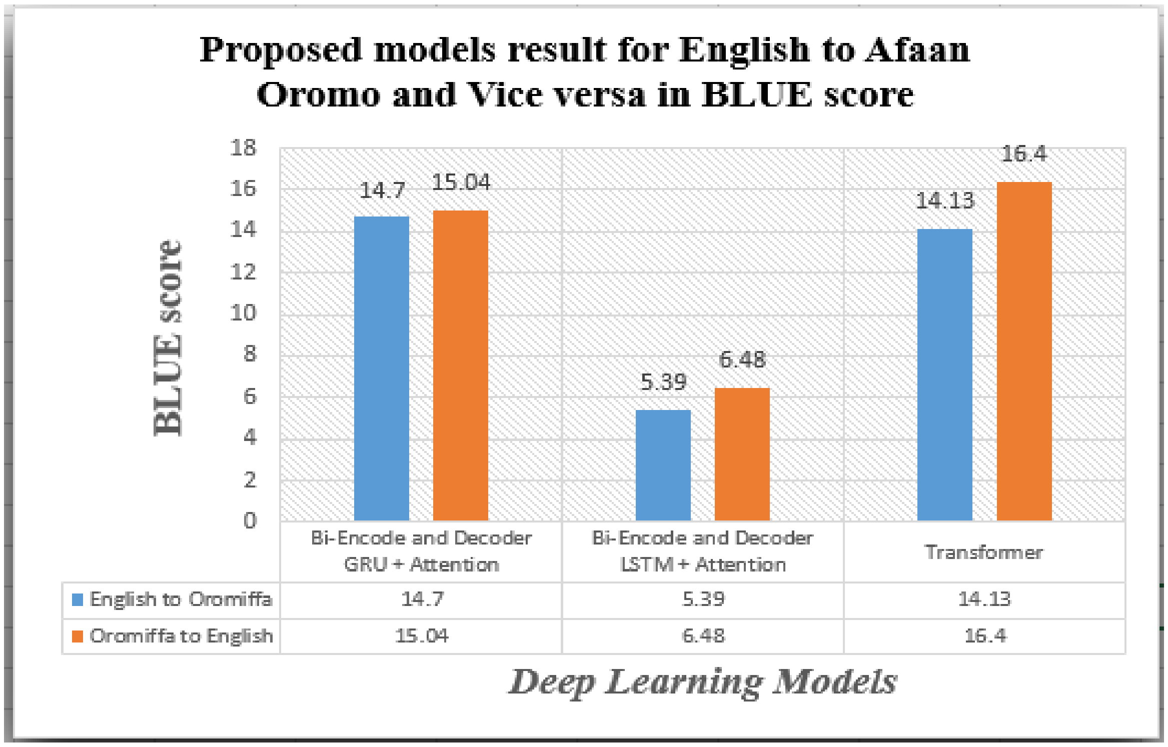
In general, a models is typically trained by maximizing its performance on a particular training dataset. The models thus memorizes the training examples but does not learn to generalize to new situations and dataset. Now that we have a trained models, we can test it on our test dataset. In addition to cross-entropy loss, another good metric we can use is perplexity, which is a measurement of the model's ability to predict the target translation. In each models configuration, test loss and perplexity of the translation scores are different and Table 5:5 shows the test loss and perplexity for each proposed models result from English to Afaan Oromo and vice versa.

Table 5:5 Test loss and perplexity of the proposed models

Models	**English to Afaan Oromo**		**Afaan Oromo to English**	
	Test loss	**Test perplexity**	**Test loss**	**Test perplexity**
Bi-encoder and decoder LSTM with Attention model	5.594	268.844	5.013	150.377
Bi-encoder and decoder GRU with Attention model	4.377	79.598	3.928	50.805
Transformer	**4.201**	**66.776**	**3.339**	**28.188**

The results obtained from proposed models is comparable with each other. The lower the perplexity, the better the model is at predicting the target translation. Here in table 5:5, the performance of transformer looks better than both models. The test perplexity of transformer model is 66.776 for English to Afaan Oromo and 28.188 for Afaan Oromo to English, which is more efficient than Bi-encoder and decoder LSTM with Attention model 268.844 for English to Afaan Oromo and 150.377 for Afaan Oromo to English and also transformer are more efficient than Bi-encoder and decoder GRU with Attention model 79.598 for English to Afaan Oromo and 50.805 for Afaan Oromo to English, where the one with the lower perplexity is better.

Previously we have only considered, the test loss and perplexity of the model. However, there are metrics that are specifically designed for measuring the quality of a translation and the most popular one is BLEU. Without going into too much detail, BLEU looks at the overlap in the predicted and actual target sequences in terms of their n-grams. It gives us a number between 0 and 1 for each sequence, where 1 means there is perfect overlap (a perfect translation), although is usually shown between 0 and 100. We define a *calculate_bleu* function which calculates the BLEU score over a provided TorchText dataset. This function creates a corpus of the actual and predicted translation for each source sentence and then calculates the BLEU score. The experimental results are depicted furthermore Figure 5.6.



	Bi-Encode and Decoder GRU + Attention	Bi-Encode and Decoder LSTM + Attention	Transformer
🟦 English to Oromiffa	14.7	5.39	14.13
🟧 Oromiffa to English	15.04	6.48	16.4

Figure 5:6 Comparison of proposed models in BLUE Score

The most useful part of a BLEU score is that it can be used to compare different models on the same dataset, where the one with the higher BLEU score is better. Thus, we got a BLEU score of ***5.39*** and ***6.48*** from English to Afaan Oromo and vice versa on Bi-encoder and decoder LSTM with Attention and a BLEU score of ***14.7*** and ***15.04*** from English to Afaan Oromo and vice versa on Bi-encoder and decoder GRU with Attention and also, we got a BLEU score of ***14.13*** and ***16.4*** from English to Afaan Oromo and vice versa on Transformer model. Transformer model achieves a better result in BLUE score of ***16.4*** from Afaan Oromo to English. All proposed system does not outperform high BLEU score and also, there is no way to directly compare our result with the papers that are discussed under related work section or other local language like Amharic to English and foreign language English to French, English to Japan etc. however they are using a completely different dataset and also their model hyper-parameter is much different. So, we cannot really compare against that either.

Since the models are trained up until a set number of training steps a comparison is also made on how fast the models train. Since we have used GPU, training time for the DNN for the same datasets on different models architectures as we proposed. Training time for each dataset for a selected number of sentences is shown in table 5:2. A summary of total training time up to 30 epoch can be seen in Table 5:6 and compare its performance with each other models.

Table 5:6 Training time of the proposed models

Model Type	**Total training Time**	
	English to Afaan Oromo	**Afaan Oromo to English**
Bi-encoder and decoder LSTM with Attention model	29 minute	22 minute
Bi-encoder and decoder GRU with Attention model	41 minute	40 minute
Transformer based model	**7 minute**	**6 minute**

In table 5:6, we analyze the efficiency of the proposed method in terms of time. Transformer model is almost faster than the Bi-encoder and decoder GRU with Attention model and Bi-encoder and decoder LSTM with Attention model. We can also observe how the addition of bidirectional encoders and increasing the hidden size is affecting training time. To perform a task to solve real world problems, the machine needs to be equipped with adequate processing power, Like GPU in order to ensure better efficiency and less time consumption.

Finally, once trained we can use the model to produce a set of translations. If we translate some test dataset, we use BLUE score, which is the most common way to evaluate translations. As we created a *translate_sentence* function, we select subsample around fifty (50) sentence from test dataset via to sentence index number based on sentence length. A sample table image that show the Afaan Oromo sentence length with BLUE score result from all proposed modes depicted under *Appendix J.* By using those selected sentences, we translate from our trained models with the *translate_sentence* function. Then, we grab some translations from our test dataset and see how well our model did and the effects on long sentences. When to compare the effects on long sentence of all proposed models, we use the same sentences from our test dataset based on it index. We are going to pick examples here so it gives us something interesting to look at, Figure 5:7 show the effects on long sentences translation quality based on the performance of all proposed models.

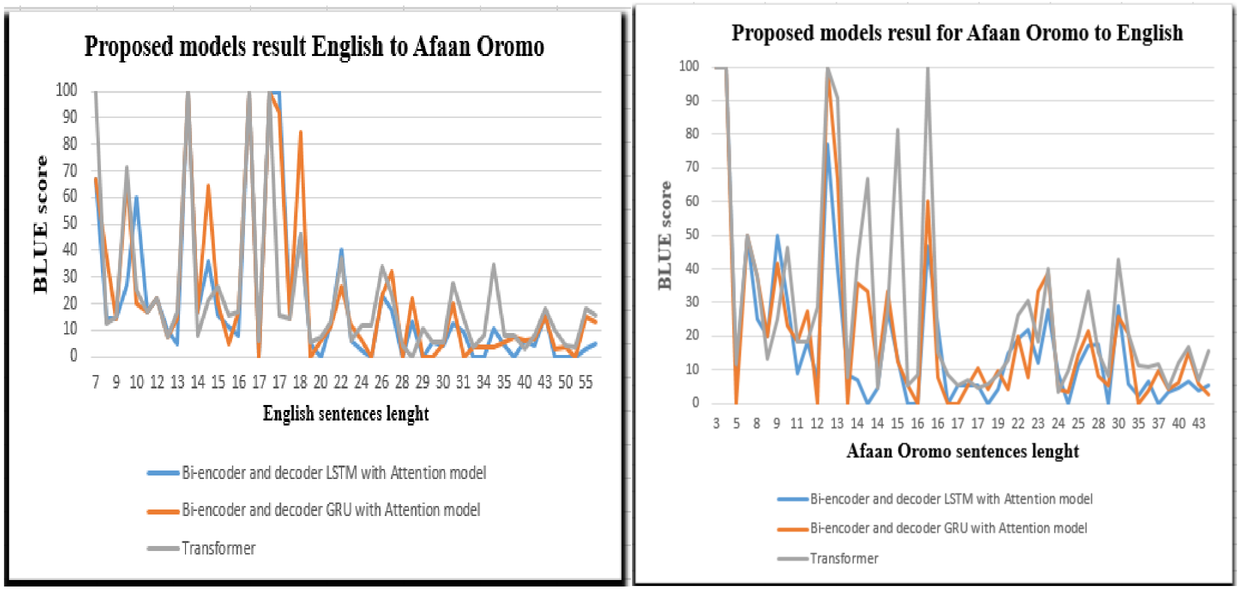
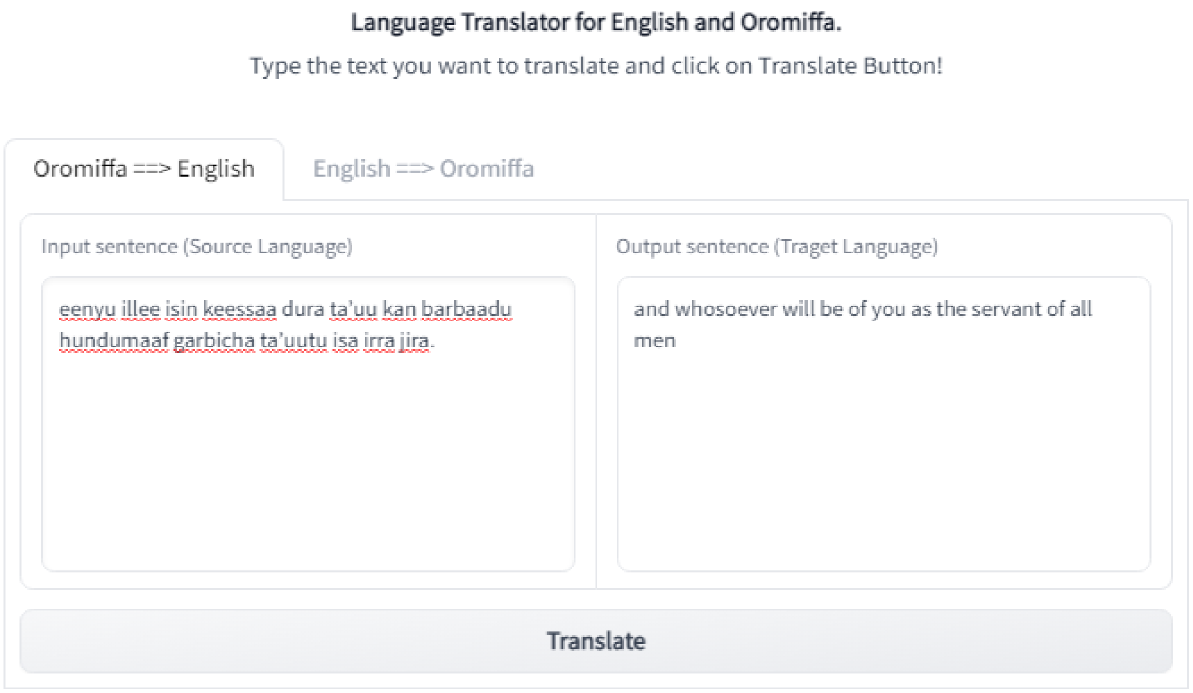


Figure 5:7 increasing the length of sentence and BLUE score results

The result is up and down in terms of the BLUE score. In all case, the Transformer model outperforms rather than both proposed models in both directions. As the length of sentence increases, the performance of the models degrades. But, as we seen on figure 5:7, Bi-encoder and decoder GRU with Attention model is compute the transformr model.

5.6 Model Inference and User interface

Inference is the process of running dataset points into a ML model to calculate an output. This process is also referred to as putting a ML model into production [93]. Our proposed system is not implemented to use, but in-order to evaluate and compare our proposed models effectiveness and efficiency regarding too long sentence, we must write the code for translations and we can use our trained model to generate translations. But for demonstration and manually evaluation purpose we develop prototype by using best BLUE score result, transformer is performed best scored as we compared previously. So, we developed a prototype for transformer model. The user interface of the prototype is developed by using Gradio. Gradio is an open source library in python for creating user interfaces. The prototype UI accepts Afaan Oromo or English from the user, sends to model and model generate a text based on it learn and displays the translated text to the user as the following figure 5:8.


Language Translator for English and Oromiffa.

Type the text you want to translate and click on Translate Button!

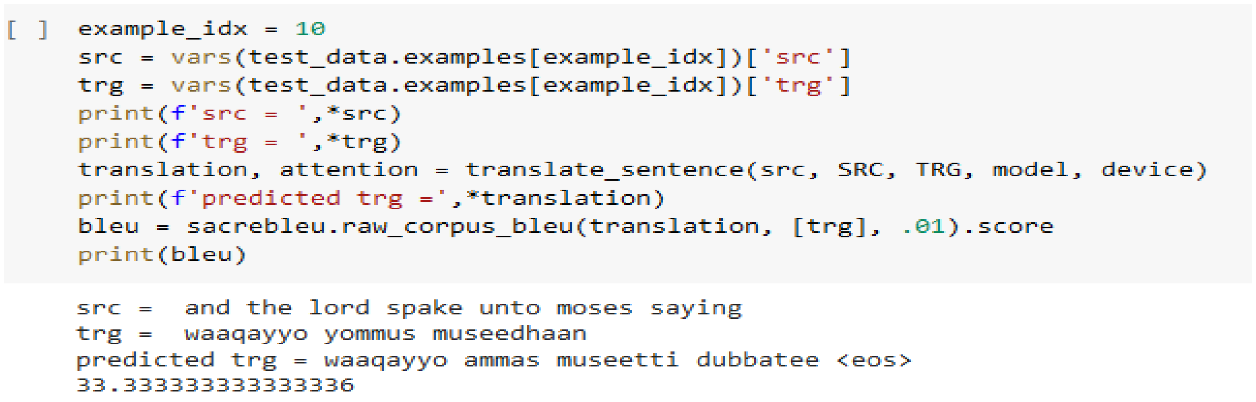
Oromiffa ==> English
English ==> Oromiffa
Input sentence (Source Language)
Output sentence (Traget Language)
Translate

eenyu illee isin keessaa dura ta'uu kan barbaadu hundumaaf garbicha ta'uutu isa irra jira.

and whosoever will be of you as the servant of all men

Figure 5:8 Screenshots of Afaan Oromo to English and vice versa machine translator UI

Next, we have a function that displays how much the model pays attention to each input token during each step of the translation. We show sample result from proposed models, by using *translate_sentence* function to get predicted translation and attention. Show this graphically by having the source sentence on the x-axis and the predicted translation on the y-axis. The lighter the square at the intersection between two words, the more attention the models gave to that source word when translating that target word. First, we try to get an example from the testing dataset via to sentence index numbers. Then we pass it into *translate_sentence* function which gives us the predicted translation tokens as well as the attention as shown below figure 5:9.



```
[ ]  example_idx = 10
     src = vars(test_data.examples[example_idx])['src']
     trg = vars(test_data.examples[example_idx])['trg']
     print(f'src = ',*src)
     print(f'trg = ',*trg)
     translation, attention = translate_sentence(src, SRC, TRG, model, device)
     print(f'predicted trg =',*translation)
     bleu = sacrebleu.raw_corpus_bleu(translation, [trg], .01).score
     print(bleu)
```

```
     src =  and the lord spake unto moses saying
     trg =  waaqayyo yommus museedhaan
     predicted trg = waaqayyo ammas museetti dubbatee <eos>
     33.333333333333336
```

Figure 5:9 Snapshot from Bi-encoder and decoder GRU with Attention model result

We can see BLUE score result and that GRU model mistranslates *“yommus museedhan”* as*”ammas museetti dubbate”*. We can view the attention of the GRU model, making sure it gives sensible looking results.

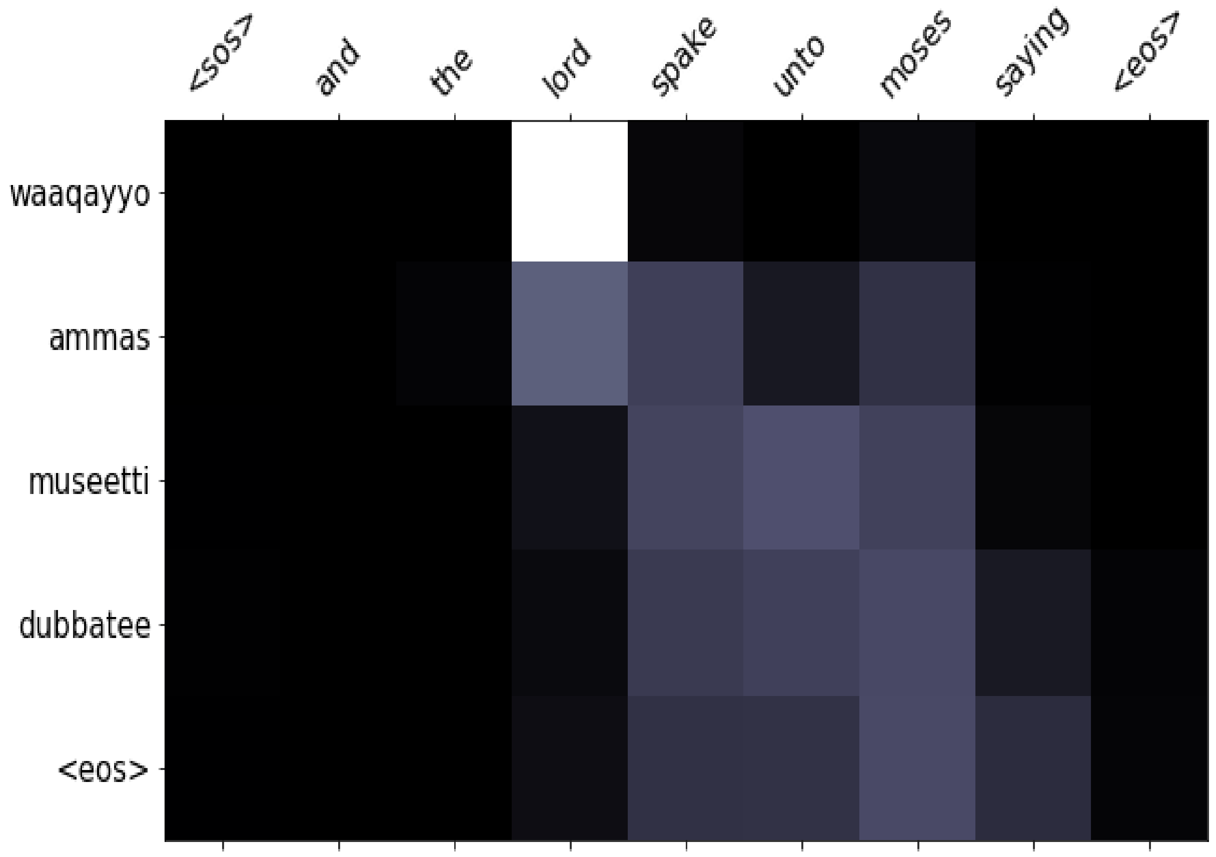
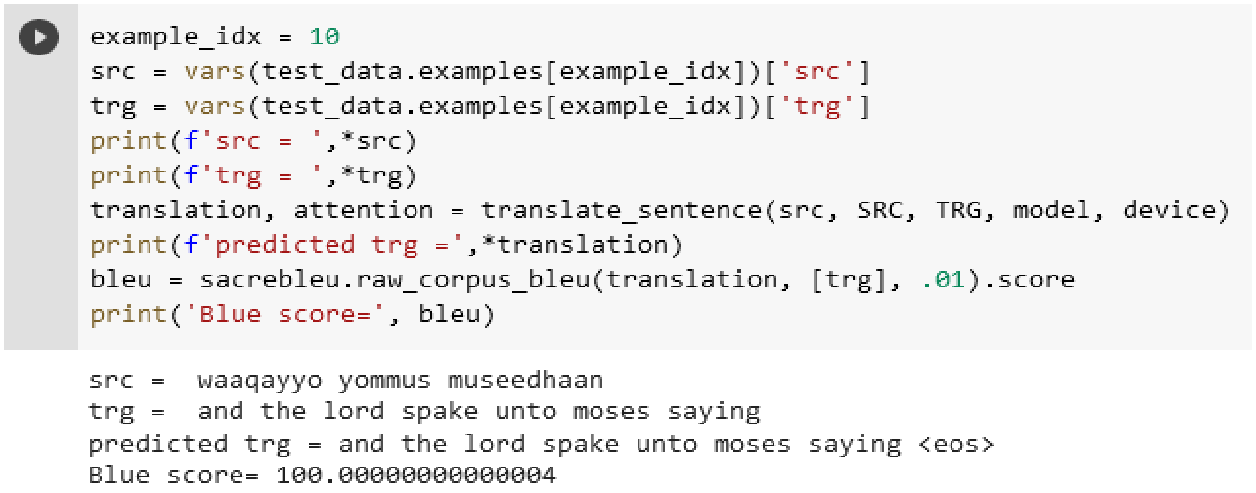


Figure 5:10 Show Attention alignment from Bi-encoder and decoder GRU with Attention model.

We can see it pays attention to *“lord”* when generating the token *“waaqayyon”*, have strong relevance than other words. So, the lighter the square at the intersection between two words, they have more attention the model gave to that source word when translating that target word.

Finally, let's check the transformer models with the above sentence. Then start translating sentences as the following Figure 5:11 and figure 5:12.



```
example_idx = 10
src = vars(test_data.examples[example_idx])['src']
trg = vars(test_data.examples[example_idx])['trg']
print(f'src = ',*src)
print(f'trg = ',*trg)
translation, attention = translate_sentence(src, SRC, TRG, model, device)
print(f'predicted trg =',*translation)
bleu = sacrebleu.raw_corpus_bleu(translation, [trg], .01).score
print('Blue score=', bleu)
```

```
src =  waaqayyo yommus museedhaan
trg =  and the lord spake unto moses saying
predicted trg = and the lord spake unto moses saying <eos>
Blue score= 100.00000000000004
```

Figure 5:11 Snapshot from transformer model result

We get a correct translation result here by using the transformer model.

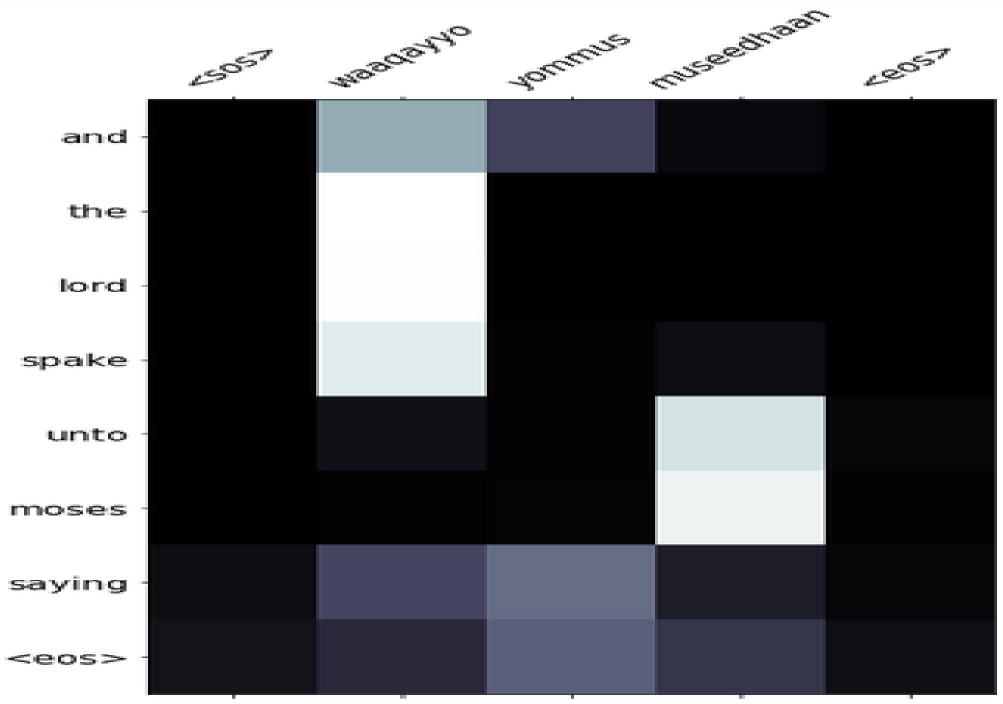


Figure 5:12 Show Attention alignment from Transformer model result

We can see it pays attention to *"the"* and *"lord"* when generating the token *"waaqayyo"* and also "*unto"* and *"moses"* when generating the tokens "*museedhaan"* have strong relevance than other words. The Transformer does meet the correct translation result, the self-alignment in the target can capture the target side relationship among the target words. The above example demonstrates that the proposed algorithms can learn target-side relationship to help translate.

5.7 Linguist Evaluation

As we noted earlier in the experiment, we used a standard BLEU score and Perplexity evaluation metric in order to evaluate proposed models. At the end, in order to ensure whether the proposed translation model is acceptable (*usability* and *quality*) or not we used the linguist manual evaluation method. Here are some factors to consider during linguistic evaluation, in order to evaluate UI usability and machine translated sentences quality.

As we mentioned in section 4.1.1, most of our dataset is collected from religious books, so we have to choose people who know about the bible in order to evaluate the UI usability and machine translated sentences quality. Because the norm of grammar in bible is different from the regular Afaan Oromo text grammar. So, we must select appropriate evaluators according to the dataset domain. Purposively, we selected six religious people (who serve in the church) and those who could speak, read and write the Afaan Oromo and English language. Then we used zoom meeting platform to discuss about the evaluation process. The reason we used zoom is because the people we chose live in different places (two in Dire Dawa city, two in Bokoji town, one in Bale Robe town and one in Hawassa town). So we had to use Zoom Rooms in order to give orientation for all evaluators. Then we send prepared questioners and the link of our machine translation UI via to telegram for evaluators in order to fill the evaluation form and to access the UI. They filled out the form according to we explained to them, afterward they send back they assessment to us via telegram. First we evaluated the usability of UI, then after we evaluated machine translated sentences quality proceed as follows.

A. Usability testing

Usability testing allows to understand the reception of the design and MT services from the user's standpoint and also, it is a key factor for increasing adoption of machine translation. Usually, during usability testing users are asked to complete tasks to see where they encounter problems and experience confusion. While there is some divergence around the attributes of usability, the majority of terms in the literature closely adhere to the International Organization for Standardization (ISO) attributes. According to Jones [94], Using the 11 attributes of usability, one can determine how to present digital content that will best satisfy users: *Learnability*, *Efficiency, Effectiveness, Memorability and retain-ability, Low error rate and error tolerance, Attitude and satisfaction, Usefulness, Control and flexibility*, *User characteristics*, *Context and purpose* and *Interface and design*. It has been validated and used in various usability studies since then [95]. Among those attributes [94], we selected six attributes with the benefits of MT in mind. Most of the time, Usability is understood here as the extent to which our proposed MT can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use. The author makes slight adjustments to it according to the actual situation of this thesis. In keeping with the Jones [94] definition given above, the measures of usability in this study are involves the following scales:

1. ***Appearance:*** Measured as the UI should be designed simple, without much visual stimuli, in order to maximize the chance of concentration, comprehension and learning of the user.
2. ***Feedback and error message:*** measured as the UI should be designed in a way that, it should alert the user when he/she made a mistake, it should display error message
3. ***Ease of use:*** measured as the UI provide information easily and quickly. It provide a single clickable button and have clear (color) iconic button.
4. ***Effectiveness:*** measured as the number of successful translated output.
5. ***Efficiency:*** measured as the overall duration of the translation completed time.
6. ***Satisfaction:*** a measurement of user satisfaction of the instructions on a post-task 5-point Likert scale.

So based on those attribute we measured proposed model quality and prototype usability. The selected users are six (6) in number. In order to evaluate UI usability, we have informed evaluators that they should use 5-10 minutes before evaluated. For each of the questionnaire's scales, the participants select one of three responses (***High = 2, Medium = 1 and Low =0***) to evaluate their attitudes towards the UI usability. The participants were asked nine questions required to encompass the categories of those selected usability attributes, all the response from the selected users were used as an input to analyze and discuss the usability of our UI.

All Result generated from nominal data were collected from structured questionnaire's in relation to usability attributes are presented in table 5:7. Results are extracted from nominal data through the statistical methods of measurement in particular for the purpose of converting nominal data in to numerical data from the total structured questionnaires under in each usability attributes from all evaluators. The result of numerical data conversions has been illustrated in table 5:7. The snapshot of a sample usability evaluation is shown under *Appendix K.* Based on the equation 5.1 below we calculated Average value for each Usability attributes.

$$\boldsymbol{AUAR} = \left(\left(\frac{\sum_{i=0}^{n}(\boldsymbol{VoR*NESV})}{\boldsymbol{NoEvaluators}} \right) \mathbf{*100} \right) / \boldsymbol{High = 2} \qquad \text{Equation 5:1}$$

AUAR = Average of each Usability attributes result
VoR = Value of the Responses, it means High = 2, Medium = 1 and Low =0
NESV = number of evaluator those selected that value
NoEvaluator = Total number of evaluator, we selected 6 evaluator.
After we get the summation value, then multiply by the percentage (100) and finally divided by the highest of the selection points (High = 2).
Finally in order to get usability result for developed prototype, we use the following equation 5:2

$$\textbf{Prototype usability Result} = \frac{\sum_{i=0}^{n}(\boldsymbol{AUAR})}{\boldsymbol{NoEvaluator}} \rightarrow \textit{usability result in percentage} \qquad \textit{Equation 5:2}$$

Table 5:7 Prototype Usability evaluation result

Attributes	**Questionnaires under in each usability attributes**	**Converting nominal data in to numerical data from questionnaires**				
		Low (0)	*Medium (1)*	*High (2)*	Average (3 pt)	Average each attribute
Appearance	*Do you like the color, font and button size of the interface?*		2	4	83.3333	88.8888 %
	Does the organization of information on the interface screen clearly presented?			6	100	
	Does the user interface provide clear visual distinction between categories?		2	4	83.3333	
Feedback and error messages	*Does the user interface include error messages in case you insert incorrect input or you click a button with not input?*		3	3	75	75 %
Ease of use	*Is the user interface easy to use for translation?*			6	100	100%
Effectiveness	*Does the user interface produce the appropriate output as you desired?*	2	3	1	41.6666	41.666 %
Efficiency	*Does the user interface take too much time to complete specific translation?*		3	3	75	75%
Satisfaction	*Are you satisfied with the user interface?*		1	5	91.666	87.5 %
	Do you feel comfortable using the user interface?		2	4	83.3333	
Prototype usability evaluation result in Average						78.0 1%

For example, in Table 5.7, the average of the "Appearance" could be calculated as ((0*0) + (1*2) + (2*4))/6 = 10/6= 1.666 then (1.666*100)/2= 83.333. Similarly we can calculate the other Questions result. In order to get the Appearance attribute result only, (83.33 + 100 +83.33)/3 = 88.8888%. The evaluation UI was tested, and the results revealed that Appearance, Feedback and error message, Ease of use, Effectiveness, Efficiency and Satisfaction are interrelate. Based on the result of the assessment from questioners, the following graph are showed.

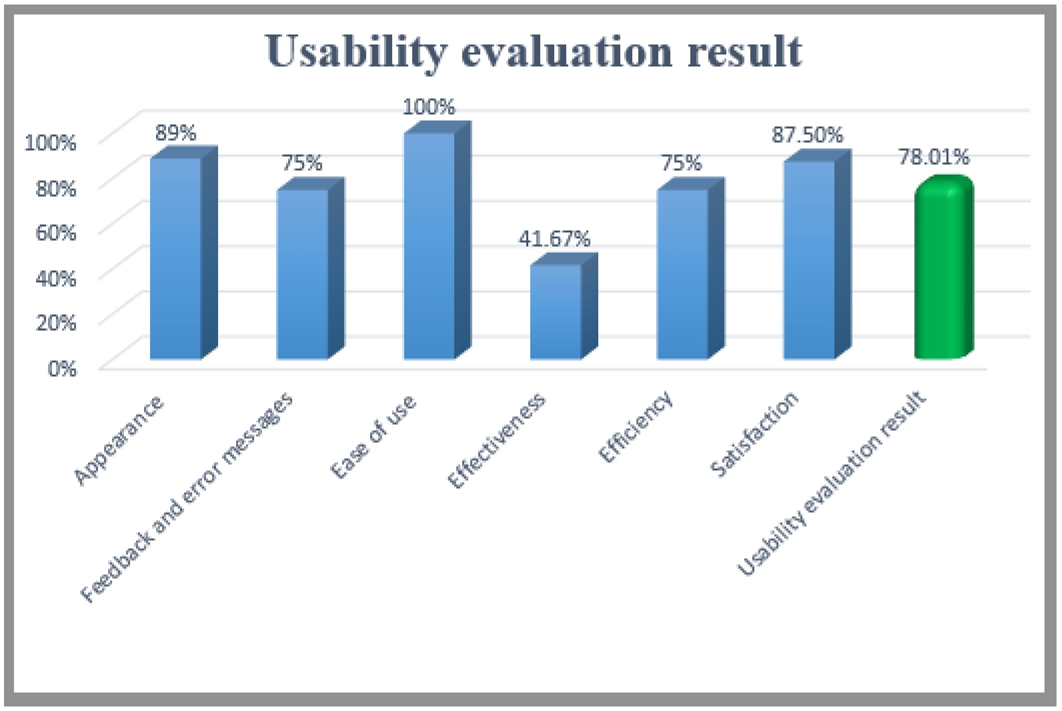


Figure 5:13 Comparison of usability attribute and Average result of usability evaluation

As we have noticed before, the results we got for BLUEU score are low, yet as you can see from the figure 5:13 above, from other usability attribute the result of effectiveness (41.67%) is low. According to some evaluators we have been told that its effectiveness outside the domain is low. As suggestion from some evaluators, the spelling of some words is incorrect when it translates for us. And as we understand, this suggestion is correct. Under the challenges we face, we will try to explain the reasons behind this problem. As we can see from the table 5:7 and figure 5:13 above, developed prototype was accepted by the evaluators with 78.01% of the results. Overall they gave us very nice and encouraging comments.

B. Quality Evaluation

Human evaluation has been done based on fluency and adequacy in some works in machine translation. This means, native speakers of the source and target languages judge the quality of translation by assessing its overall fluency and adequacy. The main disadvantage of manual evaluation is that it is time-consuming and thus too expensive to do frequently.

However, we have had people evaluate the results of our model. To conduct this assessment initially, 60 sentence pairs were prepared but reduce to 35 based on the evaluator feedback. Because they say it's very boring. As mentioned above, we selected six evaluators.

As you can see sample under the *Appendix L,* we have prepared 35 translated pair sentences Afaan Oromo to English as well 35 translated sentences English to Afaan Oromo. Totally 70 pairs of translated sentence have prepared for manual evaluation purpose and their corresponding Value of Translation Quality (VTQ) evaluation are listed below.

1. Bad Translation or Nothing Translated = 0 (abbreviated as *NT*)
2. Most not Translated =1 (abbreviated as *MnT*)
3. Partially not Translated = 2 (abbreviated as *PnT*)
4. Almost Translated = 3 (abbreviated as *AT*)
5. Correct Translated = 4 (abbreviated as *CT*)

Based on the above response values, evaluators checks the content to make sure it’s technically accurate. As a samples, the below table 5:8 shows translated pair sentence with their evaluation result. Based on the equation 5.3 below we calculated Average value for each Usability attributes.

$$ATSR = \left(\left(\frac{\sum_{i=0}^{n}(\boldsymbol{VTQ} * NESV)}{\boldsymbol{TotalNoOfEvaluator}} \right) *\mathbf{100} \right) / \text{CT} = 4 \qquad \textit{Equation 5:3}$$

ATSR = Average of each translated sentence result
VTQ = Value of Translation Quality, it means NT= 0, MnT = 1 or other based on above listed.
NESV = number of evaluator those selected that value.
NoEvaluator = Total number of evaluator, we selected 6 evaluator.
After we get the summation value, then multiply by the percentage (100) and finally divided by the highest value of the responses (Correct Translated = 4).
Finally in order to get manual evaluation average result for translated sentences, we use the following equation 5:4

$$\textbf{Translation Quality} = \frac{\sum_{i=0}^{n}(\boldsymbol{ATSR})}{\boldsymbol{TotalNoOfEvaluator}} \rightarrow \textit{Quality result in percentage} \qquad \textit{Equation 5:4}$$

Table 5:8 Sample translated sentences from Afaan Oromo to English with their response value

No	index	Translated Sentences from Afaan Oromo to English via to transformer model	Translation quality assessment					
			Value of Translation Quality (VTQ)					Aver-age
			NT (0)	MnT (1)	PnT (2)	AT (3)	CT (4)	
1	1004	**Oromiffa =** warri bet'el shaarezerii fi regiim-melekin namoota isaanii wajjin waaqayyoon araara akka kadhataniif **Predicted English =** when they had sent unto the house of god and to the men of god which have <unk>	1		3	2		50 %
2	10	**Oromiffa =** waaqayyo ammas museedhaan **Predicted English** = and the lord spake unto moses saying				1	5	95.833 %
3	1059	**Oromiffa =** pheexiros deebisee fakkeenyicha nuuf ibsi jedheen. **Predicted English =** then answered peter and said unto him declare unto us this good things?		1	1	1	3	75 %
4	1203	**Oromiffa =** kana irratti mootichi aaree dheekkamaa gara mana mootummaa isaa isa mandara samaariyaa keessaa in dhaqe. **Predicted English =** and the king of israel went to his house and came to samaria.		1	2	3		58.333 %
5	1207	**Oromiffa =** wanti hundinuu karaa isaa uumame isaafis uumame. **Predicted English =** all things were created by him and for him				1	5	95.833 %
6	1209	**Oromiffa =** isaan saba sodaachisoo fi naasisoo dha firdii ofuma isaaniitiif in baafatu ulfinas ofii isaaniitiif in kennu. **Predicted English =** they are terrible and dreadful their judgment and their dignity shall proceed of themselves.				3	3	87.5 %
7	1210	**Oromiffa =** abdii qabdanitti gammadaa rakkina obsaa waaqayyoon kadhachuuttis cimaa **Predicted English** = rejoice in hope of hope for hope for we have received of god			1	4	1	75 %
Translated sentences quality result for above sample sentences in Average								**64.2%**

For example, in Table 5.8, the average of translation quality for the first sentence could be calculated as ((0*0) + (1*0) + (2*3) + (3*2) + (4*0))/6 = 12/6= 2 then (2*100)/4= 50. Similarly we can calculate the other Questions result. Finally, we use equation 5:4 to find the average quality result of the translated sentences, in other word we can use the average of all sentences. When we come to the overall manual evaluation quality result, from Afaan Oromo to English 73.69 % and from English to Afaan Oromo 52.38 % we have get. As we seen the BLEU score result from Afaan Oromo to English is higher than English to Afaan Oromo, the same to that, the result from Afaan Oromo to English is better than English to Afaan Oromo. This indicate translation from morphological simple language to morphological rich language is more challenged for machine translation. Morphological distinctions not present in the source language need to be generated in the target language.

Given the closeness of most systems and the wide over-lapping confidence intervals it is hard to make strong statements about the correlation between human judgements and automatic scoring methods such as BLEU. But as we understand it, manual evaluation method are very tiring and exhausting, however, the words that written in differently spelling but they have the same meaning manual evaluation method is better. But manual evaluation, depend on the levels of the evaluator knowledge. So it is difficult to get the same result on the same translated sentence, by using different evaluators.

5.8 Discussion on the Result of the Study

Experiments are conducted to measure the effectiveness of DNN approach by using different algorithms on MT task. We proposed three DL algorithms as discussed earlier. However, in our domain, Bi-encoder and decoder LSTM with Attention and Bi-encoder and decoder GRU with Attention proposed models takes full account of the forward and reverse target information, and combines them efficiently and apply attention mechanisms to help to generate the target sequence. It should be able to better understand the context of each word by looking at words both before and after each word. This thesis work aims to address this discrepancy DL algorithms by discussing the development of bidirectional MT system for Afaan Oromo and English language, in considering the following two main questions that we have formulated as research questions. In discussing, this research question to be answered based on above the experiment, hence, here is their answer with the findings of the study.

Q1: How does the Transformer algorithms perform on the task of MT in comparison with a GRU and LSTM algorithms using bidirectional encoder and decoder with attention mechanism for low resource language such as Afaan Oromo?

All proposed models are good machine learning models for MT task. But when we compare with each other, the Transformer model translated result is better, especially translated from Afaan Oromo to English, and it provided more promising MT performance compared to the both proposed models. The BLUE score for Afaan Oromo to English increased from 6.48, while using the Bi-encoder and decoder LSTM with Attention model, to 15.04 while using the Bi-encoder and decoder GRU with Attention model, to 16.40 while using the Transformer model. As well as, the BLEU score for English to Afaan Oromo increased from 5.39, while using the Bi-encoder and decoder LSTM with Attention model, to 14 while using the Bi-encoder and decoder GRU with Attention model, to 14.13 while using the Transformer model And also The test perplexity of transformer model is 66.776 for English to Afaan Oromo and 28.188 for Afaan Oromo to English, which is more efficient than Bi-encoder and decoder LSTM with Attention model 268.844 for English to Afaan Oromo and 150.377 for Afaan Oromo to English and also transformer are more efficient than Bi-encoder and decoder GRU with Attention model 79.598 for English to Afaan Oromo and 50.805 for Afaan Oromo to English, where the one with the lower perplexity is better.

Transformer model is mainly to reduce the number of trainable parameters, memory demand, as well as training time. The comparison of all proposed model trainable parameters is demonstrated in table 5:4 and also the comparison of all proposed model training time is demonstrated in table 5:6. The Transformer model trains faster. Bi-encoder and decoder LSTM with Attention model and Bi-encoder and decoder GRU with Attention model required more parameters than Transformer. Bi-encoder and decoder GRU with Attention model are still the best compared to Transformers choice when the sequence length is too long as we seen on figure 5:7. Transformer model, due to its design can allow both data and model parallel training, the transformer is much more efficient than both proposed models. In simple words, we have observed in our experiments, that deep learning models are effective and efficient for low-resourced languages such as Afaan Oromo, but sensitive to hyper-parameters and quality of corpus. When we come to our proposed models, Transformer model is better for such low-resourced languages.

Q2: What are the possible main challenges in the Afaan Oromo and English machine translation system? This is answered below in section 5.9.

5.9 Main Challenges in the Afaan Oromo and English MT

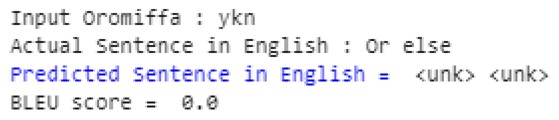
MT has its own challenges and still an active research area [16], [47]. As the number of parallel corpus increase the machine is more trained. But for luck resource language like as Afaan Oromo language, it is difficult to build more data and it is difficult to cover all domain. When to conduct the above experiment we faced different challenges. In this section we describes some challenge. Let us see one by one

5.9.1 Corpus Quality

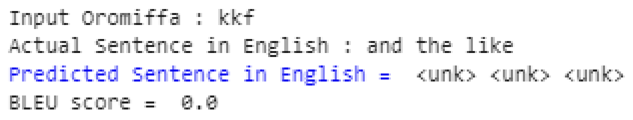
Deep learning models are sensitive to hyper-parameters and the training data they have fed. However, as discussed in section 4.1.1 the corpus collected from the previous researcher (Yitayew Solomon [5]) lacks quality. Some of these texts are not complete sentences and also duplicate sentence. Though we have tried to fix this issue manually with linguists' line by line, it was inefficient and time-consuming, we have been unable to do the whole corpus. The author described that alignment quality is a common problem, if sentences are miss aligned the performance of the translation processes becomes poor. Moreover, the poor quality of corpus cause the following challenges:

A. Identification of Abbreviations

In many cases abbreviations are written as a sequence of characters terminated with a period (.). In MT, there is no such rules for identification of abbreviation which makes very hard to classify the word either it is abbreviation. An abbreviation can also represent several different words, as is the case for *St.* which can stand for *Saint, Street, or State*. However, as *Saint* it is less likely to occur at a sentence boundary than as *Street*, or *State*. In our text some sentence wrote in word and some text wrote in abbreviation, this led to rare word or OVW. It is one challenge for MT during corpus preparation and preprocessing. To address these issues, MT systems needs prepared corpus into standard and appropriate word form for successful word generate. So we tried to write in the same form, however, when we give an abbreviation, the translation it gives us is low. We can see the following example: Or else => Yookiin (ykn), and the like => kan kana fakkaatan (kkf)



```
Input Oromiffa : ykn
Actual Sentence in English : Or else
Predicted Sentence in English =  <unk> <unk>
BLEU score =  0.0
```



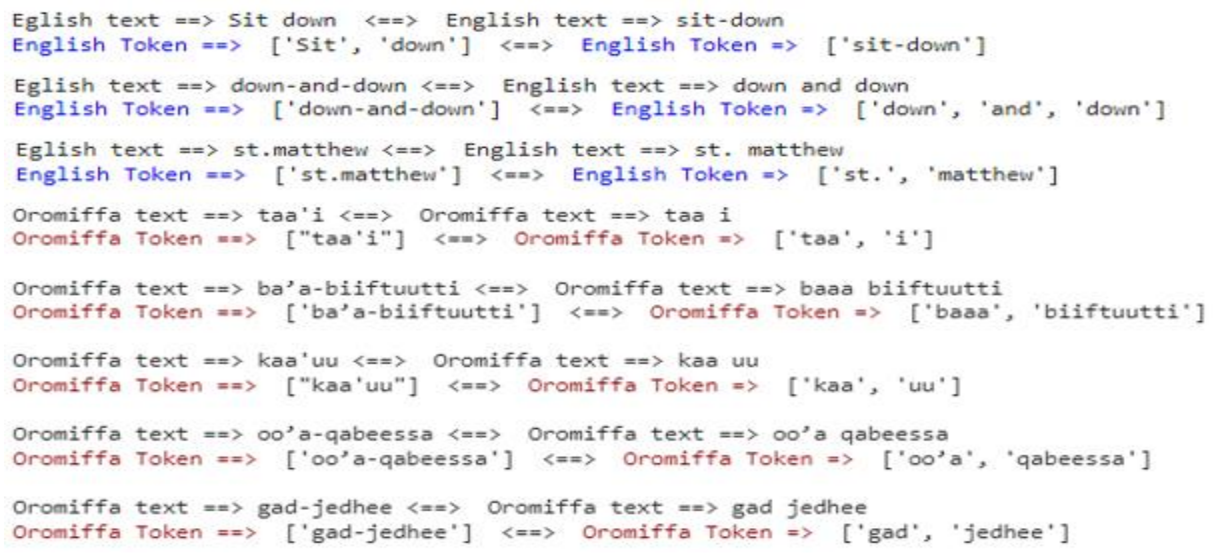
```
Input Oromiffa : kkf
Actual Sentence in English : and the like
Predicted Sentence in English =  <unk> <unk> <unk>
BLEU score =  0.0
```

Figure 5:14 Sample snapshot to show translated abbreviations result in BLEU score

B. Handling of punctuation and Word segmentation

Clear word structure depends on the writing style of the language. Actually, identification of the meaningful words is the most important prerequisite of our task in order to increase parallel corpus quality. Because DL models are sensitive to the training dataset they have fed. In some cases, it is difficult to identify compound words such as *sit down* and *sit-down* for tokenization. Compound words are unique because they have their own meaning. The word *sit down* without hyphen (-) is taken as ***two separate*** word tokens whereas the second word (*sit-down*) is taken as a single word. This will lead the model to create a large size of Word Vocabulary (rare or OVW) and learn less. This makes challenging in this study, since our bilingual corpus have inconsistent writing (punctuation marks between some words in some cases and no in other similar words).

Removing unused characters from input sentence is commonly used MT in order to increase corpus quality. This task mainly helps us to clean our dataset from different unused special characters like comma, single and double quotation. Other than the apostrophe ('), all punctuation marks used in both Afaan Oromo and English languages are the same and used for the same purpose. Apostrophe (') in English shows possession, but in Afaan Oromo it is used in writing to represent a glitch sound known as '*hudhaa*'. It plays an important role in Afaan Oromo reading and writing system. It is used to write a word in which most of the time two vowels appear together. Hence, during preprocessing sentence we must handle over score (-) in-order to tokenize compound words. Though it has two different meanings, it forms a meaningful word used to write words such as written in the figure 5:15.



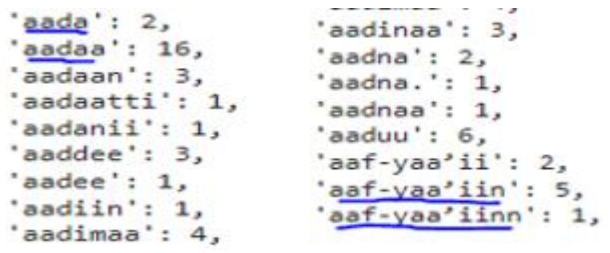
```
Eglish text ==> Sit down  <==>  English text ==> sit-down
English Token ==>  ['Sit', 'down']  <==>  English Token =>  ['sit-down']

Eglish text ==> down-and-down <==>  English text ==> down and down
English Token ==>  ['down-and-down']  <==>  English Token =>  ['down', 'and', 'down']

Eglish text ==> st.matthew <==>  English text ==> st. matthew
English Token ==>  ['st.matthew']  <==>  English Token =>  ['st.', 'matthew']

Oromiffa text ==> taa'i <==>  Oromiffa text ==> taa i
Oromiffa Token ==>  ["taa'i"]  <==>  Oromiffa Token =>  ['taa', 'i']

Oromiffa text ==> ba’a-biiftuutti <==>  Oromiffa text ==> baaa biiftuutti
Oromiffa Token ==>  ['ba’a-biiftuutti']  <==>  Oromiffa Token =>  ['baaa', 'biiftuutti']

Oromiffa text ==> kaa'uu <==>  Oromiffa text ==> kaa uu
Oromiffa Token ==>  ["kaa'uu"]  <==>  Oromiffa Token =>  ['kaa', 'uu']

Oromiffa text ==> oo’a-qabeessa <==>  Oromiffa text ==> oo’a qabeessa
Oromiffa Token ==>  ['oo’a-qabeessa']  <==>  Oromiffa Token =>  ['oo’a', 'qabeessa']

Oromiffa text ==> gad-jedhee <==>  Oromiffa text ==> gad jedhee
Oromiffa Token ==>  ['gad-jedhee']  <==>  Oromiffa Token =>  ['gad', 'jedhee']
```

Figure 5:15 Sample snapshot for show word segmentation

There is a high risk of missing words in the training dataset. So, if the model has seen *‘dhagna/’* and *‘qabaa/’* in the training dataset but the final text has *‘dhagna-qabaa’*, the model won't be able to recognize the word and it treated with an <*unk*> token. Similarly, punctuation poses another problem. For example, *Bu'aa or buaa*, let or let's, *ba’a-biiftuutti or baa-biiftuutti* will need individual tokens which is an inefficient solution. With tokens, the model won't recognize the variants of words that were not part of the dataset on which the model was trained.

C. Handling Misspelled Words

Handling of misspelled words is one challenging task for Afaan Oromo and English sentence tokenization during preprocessing. These misspelled words lead to rare words or out-of-vocabulary (OOV) and also to assign each word unique word ID for those misspelled words. Reasons of OOV or rare words includes word missing in training data, misspelled words, technical terms and foreign words that usually cannot be translated.

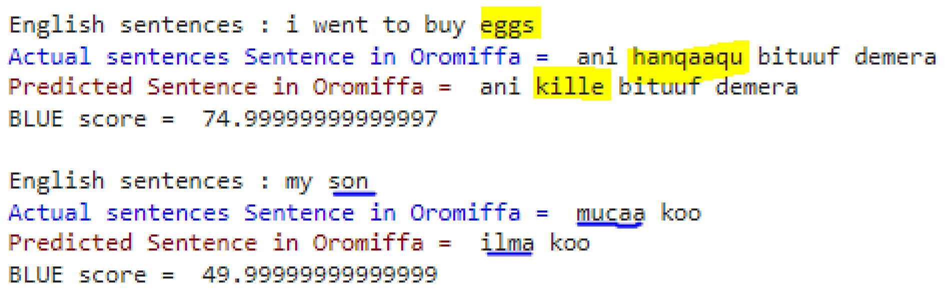


```
'aada': 2,          'aadinaa': 3,
'aadaa': 16,        'aadna': 2,
'aadaan': 3,        'aadna.': 1,
'aadaatti': 1,      'aadnaa': 1,
'aadanii': 1,       'aaduu': 6,
'aaddee': 3,        'aaf-yaa’ii': 2,
'aadee': 1,         'aaf-yaa’iin': 5,
'aadiin': 1,        'aaf-yaa’iinn': 1,
'aadimaa': 4,
```

Figure 5:16 Snapshot for sample misspelled words in list of words

5.9.2 Translation divergences

Afaan Oromo is a morphologically rich language since for morphologically rich languages it is not possible to cover all the words that exist in the language for translation. Usually a human translator translate the meaning of the text in their own context. In Afaan Oromo language, some words have different meanings in different context and different words that have the same meaning. To select appropriate translation according to situation is one challenge in MT and also real challenge for BULE matrix for measure the result of different words that have the same meaning. Because different words have 0 value, but translation is correct. And Word by word translation cannot be good option in proverb and idiomatic sentences. The following figure 5:17 show the translation divergence in BLEU score result.



```
English sentences : i went to buy eggs
Actual sentences Sentence in Oromiffa =  ani hanqaaqu bituuf demera
Predicted Sentence in Oromiffa =  ani kille bituuf demera
BLUE score =  74.99999999999997

English sentences : my son
Actual sentences Sentence in Oromiffa =  mucaa koo
Predicted Sentence in Oromiffa =  ilma koo
BLUE score =  49.99999999999999
```

Figure 5:17 Sample snapshot to show the translation divergence in BLEU score

The translation is 100 percent accurate or correct, but the BLEU score is low. Those highlighted and underlined words have the same meaning (egg => hanqaaquu or kille, son =.mucaa or ilma). This indicates that if there are many words like this, they can degrade the BLUE score result, and also it is a challenge for automatic evaluation like BLUE. In order to overcome those problem manual evaluation is best.

Despite these occurrences of word variations and some differences in pronunciation, it needs to be underlined that all Oromo varieties are mutually intelligible, and all Oromo people understand one another without great difficulty. But machine cannot easily understand those word as human. A large corpus might be required to handle such translations problem. The dialectal variations in the Oromo language are relatively negligible given the fact that it is spoken over a large geographical area. As it was observed a long, Even though speakers of all branches of Oromo understand one another without much difficulty and the Oromo varieties have the same phonemic inventory of consonants and vowels, and have a common denominator in many phonological and morphological rules, there are some dialect differences which cannot be underestimated especially with regard to the development of Afaan Oromo MT and writing system. It is also a real challenge for automatic evaluation mothed like BLEU. This is because a MT and writing system, unlike the spoken form of a language, needs to be standardized in order to foster, among other things, political and cultural unity and machines to use. A standard written form also facilitates communication and is beneficial in education as well as in MT.

CHAPTER SIX

Conclusion and Recommendation

This chapter discusses the conclusion driven from this thesis work, contribution of the study and suggests some feature research directions that utilize pre-existing and researcher's MT systems or related task.

6.1. Conclusion

The main goal of our study is to design and develop a bi-directional NMT system between English and Afaan Oromo language pair by using a deep learning approach. Neural machine translation is a novel paradigm in machine translation research. To perform this research experiments, the datasets were collects from Holy Bible (*17105 corpus prepared by researcher*) and we use the previous collected parallel corpus, prepared by Yitayew Solomon [5] (4361 corpus prepared by Yitayew). After cleaned and removed redundant sentence from the whole corpus we used 21323 corps for this experiment. In this thesis, Bi-encoder and decoder LSTM with Attention, Bi-encoder and decoder GRU with Attention and Transformer algorithms were performed on the task of MT between English and Afaan Oromo. The main idea of this thesis design and develop MT system in order to give a comprehensive comparison of the performance of those three algorithms in the context of our domain, and develop prototype for higher BLEU score result in order to demonstrate and evaluate the usability of prototype, finally evaluate translated sentences quality. At the end explored the challenge we faced during conduct those experiment, as we describe under section 5.9. We have used a small amount of dataset and less number of parameter for our experiment. For evaluating the performance of the proposed system, automatic (BLEU and Perplexity) and manual evaluation methods have been used. The proposed model after 30 epoch we got different BLUE score and perplexity result.

Frist when we compare our proposed models we can see that transformer-based model translated result is better, especially translated from Afaan Oromo to English, and it provided more promising MT performance compared to the both proposed models. The BLUE score for Afaan Oromo to English increased from 6.48, while using the Bi-encoder and decoder LSTM with Attention model, to 15.04 while using the Bi-encoder and decoder GRU with Attention model, to 16.40 while using the Transformer model. And also, perplexity of the model for Afaan Oromo to English decreased from 150.377, while using the Bi-encoder and decoder LSTM with Attention model, to 50.805

while using the Bi-encoder and decoder GRU with Attention model, to 28.188 while using the Transformer model. And also, The BLUE score for English to Afaan Oromo increased from 5.39, while using the Bi-encoder and decoder LSTM with Attention model, to 14.70 while using the Bi-encoder and decoder GRU with Attention model, to 14.13 while using the Transformer model. And also, perplexity of the model for English to Afaan Oromo decreased from 268.844, while using the Bi-encoder and decoder LSTM with Attention model, to 79.598 while using the Bi-encoder and decoder GRU with Attention model, to 66.776 while using the Transformer model.

Bi-encoder and decoder GRU with Attention model are still the best compared to Transformers choice when the sequence length is too long as we have seen on Figure 5:7. Bidirectional target-attention model was successfully applied in the LSTM and GRU algorithms. This means that both forward and reverse target-attention information can work together well. Through this thesis work, we have understood the basic difference between the LSTM, GRU and transformer units. Transformer uses less training parameter and therefore uses less memory and executes faster than LSTM as well as GRU, whereas GRU is more accurate than LSTM on our dataset. One can choose GRU if you are dealing with large sequences and accuracy is concerned, LSTM is used when you have less memory consumption and want faster results. Additionally, despite having fewer parameters, the Transformer model was able to achieve a lower loss after 30 epochs. In simple words, we have observed in our experiments, that deep learning models are effective and efficient for low-resourced languages such as Afaan Oromo, but sensitive to hyper-parameters and quality of corpus.

The most useful part of a BLEU score is that it can be used to compare different models on the same dataset, where the one with the higher BLEU score is better. Thus, we got a BLEU score of ***14.13*** from English to Afaan Oromo and ***16.4*** from Afaan Oromo to English on Transformer model achieves a better result. So we developed a prototype for transformer model. Finally evaluated that prototype usability and translated sentences quality by using manual evaluation method. At the end, developed prototype was accepted by the evaluators with 78.01% of the results. Overall they gave us very nice and encouraging comments from the evaluators. When we come to the overall manual evaluation translation quality result, from Afaan Oromo to English 73.69 % and from English to Afaan Oromo 52.38 % we have get. Given the closeness of most systems and the wide over-lapping confidence intervals it is hard to make strong statements about the correlation between human judgements and automatic scoring methods such as BLEU. But as we understand it, manual evaluation method are very tiring and exhausting, however, for the words that are written in different spelling but have the same meaning, manual evaluation method is better. But manual evaluation, depend on the levels of the evaluator knowledge. So it is difficult to get the same result on the same translated sentence, by using different evaluators

In all proposed models translation result indicates as higher BLUE score and low perplexity from Afaan Oromo to English are more efficient and better result than English to Afaan Oromo as well as in manual evaluation. This indicate that the translation from morphological simple language to morphological rich language is more challenged for MT. Morphological distinctions not present in the source language need to be generated in the target language. Afaan Oromo language are morphologically rich than English language. So the result from morphologically rich language to simple morphology language is high. This is the reason behind we obtained high result from Afaan Oromo to English.

6.2. Contribution of the Study

In the last 10 years the performance of MT has drastically increased. Even from year to year, performance of the state-of-the-art systems increases noticeably. Every year more and more dataset become available with faster and faster computers. Also, the number of translation systems being developed, both commercially and in research environments is increasing. This increasing interest in MT has resulted in many translation systems available of differing design and quality. May be, these improvements come from emerging of new algorithms, increasing the availability of parallel corpus and also availability of different approaches. So as the number of approach’s and algorithms increases, identifying which approaches and algorithms best for low resource language such as our domain and increasing parallel corpus for low resource language is one challenge. So, in order to identify best approach and algorithms we must conduct the experiment and preparing parallel corpus is important point. The other one, all deployed system and algorithms need improvement in order to increase their effectiveness and efficiency. So, in order to improve their effectiveness and efficiency, we must know the challenge about their domain. Then, when we come to our thesis work result, our trained models are not deployed for end user. But, we demonstrated the prototype to evaluators. Our parallel corpus can be used as a great input for other MT research and NLP fields. Because we have prepared around 17105 new parallel corpus. And also other researcher without conducting the experiment easily understood the problem that we explored about the domain. Generally, the output of this thesis can be used as a great input for future research works in the same area and also provide great knowledge about NLP, particularly about MT.

6.2 Future Works and Suggestions

In this thesis work, we describe different methods that influence these pre-existing systems as tools for research in MT and related fields. Suggest a few ways we could improve the model to better match the results and further as a continuation of this study. First suggest techniques for improving a translation system using additional parallel corpus and a better architecture to improve the translation quality. The corpus taken for this study was not enough and also not cover all domain of the language pairs. And also we improve the result by using larger model and ensemble. By combining multiple model's outputs into one, we may obtain better predictive performance or enhance the translation result. As the objective of our study is to implement MT only at level of text to text translation, we recommend that speech to speech MT between these language pair will be studied in future work.

As the length of sentence increase, the performance of the models degrades. This is also another challenge to maintain translation quality of the system on longer sentences. We have shown results on sentences having length up to 66 and 55 for English and Afaan Oromo words. The performance of all proposed models decreases on sentences where length increases, which may be need improvement by adding sentence length in the training dataset and changing hyper-parameter.

Reasons of OOV or rare words includes word missing in training dataset. Misspelled words that usually cannot be translated. In order to handle spelling error for Afaan Oromo writing system we must develop auto spelling systems, and we suggest this in future study and, we suggest try to conduct the experiment by preparing parallel corpus that have proverb and idiomatic sentences and also the different word that have the same meaning in order to overcame that translation divergences problems as seen in most translation systems.

References

[1] A. Gebremariam and Y. Assabie, “Amharic-to-Tigrigna Machine Translation Using Hybrid Approach,” p. 108, Oct. 2017.

[2] N. Alsohybe, N. Dahan, and F. Ba-Alwi, “Machine-Translation History and Evolution: Survey for Arabic-English Translations,” Curr. J. Appl. Sci. Technol., vol. 23, no. 4, pp. 1–19, Sep. 2017, doi: 10.9734/CJAST/2017/36124.

[3] S. Yang, Y. Wang, and X. Chu, “A Survey of Deep Learning Techniques for Neural Machine Translation,” ArXiv200207526 Cs, Feb. 2020, Accessed: Jan. 13, 2022. [Online]. Available: http://arxiv.org/abs/2002.07526

[4] D. Bahdanau, K. Cho, and Y. Bengio, “Neural Machine Translation by Jointly Learning to Align and Translate,” ArXiv14090473 Cs Stat, May 2016, [Online]. Available: http://arxiv.org/abs/1409.0473

[5] S. Yitayew, “Optimal Alignment for Bi-directional Afaan Oromo-English Statistical Machine Translation,” A.A.U, 2017. [Online]. Available: http://etd.aau.edu.et/handle/123456789/14063

[6] X. Ariel and C. Wenhan, “Russian-English Bidirectional Machine Translation System,” p. 6, Nov. 2020.

[7] S. Dereje, “Hybrid Artificial Neural Machine Translation using Deep Learning Techniques English-to-Afaan Oromoo,” ASTU, 2019. [Online]. Available: http://213.55.101.20:8080/xmlui/handle/123456789/1555

[8] “Top Languages of the Internet, Today and Tomorrow.” https://resources.unbabel.com/blog/top-languages-of-the-internet (accessed Jun. 12, 2022).

[9] H. Naser, “Let’s Talk Artificial Intelligence and Its Subsets,” VEXXHOST, Jun. 02, 2020. https://vexxhost.com/blog/artificial-intelligence-subsets/ (accessed Nov. 20, 2021).

[10] Serokell, “Artificial Intelligence vs. Machine Learning vs. Deep Learning: What’s the Difference,” Medium. Nov. 23, 2020. [Online]. Available: https://ai.plainenglish.io/artificial-intelligence-vs-machine-learning-vs-deep-learning-whats-the-difference-dccce18efe7f

[11] Bijimol T.K, Dr. John T. “Abraham A Study of Machine Translation methods”, February 2014

[12] S. Yang, Y. Wang, and X. Chu, “A Survey of Deep Learning Techniques for Neural Machine Translation,” ArXiv200207526 Cs, Feb. 2020, [Online]. Available: http://arxiv.org/abs/2002.07526

[13] N. Wayessa and S. Abas, “Multi-Class Sentiment Analysis from Afaan Oromo Text Based On Supervised Machine Learning Approaches,” p. 9, 2020

[14] T. Thomas, “Language Translation,” Github. https://github.com/tommytracey/AIND-Capstone/blob/master/README.md, accessed Jul. 09, 2022

[15] “Gowda et al. - 2021 - Many-to-English Machine Translation Tools, Data, a.pdf.”

[16] P. Koehn and R. Knowles, “Six Challenges for Neural Machine Translation,” in Proceedings of the First Workshop on Neural Machine Translation, Vancouver, 2017, pp. 28–39. doi: 10.18653/v1/W17-3204.

[17] E. Greenstein and D. Penner, “Japanese-to-English Machine Translation Using Recurrent Neural Networks,” p. 7. Jul. 09, 2022

[18] K. Shaalan, A. Hendam, and A. Rafea, “An English-Arabic Bi-directional Machine Translation Tool in the Agriculture Domain,” in Intelligent Information Processing V, vol. 340, Z. Shi, S. Vadera, A. Aamodt, and D. Leake, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 281–290. [Online]. Available: http://link.springer.com/10.1007/978-3-642-16327-2_34

[19] S. Laskar, A. Dutta, Dr. P. Pakray, and S. Bandyopadhyay, “Neural Machine Translation: English to Hindi,” Dec. 2019, pp. 1–6. doi: 10.1109/CICT48419.2019.9066238.

[20] T. D. Singh and S. Bandyopadhyay, “Manipuri-English Bidirectional Statistical Machine Translation Systems using Morphology and Dependency Relations,” p. 9.

[21] “Top ten most spoken language in the world,” listverse.com. https://www.listverse.com, accessed Jul. 09, 2022

[22] Biruk Abel, “Geez to Amharic Machine Translation,” p. 94, May, 2018

[23] A. Gebremariam and Y. Assabie, “Amharic-to-Tigrigna Machine Translation Using Hybrid Approach,” p. 108, Oct. 2017.

[24] A. Yeabsira, “Context Based Machine Translation with Recurrent Neural Network for English - Amharic Translation,” Addis Ababa University, 2020. [Online]. Available: http://etd.aau.edu.et/handle/123456789/22639

[25] B. Arfaso, “Bi-Directional English-Afan Oromo Machine Translation Using Convolutional Neural Network,” Addis Ababa University, 2019. [Online]. Available: http://etd.aau.edu.et/handle/123456789/20924

[26] J. R. Venable, J. Pries-Heje, and R. L. Baskerville, “Choosing a Design Science Research Methodology,” p. 12, 2017.

[27] S. O’Riain, E. Curry, and P. Buitelaar, “Engaging Practitioners within Design Science Research: A Natural Language Processing Case Study,” Dec. 2012, vol. 388. doi: 10.1007/978-3-319-04090-5_14.

[28] J. R. Venable, J. Pries-Heje, and R. L. Baskerville, “Choosing a Design Science Research Methodology,” p. 12, 2017.

[29] N. Ganatra and A. Patel, “A Comprehensive Study of Deep Learning Architectures, Applications and Tools,” Int. J. Comput. Sci. Eng., vol. 6, pp. 701–705, Dec. 2018, doi: 10.26438/ijcse/v6i12.701705.

[30] “Okpor - 2014 - Machine Translation Approaches Issues and Challen.pdf.” Accessed: Dec. 15, 2021. [Online]. Available: https://www.ijcsi.org/papers/IJCSI-11-5-2-159-165.pdf

[31] N. Alsohybe, N. Dahan, and F. Ba-Alwi, “Machine-Translation History and Evolution: Survey for Arabic-English Translations,” Curr. J. Appl. Sci. Technol., vol. 23, no. 4, pp. 1–19, Sep. 2017, doi: 10.9734/CJAST/2017/36124.

[32] A. Garg and M. Agarwal, “Machine Translation: A Literature Review,” Dec. 2018, [Online]. Available: http://arxiv.org/abs/1901.01122

[33] E. Greenstein and D. Penner, “Japanese-to-English Machine Translation Using Recurrent Neural Networks,” p. 7.

[34] P. Koehn and R. Knowles, “Six challenges for neural machine translation,” ArXiv Prepr. ArXiv170603872, 2017.

[35] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A Survey of the Recent Architectures of Deep Convolutional Neural Networks,” Artif. Intell. Rev., vol. 53, no. 8, pp. 5455–5516, Dec. 2020, doi: 10.1007/s10462-020-09825-6.

[36] Moussallem et al. “Machine Translation using Semantic Web Technologies”, 2018

[37] X. Zeng, W. Ouyang, B. Yang, J. Yan, and X. Wang, “Gated Bi-directional CNN for Object Detection,” in Computer Vision – ECCV 2016, vol. 9911, B. Leibe, J. Matas, N.

Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 354–369. [Online]. Available: http://link.springer.com/10.1007/978-3-319-46478-7_22

[38] N. Ganatra and A. Patel, “A Comprehensive Study of Deep Learning Architectures, Applications and Tools,” Int. J. Comput. Sci. Eng., vol. 6, pp. 701–705, Dec. 2018, doi: 10.26438/ijcse/v6i12.701705.

[39] M. S. Maučec and G. Donaj, Machine Translation and the Evaluation of Its Quality. IntechOpen, 2019. [Online]. Available: https://www.intechopen.com/chapters/68953

[40] M. Albadr, S. Tiun, and F. Al-Dhief, “Evaluation of machine translation systems and related procedures,” J. Eng. Appl. Sci., vol. 13, pp. 3961–3972, Jun. 2018.

[41] M. Albadr, S. Tiun, and F. Al-Dhief, “Evaluation of machine translation systems and related procedures,” J. Eng. Appl. Sci., vol. 13, pp. 3961–3972, Jun. 2018.

[42] M. S. Maučec and G. Donaj, Machine Translation and the Evaluation of Its Quality. IntechOpen, 2019. doi: 10.5772/intechopen.89063.

[43] Anugu and Ramesh “A Survey on Hybrid Machine Translation”, 2020

[44] Moussallem et al. “Machine Translation using Semantic Web Technologies”, 2018

[45] Z. Tan et al., “Neural machine translation: A review of methods, resources, and tools,” AI Open, vol. 1, pp. 5–21, Jan. 2020, doi: 10.1016/j.aiopen.2020.11.001.

[46] H. Wang, H. Wu, Z. He, L. Huang, and K. Church, “Progress in Machine Translation,” Engineering, Jul. 2021, doi: 10.1016/j.eng.2021.03.023.

[47] M. D. Okpor, “Machine Translation Approaches: Issues and Challenges,” vol. 11, no. 5, p. 7, 2014.

[48] J. Mtsweni, E. Biermann, and L. Pretorius, “iSemServ: A model-driven approach for developing semantic web services,” South Afr. Comput. J., vol. 52, Jun. 2014, doi: 10.18489/sacj.v52i0.185.

[49] M. Escribe, “Human Evaluation of Neural Machine Translation: The Case of Deep Learning,” in Proceedings of the Human-Informed Translation and Interpreting Technology Workshop (HiT-IT 2019), Varna, Bulgaria, Sep. 2019, pp. 36–46. doi: 10.26615/issn.2683-0078.2019_005.

[50] P. Li, “A Survey of Machine Translation Methods,” TELKOMNIKA Indones. J. Electr. Eng., vol. 11, Jul. 2013, doi: 10.11591/telkomnika.v11i12.2780.

[51] Jinbin, “Perplexity: Your PP Metric,” unpackAI. Feb. 01, 2021. [Online]. Available: https://medium.com/unpackai/perplexity-your-pp-metric-3dd96441f6d4

[52] C. Chelba and F. Pereira, “Multinomial Loss on Held-out Data for the Sparse Non-negative Matrix Language Model,” ArXiv151101574 Cs, Feb. 2016, [Online]. Available: http://arxiv.org/abs/1511.01574

[53] J. Zhou, Y. Cao, X. Wang, P. Li, and W. Xu, “Deep recurrent models with fast-forward connections for neural machine translation,” Trans. Assoc. Comput. Linguist., vol. 4, pp. 371–383, 2016.

[54] M. Johnson et al., “Google’s multilingual neural machine translation system: Enabling zero-shot translation,” Trans. Assoc. Comput. Linguist., vol. 5, pp. 339–351, 2017.

[55] D. Goswami, “Comparison of Sigmoid, Tanh and ReLU Activation Functions,” AITUDE. Aug. 19, 2020. [Online]. Available: https://www.aitude.com/comparison-of-sigmoid-tanh-and-relu-activation-functions/

[56] Mishra and Gupta “Deep Machine Learning and Neural Networks an Overview” 2017

[57] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A Survey of the Recent Architectures of Deep Convolutional Neural Networks,” Artif. Intell. Rev., vol. 53, no. 8, pp. 5455–5516, Dec. 2020, doi: 10.1007/s10462-020-09825-6.

[58] M. Bahi and M. Batouche, “Deep Learning for Ligand-Based Virtual Screening in Drug Discovery,” Oct. 2018, pp. 1–5. doi: 10.1109/PAIS.2018.8598488.

[59] T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent Trends in Deep Learning Based Natural Language Processing,” ArXiv170802709 Cs, Nov. 2018, [Online]. Available: http://arxiv.org/abs/1708.02709

[60] X. Ma and E. Hovy, “End-to-end sequence labeling via bi-directional lstm-cnns-crf,” ArXiv Prepr. ArXiv160301354, 2016.

[61] S. Chowdhury et al., “A multitask bi-directional RNN model for named entity recognition on Chinese electronic medical records,” BMC Bioinformatics, vol. 19, no. S17, p. 499, Dec. 2018, doi: 10.1186/s12859-018-2467-9.

[62] H. Kim and J.-H. Lee, “A recurrent neural networks approach for estimating the quality of machine translation output,” in Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 494–498.

[63] M. Escribe, “Human Evaluation of Neural Machine Translation: The Case of Deep Learning,” in Proceedings of the Human-Informed Translation and Interpreting Technology Workshop (HiT-IT 2019), Varna, Bulgaria, Sep. 2019, pp. 36–46. doi: 10.26615/issn.2683-0078.2019_005.

[64] “Recurrent Neural Network (RNN) Tutorial for Beginners,” Simplilearn.com. [Online]. Available: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

[65] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training Recurrent Neural Networks,” ArXiv12115063 Cs, Feb. 2013, [Online]. Available: http://arxiv.org/abs/1211.5063

[66] Z. Hu, J. ZHANG, and Y. Ge, “Handling Vanishing Gradient Problem Using Artificial Derivative,” IEEE Access, vol. PP, pp. 1–1, Jan. 2021, doi: 10.1109/ACCESS.2021.3054915.

[67] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” Dec. 2014.

[68] A. Vaswani et al., “Attention Is All You Need,” ArXiv170603762 Cs, Dec. 2017, [Online]. Available: http://arxiv.org/abs/1706.03762

[69] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” ArXiv Prepr. ArXiv150804025, 2015.

[70] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” ArXiv Prepr. ArXiv14091259, 2014.

[71] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” ArXiv Prepr. ArXiv150804025, 2015.

[72] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, “Addressing the rare word problem in neural machine translation,” ArXiv Prepr. ArXiv14108206, 2014.

[73] D. Datta, P. Evangeline, D. Mittal, and A. Jain, “Neural Machine Translation using Recurrent Neural Network,” Int. J. Eng. Adv. Technol., vol. 9, pp. 1395–1400, Apr. 2020, doi: 10.35940/ijeat.D7637.049420.

[74] M. Mijwil, “Brief History of the English language,” Feb. 2018.

[75] K. K. Tune, V. Varma, P. Pingali, K. K. Tune, V. Varma, and P. Pingali, “Evaluation of Oromo-English Cross-Language Information Retrieval,” 2007.

[76] B. Anbase, “Applications of Information Retrieval for Afaan Oromo text based on Semantic based Indexing,” p. 93, 2019.

[77] C. Wakuma Olbasa, “Choice for a Working Language in Ethiopia: A case study among graduating classes of Oromo speakers in selected public universities,” Macrolinguistics, vol. 6, no. 9, pp. 98–115, Dec. 2018, doi: 10.26478/ja2018.6.9.9.

[78] A. Eisele, “English – Oromo Machine Translation: An Experiment Using a Statistical Approach,” p. 4.

[79] D. Tesfaye, “A rule-based Afan Oromo Grammar Checker,” Int. J. Adv. Comput. Sci. Appl., vol. 2, no. 8, 2011, doi: 10.14569/IJACSA.2011.020823.

[80] N. Wayessa and S. Abas, “Multi-Class Sentiment Analysis from Afaan Oromo Text Based On Supervised Machine Learning Approaches,” p. 9, 2019.

[81] Federico Gaspari, “A survey of machine translation competences” p. 11. February 2015

[82] Rui Wang , Xu Tan et all., “A Survey on Low-Resource Neural Machine Translation,” 2017. Accessed: Jan. 21, 2022. [Online]. Available: https://www.ijcai.org/proceedings/2021/0629.pdf

[83] T. Shah, “About Train, Validation and Test Sets in Machine Learning,” Medium. Jul. 10, 2020. [Online]. Available: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7

[84] R. Kohavi, “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection,” vol. 14, Mar. 2001.

[85] “Splitting into train, dev and test sets.” https://cs230.stanford.edu/blog/split/ (accessed Jul. 21, 2021).

[86] J. O. Z. M. D. F. für G. und U. H. · I. Asthma and A. PreventionPhD, “Jimmy OMONY Institute for Asthma and Allergy Prevention (IAP),” ResearchGate. [Online]. Available: https://www.researchgate.net/profile/Jimmy-Omony

[87] J. Brownlee, “Data Leakage in Machine Learning,” Machine Learning Mastery. Aug. 01, 2016. [Online]. Available: https://machinelearningmastery.com/data-leakage-machine-learning/

[88] Dr. Sinisa Colic, “Using Google Colab for a coding activity.” [Online]. Available: https://ito-engineering.screenstepslive.com/s/ito_fase/m/96777/l/1269075-using-google-colab-for-a-coding-activity-prof-sinisa-colic

[89] “Introduction to PyTorch for Deep Learning,” KDnuggets. [Online]. Available: https://www.kdnuggets.com/introduction-to-pytorch-for-deep-learning.html/

[90] J. Kreutzer, J. Bastings, and S. Riezler, “Joey NMT: A Minimalist NMT Toolkit for Novices,” ArXiv190712484 Cs, Jun. 2020, [Online]. Available: http://arxiv.org/abs/1907.12484

[91] G. Klein, F. Hernandez, V. Nguyen, and J. Senellart, “The OpenNMT Neural Machine Translation Toolkit: 2020 Edition,” vol. 1, p. 8, 2020.

[92] K. Nyuytiymbiy, “Parameters and Hyperparameters in Machine Learning and Deep Learning,” Medium. Jan. 15, 2022. [Online]. Available: https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac

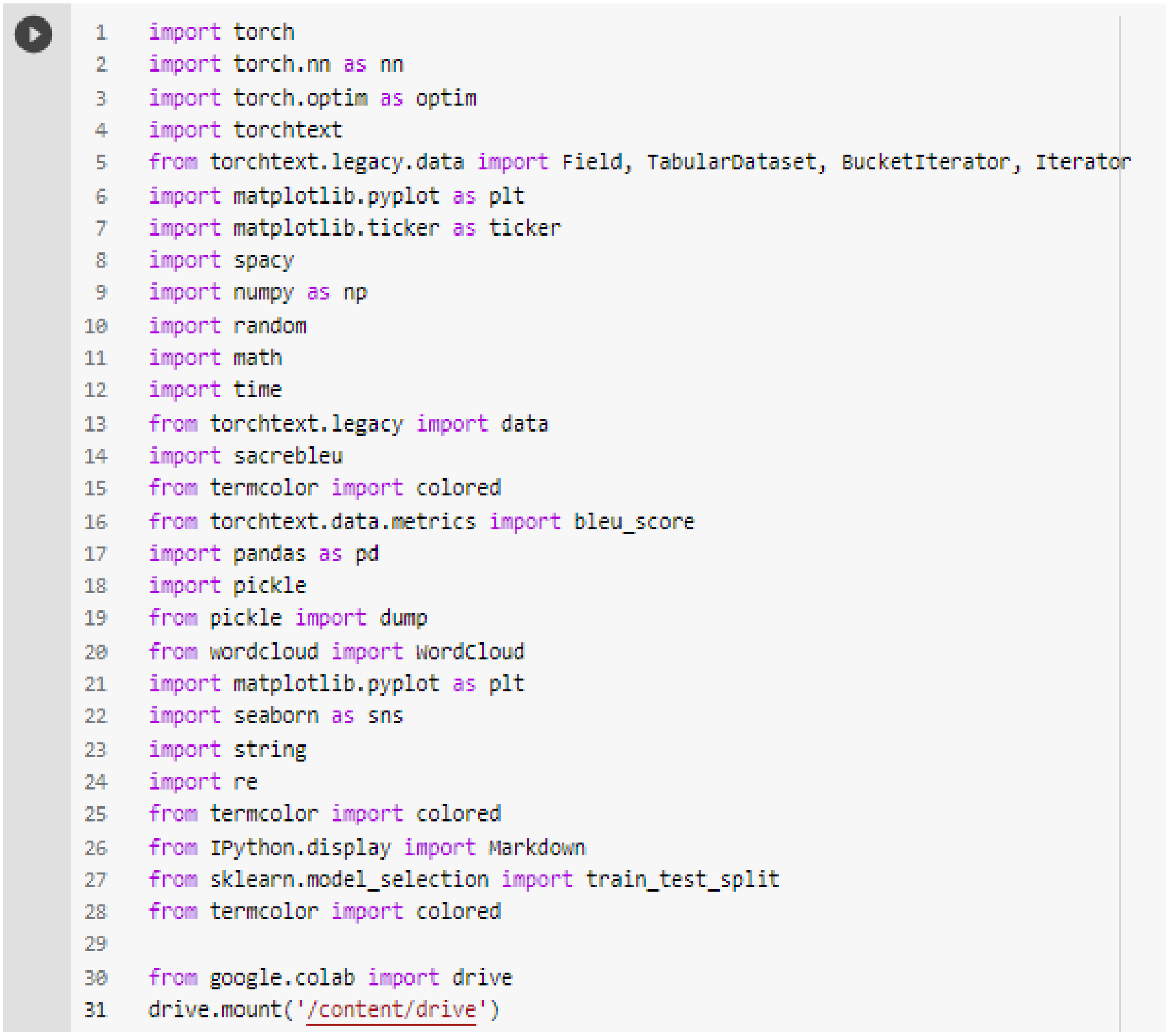
[93] “Machine Learning Inference,” Hazelcast. [Online]. Available: https://hazelcast.com/glossary/machine-learning-inference/ (accessed Jul. 09, 2022)

[94] S. Jones, “The attributes of usability and how to exploit them,” Econsultancy, Feb. 18, 2014. https://econsultancy.com/the-attributes-of-usability-and-how-to-exploit-them/ (accessed Jul. 09, 2022).

[95] S. Doherty and S. Brien, “A User-Based Usability Assessment of Raw Machine Translated Technical Instructions,” Dec. 2012.

Appendix

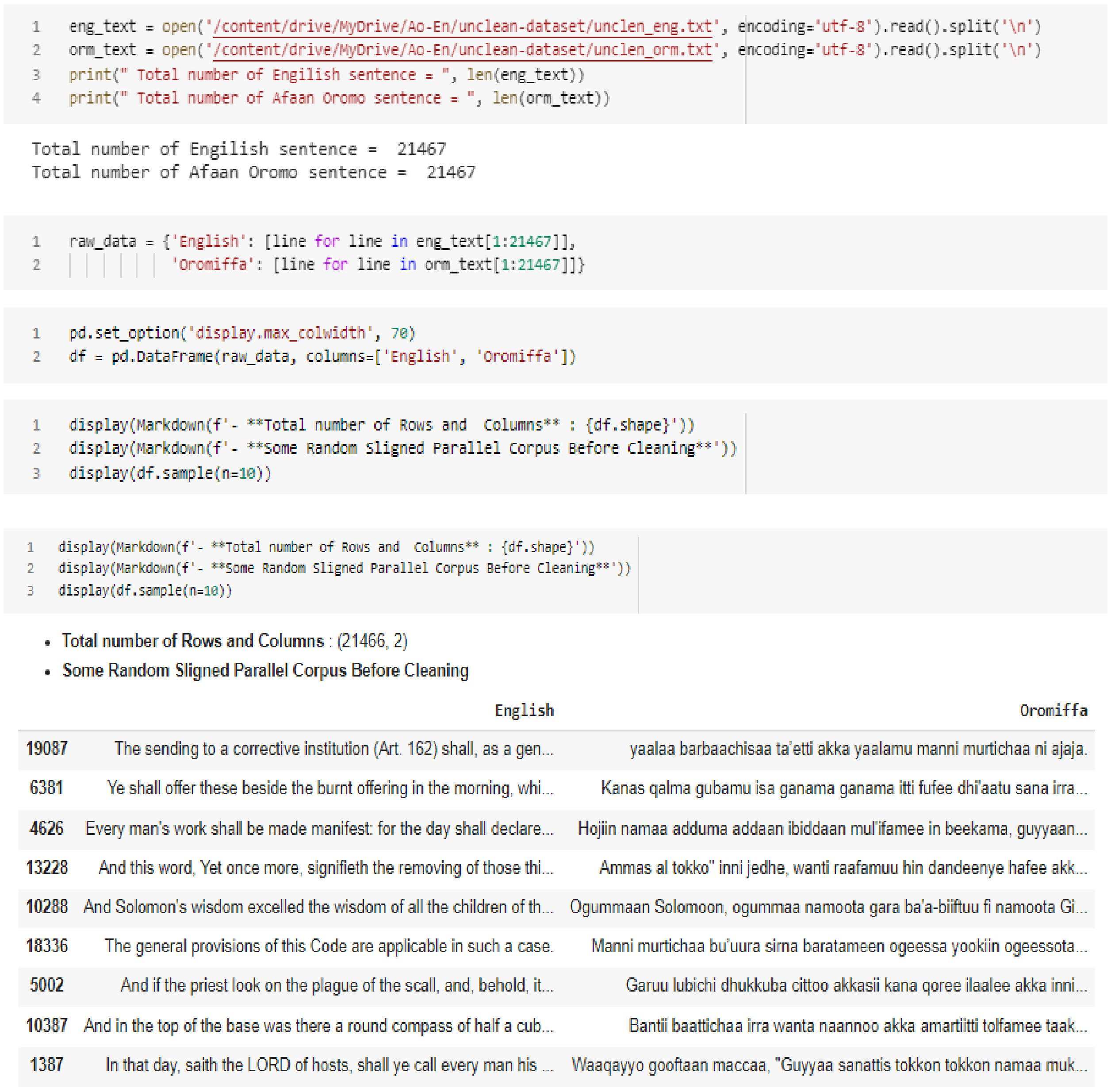
Appendix A: Imort all requerd library and mount google dirve in order to access file from Google drive on Colab Notebook.



```python
import torch
import torch.nn as nn
import torch.optim as optim
import torchtext
from torchtext.legacy.data import Field, TabularDataset, BucketIterator, Iterator
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
import spacy
import numpy as np
import random
import math
import time
from torchtext.legacy import data
import sacrebleu
from termcolor import colored
from torchtext.data.metrics import bleu_score
import pandas as pd
import pickle
from pickle import dump
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import seaborn as sns
import string
import re
from termcolor import colored
from IPython.display import Markdown
from sklearn.model_selection import train_test_split
from termcolor import colored

from google.colab import drive
drive.mount('/content/drive')
```

Appendix-Figure: I Snapshot of python Code for Import all required library and mount google drive

Appendix B: Python Code for read and show sample of parallel corpus before cleaning



```
eng_text = open('/content/drive/MyDrive/Ao-En/unclean-dataset/unclen_eng.txt', encoding='utf-8').read().split('\n')
orm_text = open('/content/drive/MyDrive/Ao-En/unclean-dataset/unclen_orm.txt', encoding='utf-8').read().split('\n')
print(" Total number of Engilish sentence = ", len(eng_text))
print(" Total number of Afaan Oromo sentence = ", len(orm_text))
```

```
Total number of Engilish sentence =  21467
Total number of Afaan Oromo sentence =  21467
```

```
raw_data = {'English': [line for line in eng_text[1:21467]],
            'Oromiffa': [line for line in orm_text[1:21467]]}
```

```
pd.set_option('display.max_colwidth', 70)
df = pd.DataFrame(raw_data, columns=['English', 'Oromiffa'])
```

```
display(Markdown(f'- **Total number of Rows and  Columns** : {df.shape}'))
display(Markdown(f'- **Some Random Sligned Parallel Corpus Before Cleaning**'))
display(df.sample(n=10))
```

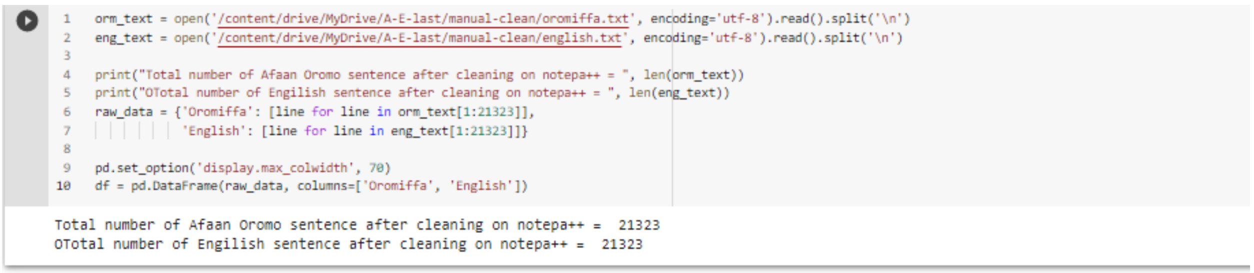
```
display(Markdown(f'- **Total number of Rows and  Columns** : {df.shape}'))
display(Markdown(f'- **Some Random Sligned Parallel Corpus Before Cleaning**'))
display(df.sample(n=10))
```

- **Total number of Rows and Columns** : (21466, 2)
- **Some Random Sligned Parallel Corpus Before Cleaning**

	English	Oromiffa
19087	The sending to a corrective institution (Art. 162) shall, as a gen...	yaalaa barbaachisaa ta'etti akka yaalamu manni murtichaa ni ajaja.
6381	Ye shall offer these beside the burnt offering in the morning, whi...	Kanas qalma gubamu isa ganama ganama itti fufee dhi'aatu sana irra...
4626	Every man's work shall be made manifest: for the day shall declare...	Hojiin namaa adduma addaan ibiddaan mul'ifamee in beekama, guyyaan...
13228	And this word, Yet once more, signifieth the removing of those thi...	Ammas al tokko" inni jedhe, wanti raafamuu hin dandeenye hafee akk...
10288	And Solomon's wisdom excelled the wisdom of all the children of th...	Ogummaan Solomoon, ogummaa namoota gara ba'a-biiftuu fi namoota Gi...
18336	The general provisions of this Code are applicable in such a case.	Manni murtichaa bu'uura sirna baratameen ogeessa yookiin ogeessota...
5002	And if the priest look on the plague of the scall, and, behold, it...	Garuu lubichi dhukkuba cittoo akkasii kana qoree ilaalee akka inni...
10387	And in the top of the base was there a round compass of half a cub...	Bantii baattichaa irra wanta naannoo akka amartiitti tolfamee taak...
1387	In that day, saith the LORD of hosts, shall ye call every man his ...	Waaqayyo gooftaan maccaa, "Guyyaa sanattis tokkon tokkon namaa muk...

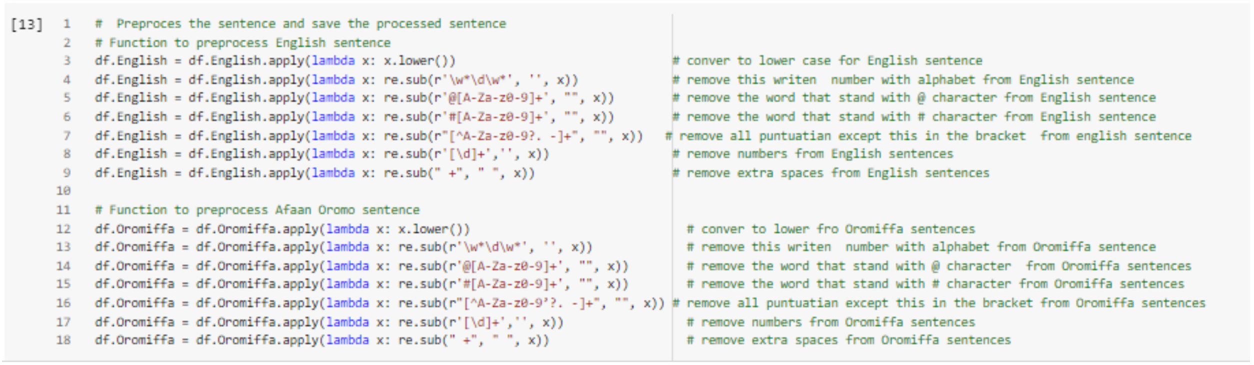
Appendix-Figure: II Snapshot for fragment code for read and show sample of parallel corpus before cleaning

Appendix C: Python Code for cleaning and show some sample of parallel corpus after cleaning

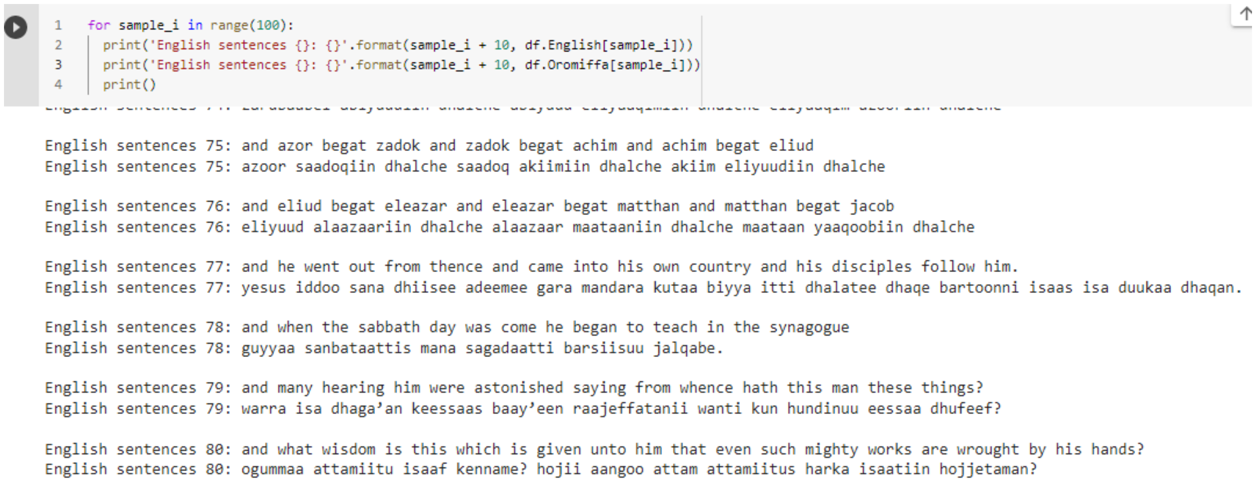


```python
orm_text = open('/content/drive/MyDrive/A-E-last/manual-clean/oromiffa.txt', encoding='utf-8').read().split('\n')
eng_text = open('/content/drive/MyDrive/A-E-last/manual-clean/english.txt', encoding='utf-8').read().split('\n')

print("Total number of Afaan Oromo sentence after cleaning on notepa++ = ", len(orm_text))
print("OTotal number of Engilish sentence after cleaning on notepa++ = ", len(eng_text))
raw_data = {'Oromiffa': [line for line in orm_text[1:21323]],
            'English': [line for line in eng_text[1:21323]]}

pd.set_option('display.max_colwidth', 70)
df = pd.DataFrame(raw_data, columns=['Oromiffa', 'English'])
```

```
Total number of Afaan Oromo sentence after cleaning on notepa++ =  21323
OTotal number of Engilish sentence after cleaning on notepa++ =  21323
```



```python
[13]
#  Preproces the sentence and save the processed sentence
# Function to preprocess English sentence
df.English = df.English.apply(lambda x: x.lower())                          # conver to lower case for English sentence
df.English = df.English.apply(lambda x: re.sub(r'\w*\d\w*', '', x))         # remove this writen  number with alphabet from English sentence
df.English = df.English.apply(lambda x: re.sub(r'@[A-Za-z0-9]+', "", x))    # remove the word that stand with @ character from English sentence
df.English = df.English.apply(lambda x: re.sub(r'#[A-Za-z0-9]+', "", x))    # remove the word that stand with # character from English sentence
df.English = df.English.apply(lambda x: re.sub(r"[^A-Za-z0-9?. -]+", "", x))  # remove all puntuatian except this in the bracket  from english sentence
df.English = df.English.apply(lambda x: re.sub(r'[\d]+','', x))             # remove numbers from English sentences
df.English = df.English.apply(lambda x: re.sub(" +", " ", x))               # remove extra spaces from English sentences

# Function to preprocess Afaan Oromo sentence
df.Oromiffa = df.Oromiffa.apply(lambda x: x.lower())                          # conver to lower fro Oromiffa sentences
df.Oromiffa = df.Oromiffa.apply(lambda x: re.sub(r'\w*\d\w*', '', x))         # remove this writen  number with alphabet from Oromiffa sentence
df.Oromiffa = df.Oromiffa.apply(lambda x: re.sub(r'@[A-Za-z0-9]+', "", x))    # remove the word that stand with @ character  from Oromiffa sentences
df.Oromiffa = df.Oromiffa.apply(lambda x: re.sub(r'#[A-Za-z0-9]+', "", x))    # remove the word that stand with # character from Oromiffa sentences
df.Oromiffa = df.Oromiffa.apply(lambda x: re.sub(r"[^A-Za-z0-9’?. -]+", "", x)) # remove all puntuatian except this in the bracket from Oromiffa sentences
df.Oromiffa = df.Oromiffa.apply(lambda x: re.sub(r'[\d]+','', x))             # remove numbers from Oromiffa sentences
df.Oromiffa = df.Oromiffa.apply(lambda x: re.sub(" +", " ", x))               # remove extra spaces from Oromiffa sentences
```

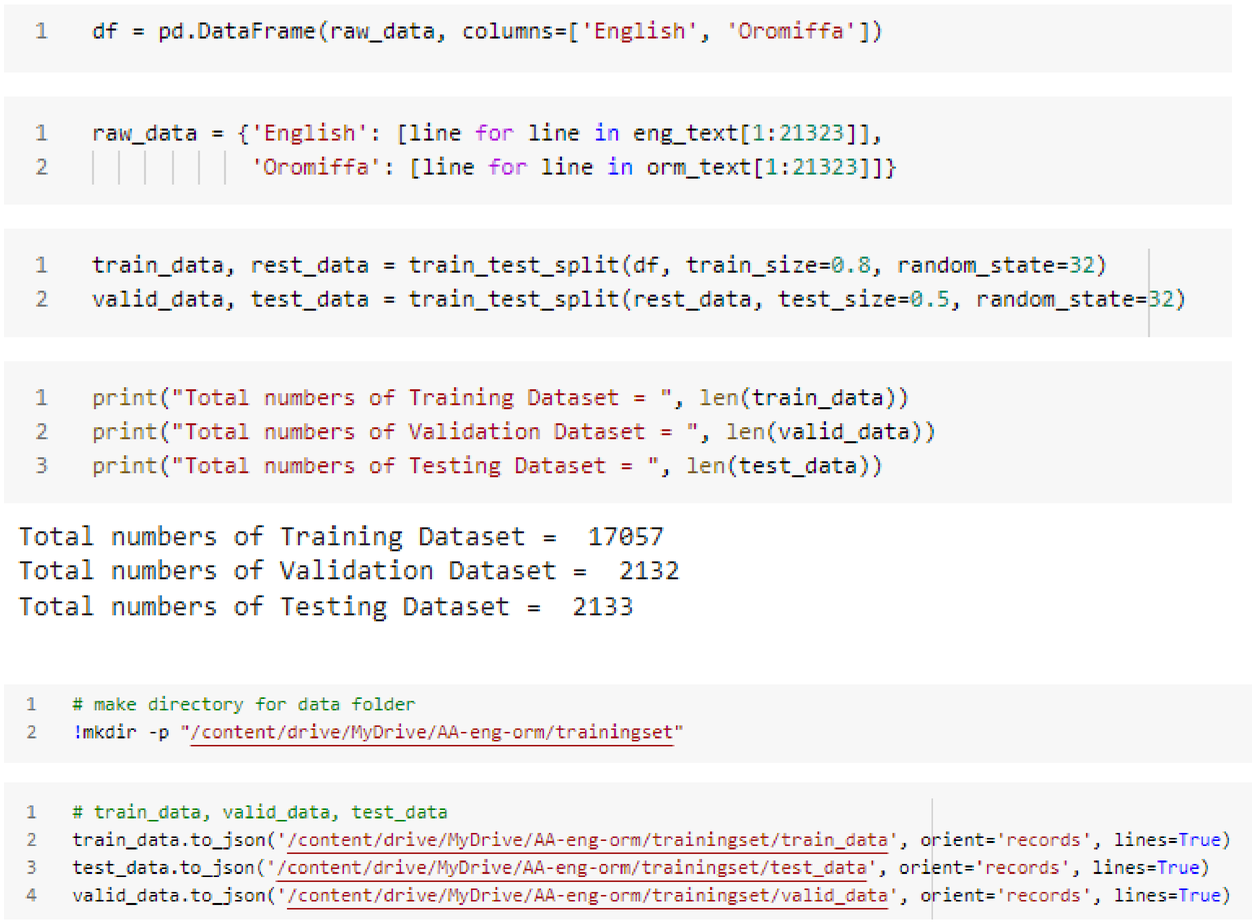


```python
for sample_i in range(100):
  print('English sentences {}: {}'.format(sample_i + 10, df.English[sample_i]))
  print('English sentences {}: {}'.format(sample_i + 10, df.Oromiffa[sample_i]))
  print()
```

```
English sentences 75: and azor begat zadok and zadok begat achim and achim begat eliud
English sentences 75: azoor saadoqiin dhalche saadoq akiimiin dhalche akiim eliyuudiin dhalche

English sentences 76: and eliud begat eleazar and eleazar begat matthan and matthan begat jacob
English sentences 76: eliyuud alaazaariin dhalche alaazaar maataaniin dhalche maataan yaaqoobiin dhalche

English sentences 77: and he went out from thence and came into his own country and his disciples follow him.
English sentences 77: yesus iddoo sana dhiisee adeemee gara mandara kutaa biyya itti dhalatee dhaqe bartoonni isaas isa duukaa dhaqan.

English sentences 78: and when the sabbath day was come he began to teach in the synagogue
English sentences 78: guyyaa sanbataattis mana sagadaatti barsiisuu jalqabe.

English sentences 79: and many hearing him were astonished saying from whence hath this man these things?
English sentences 79: warra isa dhaga’an keessaas baay’een raajeffatanii wanti kun hundinuu eessaa dhufeef?

English sentences 80: and what wisdom is this which is given unto him that even such mighty works are wrought by his hands?
English sentences 80: ogummaa attamiitu isaaf kenname? hojii aangoo attam attamiitus harka isaatiin hojjetaman?
```

Appendix-Figure: III Snapshot of code for cleaning and show some sample of parallel corpus after cleaning

Appendix D: Python Code for Code for split and save the dataset on Google drive



```python
df = pd.DataFrame(raw_data, columns=['English', 'Oromiffa'])
```

```python
raw_data = {'English': [line for line in eng_text[1:21323]],
            'Oromiffa': [line for line in orm_text[1:21323]]}
```

```python
train_data, rest_data = train_test_split(df, train_size=0.8, random_state=32)
valid_data, test_data = train_test_split(rest_data, test_size=0.5, random_state=32)
```

```python
print("Total numbers of Training Dataset = ", len(train_data))
print("Total numbers of Validation Dataset = ", len(valid_data))
print("Total numbers of Testing Dataset = ", len(test_data))
```

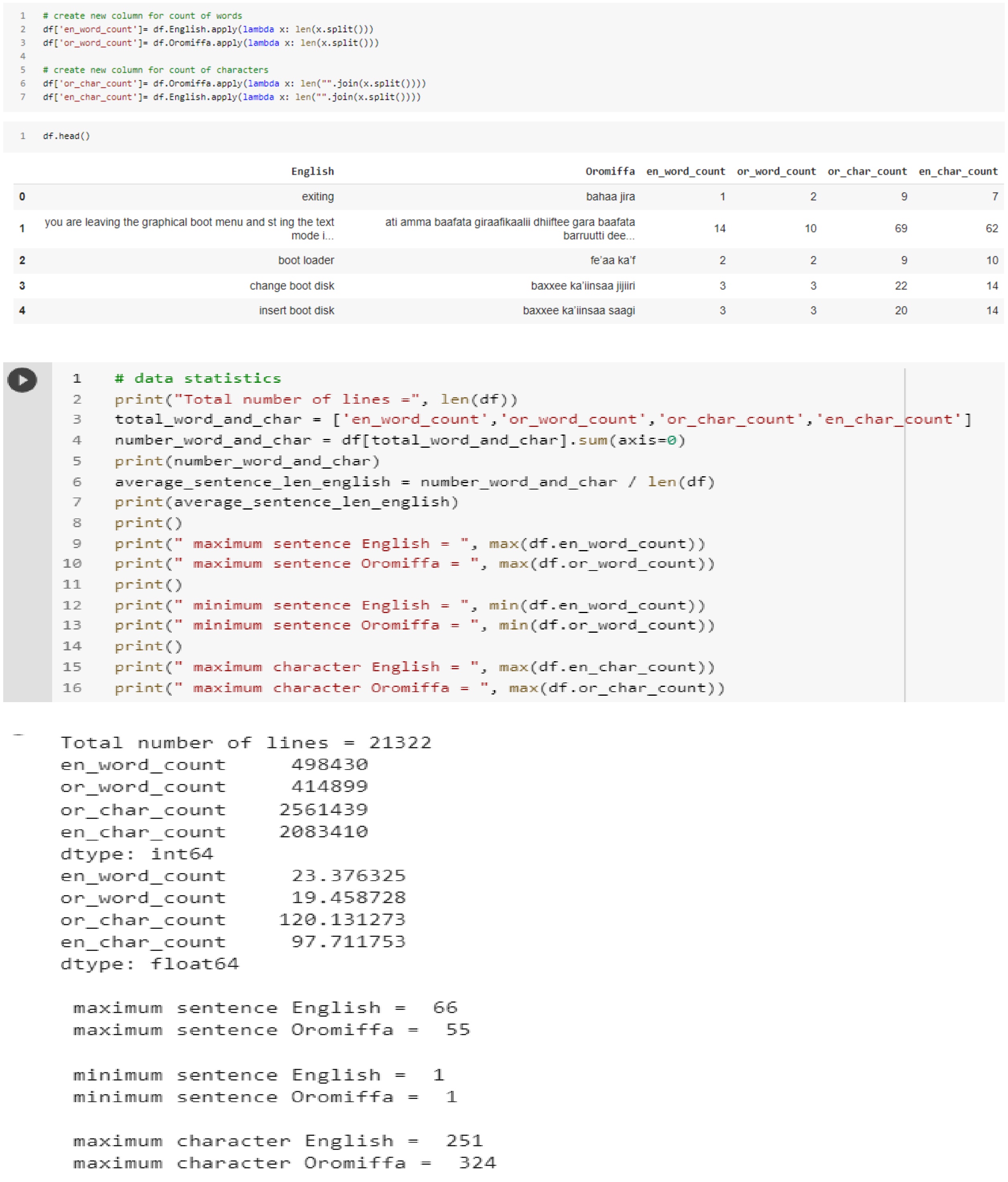
```
Total numbers of Training Dataset =  17057
Total numbers of Validation Dataset =  2132
Total numbers of Testing Dataset =  2133
```

```python
# make directory for data folder
!mkdir -p "/content/drive/MyDrive/AA-eng-orm/trainingset"
```

```python
# train_data, valid_data, test_data
train_data.to_json('/content/drive/MyDrive/AA-eng-orm/trainingset/train_data', orient='records', lines=True)
test_data.to_json('/content/drive/MyDrive/AA-eng-orm/trainingset/test_data', orient='records', lines=True)
valid_data.to_json('/content/drive/MyDrive/AA-eng-orm/trainingset/valid_data', orient='records', lines=True)
```

Appendix-Figure: IV Snapshot of code for Code for split and save the dataset on Google drive

Appendix E: Code for create new column for count words and character and show their details statistics of parallel corpus after cleaning



```python
# create new column for count of words
df['en_word_count']= df.English.apply(lambda x: len(x.split()))
df['or_word_count']= df.Oromiffa.apply(lambda x: len(x.split()))

# create new column for count of characters
df['or_char_count']= df.Oromiffa.apply(lambda x: len("".join(x.split())))
df['en_char_count']= df.English.apply(lambda x: len("".join(x.split())))
```

```python
df.head()
```

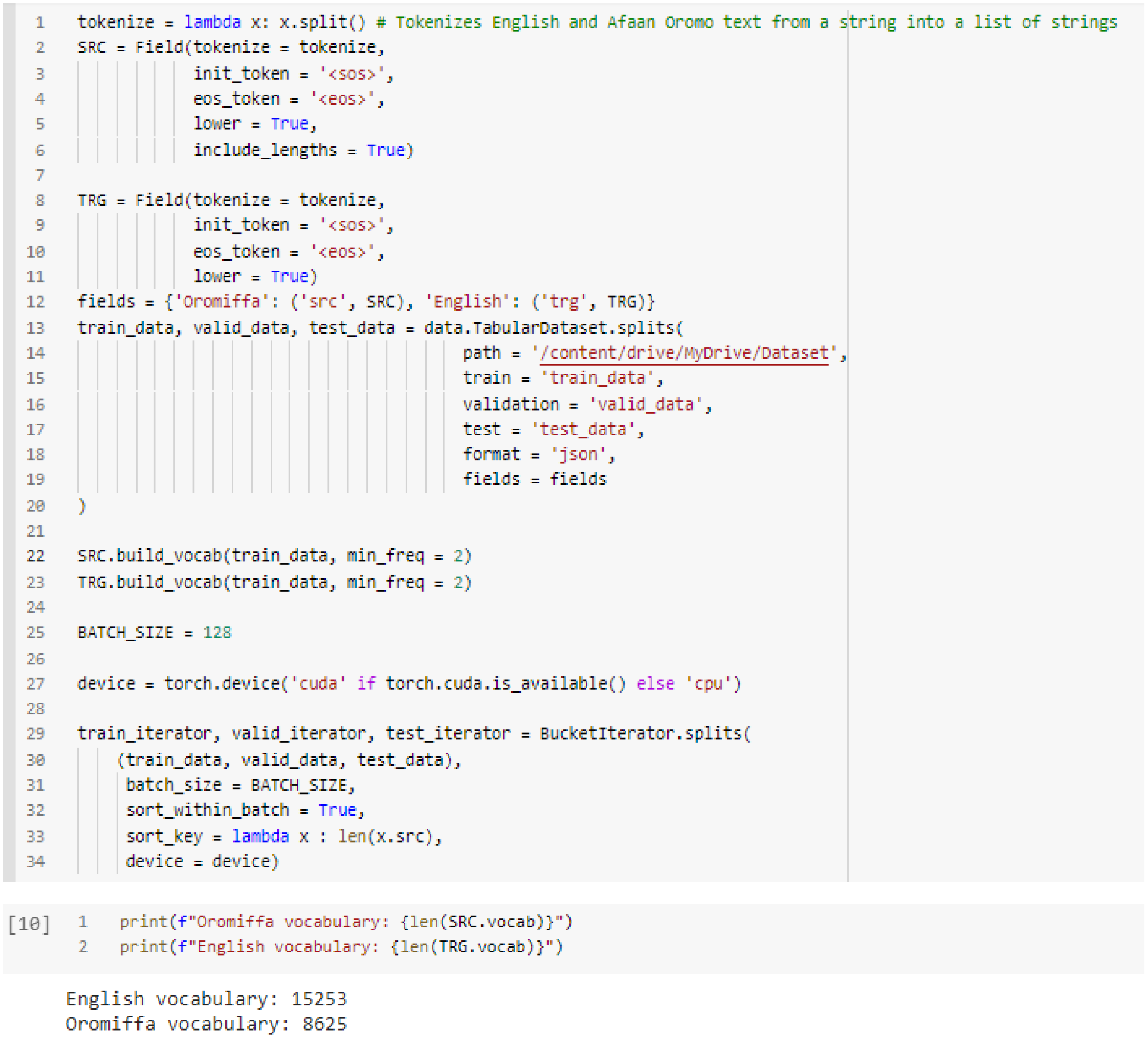
	English	Oromiffa	en_word_count	or_word_count	or_char_count	en_char_count
0	exiting	bahaa jira	1	2	9	7
1	you are leaving the graphical boot menu and st ing the text mode i...	ati amma baafata giraafikaalii dhiiftee gara baafata barruutti dee...	14	10	69	62
2	boot loader	fe'aa ka'f	2	2	9	10
3	change boot disk	baxxee ka'iinsaa jijiiri	3	3	22	14
4	insert boot disk	baxxee ka'iinsaa saagi	3	3	20	14

```python
# data statistics
print("Total number of lines =", len(df))
total_word_and_char = ['en_word_count','or_word_count','or_char_count','en_char_count']
number_word_and_char = df[total_word_and_char].sum(axis=0)
print(number_word_and_char)
average_sentence_len_english = number_word_and_char / len(df)
print(average_sentence_len_english)
print()
print(" maximum sentence English = ", max(df.en_word_count))
print(" maximum sentence Oromiffa = ", max(df.or_word_count))
print()
print(" minimum sentence English = ", min(df.en_word_count))
print(" minimum sentence Oromiffa = ", min(df.or_word_count))
print()
print(" maximum character English = ", max(df.en_char_count))
print(" maximum character Oromiffa = ", max(df.or_char_count))
```

```
Total number of lines = 21322
en_word_count      498430
or_word_count      414899
or_char_count     2561439
en_char_count     2083410
dtype: int64
en_word_count      23.376325
or_word_count      19.458728
or_char_count     120.131273
en_char_count      97.711753
dtype: float64

 maximum sentence English =  66
 maximum sentence Oromiffa =  55

 minimum sentence English =  1
 minimum sentence Oromiffa =  1

 maximum character English =  251
 maximum character Oromiffa =  324
```

Appendix-Figure: V Snapshot of code for create new column for count words and character and show their details statistics of parallel corpus after cleaning

Appendix F: Code for create tokenizers, building vocabulary and iterator for source and target language and print vocabulary size.



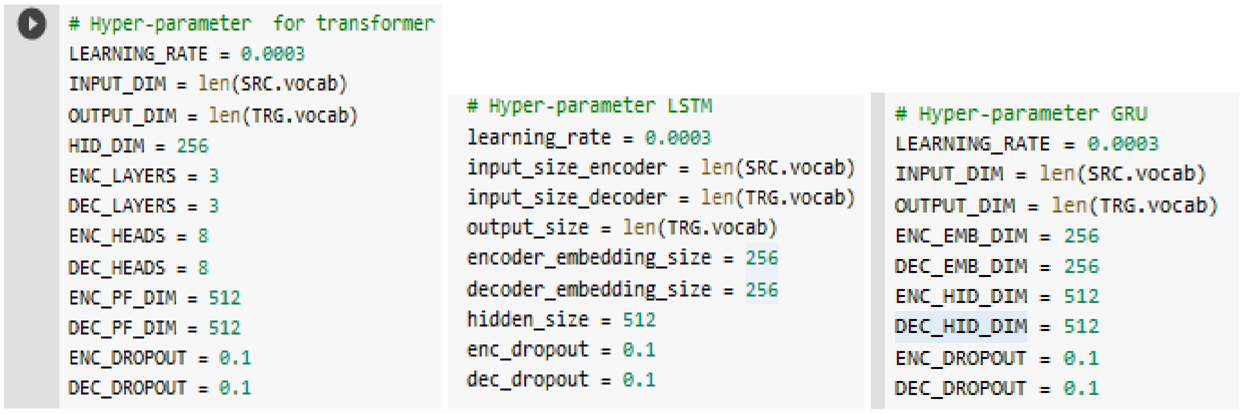
```
tokenize = lambda x: x.split() # Tokenizes English and Afaan Oromo text from a string into a list of strings
SRC = Field(tokenize = tokenize,
            init_token = '<sos>',
            eos_token = '<eos>',
            lower = True,
            include_lengths = True)

TRG = Field(tokenize = tokenize,
            init_token = '<sos>',
            eos_token = '<eos>',
            lower = True)
fields = {'Oromiffa': ('src', SRC), 'English': ('trg', TRG)}
train_data, valid_data, test_data = data.TabularDataset.splits(
                                        path = '/content/drive/MyDrive/Dataset',
                                        train = 'train_data',
                                        validation = 'valid_data',
                                        test = 'test_data',
                                        format = 'json',
                                        fields = fields
)

SRC.build_vocab(train_data, min_freq = 2)
TRG.build_vocab(train_data, min_freq = 2)

BATCH_SIZE = 128

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

train_iterator, valid_iterator, test_iterator = BucketIterator.splits(
    (train_data, valid_data, test_data),
     batch_size = BATCH_SIZE,
     sort_within_batch = True,
     sort_key = lambda x : len(x.src),
     device = device)
```

```
[10] print(f"Oromiffa vocabulary: {len(SRC.vocab)}")
     print(f"English vocabulary: {len(TRG.vocab)}")
```

```
English vocabulary: 15253
Oromiffa vocabulary: 8625
```

Appendix-Figure: VI Snapshot code for create tokenizers, building vocabulary and iterator for source and target language and print vocabulary size.

Appendix G: All proposed models hyper-parameter



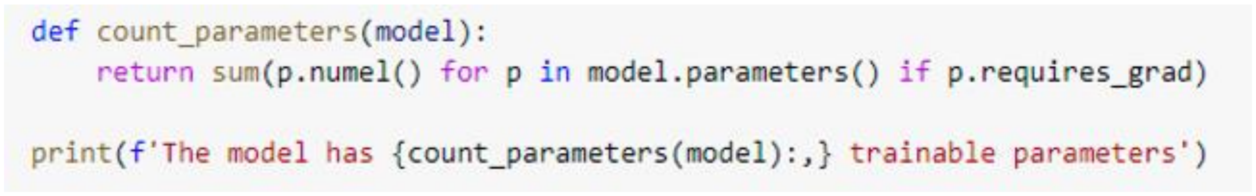
```
# Hyper-parameter  for transformer
LEARNING_RATE = 0.0003
INPUT_DIM = len(SRC.vocab)
OUTPUT_DIM = len(TRG.vocab)
HID_DIM = 256
ENC_LAYERS = 3
DEC_LAYERS = 3
ENC_HEADS = 8
DEC_HEADS = 8
ENC_PF_DIM = 512
DEC_PF_DIM = 512
ENC_DROPOUT = 0.1
DEC_DROPOUT = 0.1
```

```
# Hyper-parameter LSTM
learning_rate = 0.0003
input_size_encoder = len(SRC.vocab)
input_size_decoder = len(TRG.vocab)
output_size = len(TRG.vocab)
encoder_embedding_size = 256
decoder_embedding_size = 256
hidden_size = 512
enc_dropout = 0.1
dec_dropout = 0.1
```

```
# Hyper-parameter GRU
LEARNING_RATE = 0.0003
INPUT_DIM = len(SRC.vocab)
OUTPUT_DIM = len(TRG.vocab)
ENC_EMB_DIM = 256
DEC_EMB_DIM = 256
ENC_HID_DIM = 512
DEC_HID_DIM = 512
ENC_DROPOUT = 0.1
DEC_DROPOUT = 0.1
```

Appendix-Figure: VII Snapshot of fragment code of all proposed models hyper-parameter

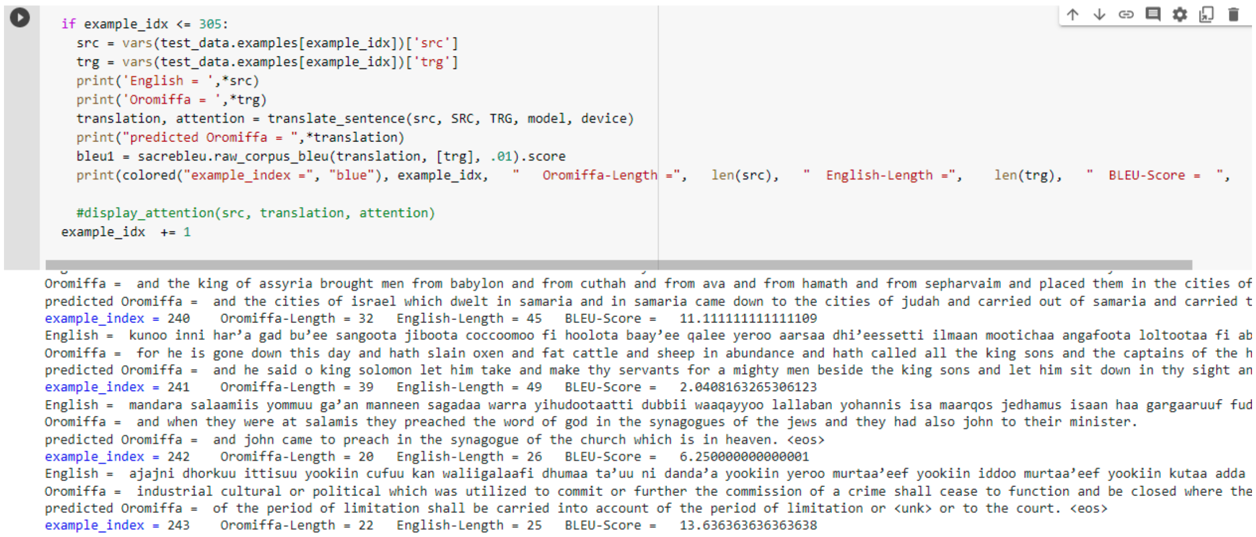
Appendix H: Sample code of define function for count models parameter.



```
def count_parameters(model):
    return sum(p.numel() for p in model.parameters() if p.requires_grad)

print(f'The model has {count_parameters(model):,} trainable parameters')
```

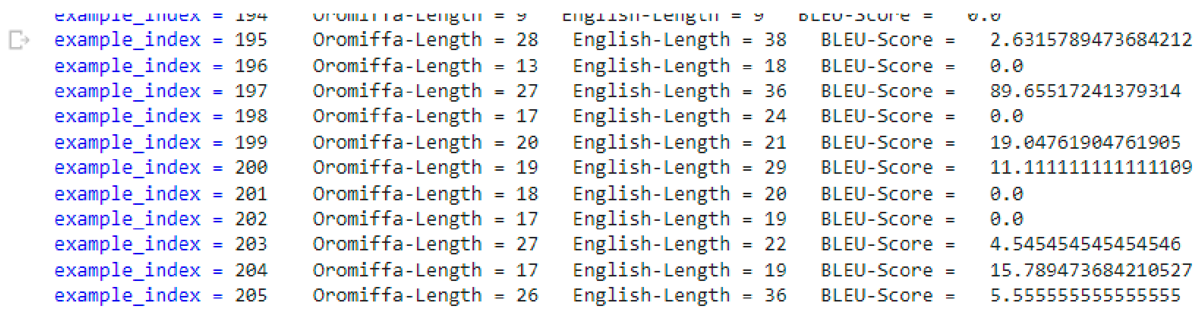
Appendix-Figure: VIII Snapshot of sample code of define function for count models parameter

Appendix I: Code for show source and target sentences length and their translated BLUE score in order to compare the effect of long sentences



```
if example_idx <= 305:
  src = vars(test_data.examples[example_idx])['src']
  trg = vars(test_data.examples[example_idx])['trg']
  print('English = ',*src)
  print('Oromiffa = ',*trg)
  translation, attention = translate_sentence(src, SRC, TRG, model, device)
  print("predicted Oromiffa = ",*translation)
  bleu1 = sacrebleu.raw_corpus_bleu(translation, [trg], .01).score
  print(colored("example_index =", "blue"), example_idx,   "   Oromiffa-Length =",   len(src),   "  English-Length =",   len(trg),   "  BLEU-Score =  ",

  #display_attention(src, translation, attention)
example_idx  += 1
```

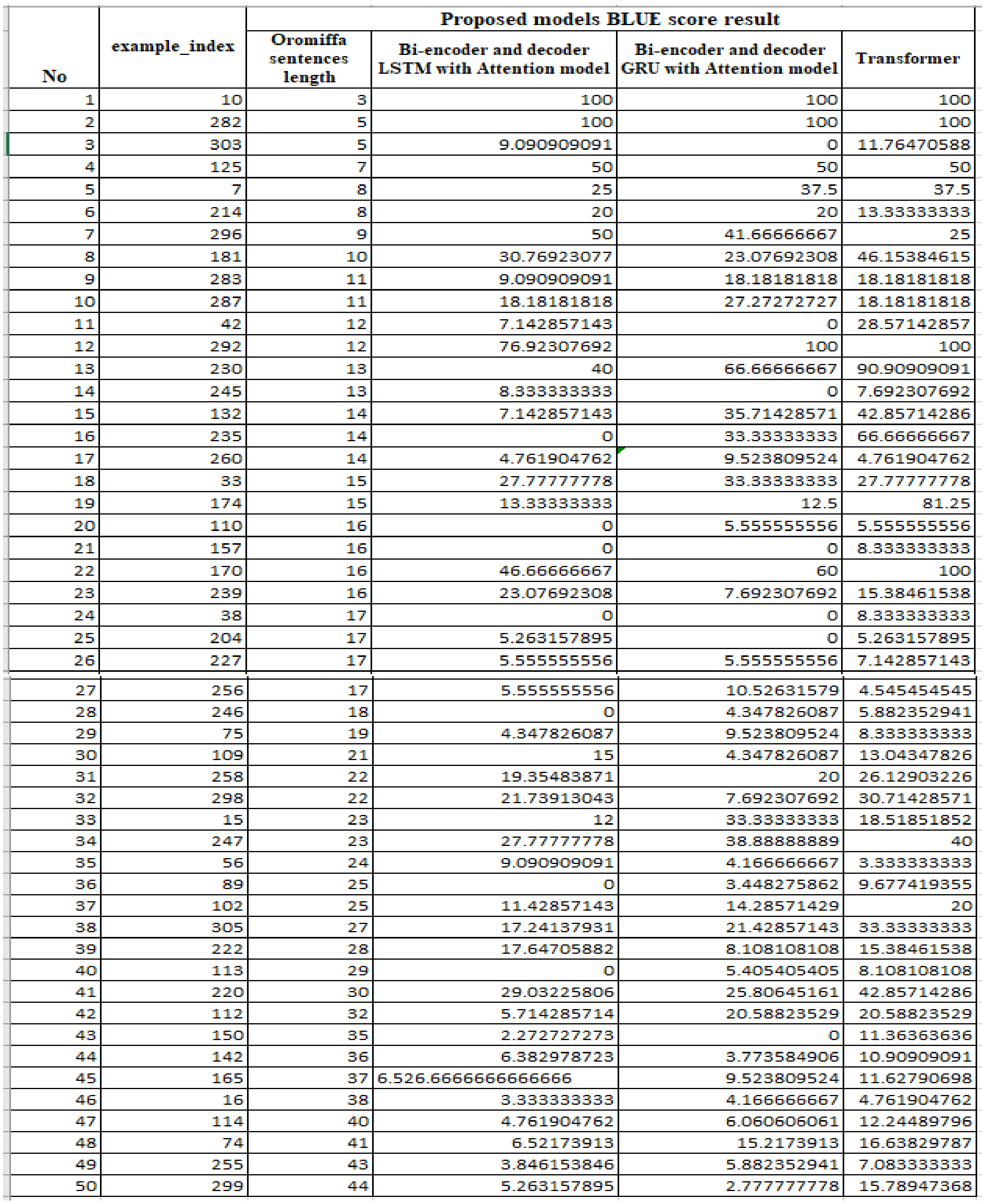
```
Oromiffa =  and the king of assyria brought men from babylon and from cuthah and from ava and from hamath and from sepharvaim and placed them in the cities of
predicted Oromiffa =  and the cities of israel which dwelt in samaria and in samaria came down to the cities of judah and carried out of samaria and carried t
example_index = 240    Oromiffa-Length = 32    English-Length = 45    BLEU-Score =    11.111111111111109
English =  kunoo inni har’a gad bu’ee sangoota jiboota coccoomoo fi hoolota baay’ee qalee yeroo aarsaa dhi’eessetti ilmaan mootichaa angafoota loltootaa fi ab
Oromiffa =  for he is gone down this day and hath slain oxen and fat cattle and sheep in abundance and hath called all the king sons and the captains of the h
predicted Oromiffa =  and he said o king solomon let him take and make thy servants for a mighty men beside the king sons and let him sit down in thy sight an
example_index = 241    Oromiffa-Length = 39    English-Length = 49    BLEU-Score =    2.0408163265306123
English =  mandara salaamiis yommuu ga’an manneen sagadaa warra yihudootaatti dubbii waaqayyoo lallaban yohannis isa maarqos jedhamus isaan haa gargaaruuf fud
Oromiffa =  and when they were at salamis they preached the word of god in the synagogues of the jews and they had also john to their minister.
predicted Oromiffa =  and john came to preach in the synagogue of the church which is in heaven. <eos>
example_index = 242    Oromiffa-Length = 20    English-Length = 26    BLEU-Score =    6.250000000000001
English =  ajajni dhorkuu ittisuu yookiin cufuu kan waliigalaafi dhumaa ta’uu ni danda’a yookiin yeroo murtaa’eef yookiin iddoo murtaa’eef yookiin kutaa adda
Oromiffa =  industrial cultural or political which was utilized to commit or further the commission of a crime shall cease to function and be closed where the
predicted Oromiffa =  of the period of limitation shall be carried into account of the period of limitation or <unk> or to the court. <eos>
example_index = 243    Oromiffa-Length = 22    English-Length = 25    BLEU-Score =    13.636363636363638
```



```
example_index = 194    Oromiffa-Length = 9    English-Length = 9    BLEU-Score =    0.0
example_index = 195    Oromiffa-Length = 28   English-Length = 38   BLEU-Score =    2.6315789473684212
example_index = 196    Oromiffa-Length = 13   English-Length = 18   BLEU-Score =    0.0
example_index = 197    Oromiffa-Length = 27   English-Length = 36   BLEU-Score =    89.65517241379314
example_index = 198    Oromiffa-Length = 17   English-Length = 24   BLEU-Score =    0.0
example_index = 199    Oromiffa-Length = 20   English-Length = 21   BLEU-Score =    19.04761904761905
example_index = 200    Oromiffa-Length = 19   English-Length = 29   BLEU-Score =    11.111111111111109
example_index = 201    Oromiffa-Length = 18   English-Length = 20   BLEU-Score =    0.0
example_index = 202    Oromiffa-Length = 17   English-Length = 19   BLEU-Score =    0.0
example_index = 203    Oromiffa-Length = 27   English-Length = 22   BLEU-Score =    4.545454545454546
example_index = 204    Oromiffa-Length = 17   English-Length = 19   BLEU-Score =    15.789473684210527
example_index = 205    Oromiffa-Length = 26   English-Length = 36   BLEU-Score =    5.555555555555555
```

Appendix-Figure: IX Snapshot of fragment code of show source and target sentences length and their translated BLUE score.

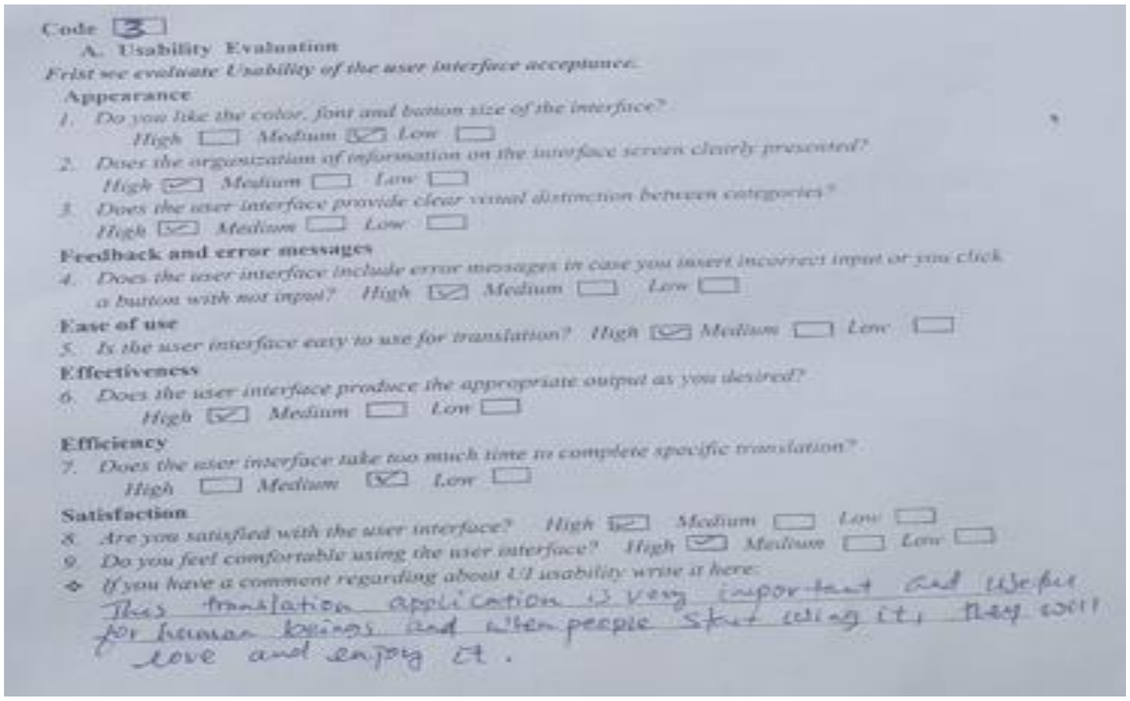
Appendix J: A sample table image that show the Afaan Oromo sentence length with BLUE score result from all proposed



No	example_index	Oromiffa sentences length	Bi-encoder and decoder LSTM with Attention model	Bi-encoder and decoder GRU with Attention model	Transformer
			Proposed models BLUE score result		
1	10	3	100	100	100
2	282	5	100	100	100
3	303	5	9.090909091	0	11.76470588
4	125	7	50	50	50
5	7	8	25	37.5	37.5
6	214	8	20	20	13.33333333
7	296	9	50	41.66666667	25
8	181	10	30.76923077	23.07692308	46.15384615
9	283	11	9.090909091	18.18181818	18.18181818
10	287	11	18.18181818	27.27272727	18.18181818
11	42	12	7.142857143	0	28.57142857
12	292	12	76.92307692	100	100
13	230	13	40	66.66666667	90.90909091
14	245	13	8.333333333	0	7.692307692
15	132	14	7.142857143	35.71428571	42.85714286
16	235	14	0	33.33333333	66.66666667
17	260	14	4.761904762	9.523809524	4.761904762
18	33	15	27.77777778	33.33333333	27.77777778
19	174	15	13.33333333	12.5	81.25
20	110	16	0	5.555555556	5.555555556
21	157	16	0	0	8.333333333
22	170	16	46.66666667	60	100
23	239	16	23.07692308	7.692307692	15.38461538
24	38	17	0	0	8.333333333
25	204	17	5.263157895	0	5.263157895
26	227	17	5.555555556	5.555555556	7.142857143
27	256	17	5.555555556	10.52631579	4.545454545
28	246	18	0	4.347826087	5.882352941
29	75	19	4.347826087	9.523809524	8.333333333
30	109	21	15	4.347826087	13.04347826
31	258	22	19.35483871	20	26.12903226
32	298	22	21.73913043	7.692307692	30.71428571
33	15	23	12	33.33333333	18.51851852
34	247	23	27.77777778	38.88888889	40
35	56	24	9.090909091	4.166666667	3.333333333
36	89	25	0	3.448275862	9.677419355
37	102	25	11.42857143	14.28571429	20
38	305	27	17.24137931	21.42857143	33.33333333
39	222	28	17.64705882	8.108108108	15.38461538
40	113	29	0	5.405405405	8.108108108
41	220	30	29.03225806	25.80645161	42.85714286
42	112	32	5.714285714	20.58823529	20.58823529
43	150	35	2.272727273	0	11.36363636
44	142	36	6.382978723	3.773584906	10.90909091
45	165	37	6.526.666666666666	9.523809524	11.62790698
46	16	38	3.333333333	4.166666667	4.761904762
47	114	40	4.761904762	6.060606061	12.24489796
48	74	41	6.52173913	15.2173913	16.63829787
49	255	43	3.846153846	5.882352941	7.083333333
50	299	44	5.263157895	2.777777778	15.78947368

Appendix-Figure: X A sample Afaan Oromo sentence length with BLUE score result from all proposed.

Appendix K: Manual prototype UI usability Evaluation form



Code 3

A. Usability Evaluation

Frist we evaluate Usability of the user interface acceptance.

Appearance

1. *Do you like the color, font and button size of the interface?*
   High ☐ *Medium* ☑ *Low* ☐
2. *Does the organization of information on the interface screen clearly presented?*
   High ☑ *Medium* ☐ *Low* ☐
3. *Does the user interface provide clear visual distinction between categories?*
   High ☑ *Medium* ☐ *Low* ☐

Feedback and error messages

4. *Does the user interface include error messages in case you insert incorrect input or you click a button with not input?* *High* ☑ *Medium* ☐ *Low* ☐

Ease of use

5. *Is the user interface easy to use for translation?* *High* ☑ *Medium* ☐ *Low* ☐

Effectiveness

6. *Does the user interface produce the appropriate output as you desired?*
   High ☑ *Medium* ☐ *Low* ☐

Efficiency

7. *Does the user interface take too much time to complete specific translation?*
   High ☐ *Medium* ☑ *Low* ☐

Satisfaction

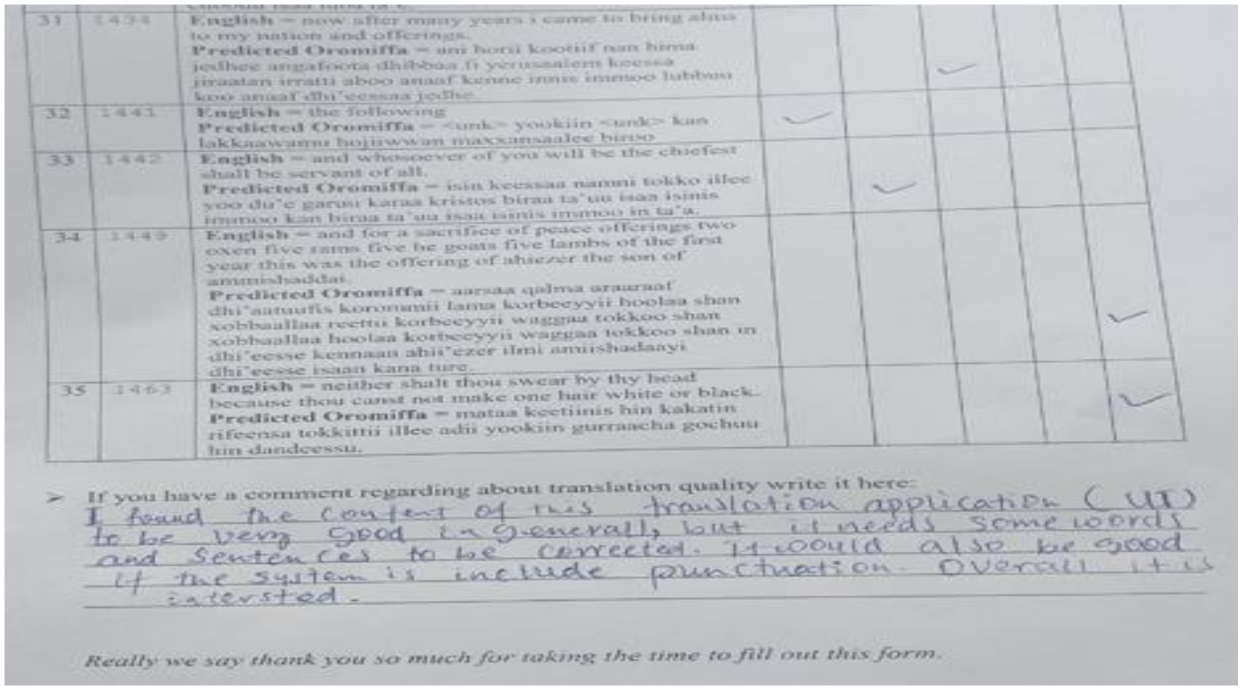
8. *Are you satisfied with the user interface?* *High* ☑ *Medium* ☐ *Low* ☐
9. *Do you feel comfortable using the user interface?* *High* ☑ *Medium* ☐ *Low* ☐

❖ *If you have a comment regarding about UI usability write it here:*

This translation application is very important and usefull
for human beings and when people start using it, they will
love and enjoy it.

Appendix-Figure: XI A sample photo of the manual prototype UI usability evaluation from

Appendix L: Manual Evaluation form for translated sentences



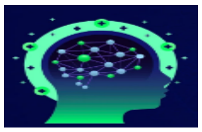
31	1434	**English** – now after many years i came to bring alms to my nation and offerings. **Predicted Oromiffa** – ani bara kootiif nan hima jedhee angafoota dhibbaa fi yerusaalem keessa jiraatan irratti aboo anaaf kenne innis immoo lubbuu koo anaaf dhi'eessaa jedhe.			✓		
32	1441	**English** – the following **Predicted Oromiffa** – <unk> yookiin <unk> kan lakkaawamu hojiiwwan maxxansaalee biroo	✓				
33	1442	**English** – and whosoever of you will be the chiefest shall be servant of all. **Predicted Oromiffa** – isin keessaa namni tokko illee yoo du'e garuu karaa kristos biraa ta'uu isaa isinis immoo kan biraa ta'uu isaa isinis immoo in ta'a.		✓			
34	1449	**English** – and for a sacrifice of peace offerings two oxen five rams five he goats five lambs of the first year this was the offering of ahiezer the son of ammishaddai. **Predicted Oromiffa** – aarsaa qalma araaraaf dhi'aatuufis korommii lama korbeeyyii hoolaa shan xobbaallaa reettu korbeeyyii waggaa tokkoo shan xobbaallaa hoolaa korbeeyyii waggaa tokkoo shan in dhi'eesse kennaan ahii'ezer ilmi amiishadaayi dhi'eesse isaan kana ture.					✓
35	1463	**English** – neither shalt thou swear by thy head because thou canst not make one hair white or black. **Predicted Oromiffa** – mataa keetiinis hin kakatin rifeensa tokkittii illee adii yookiin gurraacha gochuu hin dandeessu.					✓

➢ If you have a comment regarding about translation quality write it here:

I found the content of this translation application (UI)
to be very good in generall, but it needs some words
and sentences to be corrected. It would also be good
if the system is include punctuation. Overall it is
interested.

Really we say thank you so much for taking the time to fill out this form.

Appendix-Figure: XII A sample photo of the manual translation quality evaluation from.



Bi-Directional English to Afaan Oromo Neural Machine Translation using a Deep Learning Approach

Waldeyes Adere Gari

Declaration

I hereby declare that this thesis represents my own work, which has been done after registration for the degree of Masters at Debre Berhan University and has not been previously submitted for any other degree or professional qualification. I confirm that the work of this thesis is my own, and all sources of materials used for the thesis have been duly acknowledged.

Name: Waldeyes Adere Gari

Signature: _______________

Date of submission: _______________

Place: Debre Berhan University, Debre Berhan, Ethiopia

Name	Signature	Date
Gaddisa Olani Ganfure (PhD)	[illegible]	05/09/2022