



**Attention-based Neural Machine Translation from English-  
Wolaytta**

**A Thesis Presented**

**by**

**Mekdes Melese**

**to**

**The Faculty of Informatics**

**of**

**St. Mary's University**

**In Partial Fulfillment of the  
Requirements for the Degree of Master  
of Science**

**in**

**Computer Science**

**January 2023**

# ACCEPTANCE

**Attention-based Neural Machine Translation from English-Wolaytta**

**By**

**Mekdes Melese**

**Accepted by the Faculty of Informatics, St. Mary's University, in partial  
fulfilment of the requirements for the degree of Master of Science in  
Computer Science**

**Thesis Examination Committee:**

---

**Internal Examiner**

**{Full Name, Signature and Date}**

**Dr. Minale Ashagrie**



**Feb, 23,2023**

**External Examiner**

**{Full Name, Signature and Date}**

---

**Dean, Faculty of Informatics**

**{Full Name, Signature and Date}**

**January 2023**

## DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been duly acknowledged.

Mekdes Melese Mekuria

Full Name of Student

---

Signature

Addis Ababa

Ethiopia

This thesis has been submitted for examination with my approval as advisor.

Alemebante Mulu Kumlign (PhD)

Full Name of Advisor

Addis Ababa

Ethiopia

January 2023

## **ACKNOWLEDGMENTS**

First and foremost, praise be to the Almighty God who gave me the opportunity, strength, and wisdom to achieve whatever I have achieved so far.

Next, I am greatly indebted to a number of people who assisted me to successfully complete this thesis. My most profound thanks go to my advisor Alemebante Mulu (PhD) for his support and guidance during the running of this thesis.

I would also like to thank my friends and classmates who take part in this research for their collaborative effort during data collection. I want to offer my profound gratitude to everyone whose assistance made this job possible.

The completion of this thesis would not have been possible without the guidance and support of my family. I would like to thank my parents; whose love and guidance are with me in whatever I pursue.

## TABLE OF CONTENTS

ACKNOWLEDGMENTS .....	i
TABLE OF CONTENTS.....	ii
LIST OF ACRONYMS .....	vi
LIST OF FIGURES .....	vii
LIST OF TABLES .....	viii
ABSTRACT.....	ix
CHAPTER ONE.....	1
INTRODUCTION .....	1
1.1. Background of the Study .....	1
1.2. Statement of the Problem.....	2
1.3 Research Motivation .....	4
1.4. Research Questions.....	5
1.5. Objectives of the Study .....	5
1.5.1. General Objective .....	5
1.5.2. Specific Objectives .....	6
1.6. Scope and Limitations of the Study .....	6
1.7. Significance of the Research.....	6
1.8. Research Methododlogy .....	7
1.8.1. Literature Review.....	7
1.8.2. Data Collection .....	7
1.8.3. Tools .....	7
1.9. Thesis Organization .....	8

CHAPTER TWO .....	9
LITERATURE REVIEW .....	9
2.1. Introduction.....	9
2.2. Why Machine Translation?.....	9
2.3. An Overview of Machine Translation .....	10
2.4. Approaches of Machine Translation.....	12
2.4.1. Rule Based Machine Translation (RBMT).....	12
2.4.2. Corpus-based Machine Translation .....	16
2.4.3. Hybrid Machine Translation Approach .....	20
2.4.4. Neural Machine Translation Approach.....	21
2.5. Encoder-Decoder Architecture .....	23
2.6. Attention .....	25
2.7. Language Modelling .....	25
2.8. Neural Language Model (NLM).....	26
2.9. Network Models in Neural Machine Translation .....	27
2.9.1. Recurrent Neural Network.....	29
2.9.2. Long Short-Term Memory.....	30
2.10. Related Work on Machine Translation.....	33
2.10.1. Machine Translation Involving Wolaytta Language .....	33
2.10.2. Machine Translation Involving other Ethiopian Languages.....	35
2.10.3. Machine Translation Involving Non- Ethiopian (Foreign) Languages.....	39
CHAPTER THREE .....	42
THE WOLAYTTA LANGUAGE.....	42
3.1. Introduction.....	42
3.2. Overview of Wolaytta Language .....	42

3.3.	Morphology of Wolaytta Language.....	43
3.3.1.	Inflectional Morphology and Derivational Morphology .....	45
3.3.2.	Derivational Morphology in Wolaytta.....	50
3.4.	Word Formation in Wolaytta.....	52
3.5.	Wolaytta Language Writing System.....	53
3.6.	Punctuation Marks in Wolaytta Language.....	54
3.7.	Wolaytta Language Sentence Structure .....	54
CHAPTER FOUR.....		55
PROPOSED ARCHITECTURE AND RESEARCH METHODOLOGY .....		55
4.1.	Introduction.....	55
4.2.	Proposed Architecture of English-Wolaytta NMT .....	55
4.3.	Corpus Collection and Preparation .....	57
4.4.	Text Preprocessing.....	58
4.4.1.	Normalization .....	58
4.4.2.	Data Cleaning.....	59
4.5.	Tokenization and Padding.....	59
4.6.	Word Representation .....	60
4.7.	Data Splitting for Model Training and Testing.....	60
4.8.	Encoder .....	61
4.9.	Decoder .....	62
4.10.	Attention Mechanism.....	62
4.11.	Model Evaluation Metrics.....	63
CHAPTER FIVE .....		64
EXPERIMENTAL RESULT AND DISCUSSION.....		64
5.1.	Introduction.....	64

5.2.	English-Wolaytta NMT Model Building and Training .....	64
5.2.1.	English-Wolaytta NMT using Non-attention Mechanism.....	64
5.2.2.	English-Wolaytta NMT using Attention Mechanism.....	65
5.3.	English-Wolaytta NMT Testing and Translation .....	66
5.4.	Discussion of the Result.....	67
CHAPTER SIX.....		69
CONCLUSION AND FUTURE WORKS .....		69
6.1.	Introduction.....	69
6.2.	Conclusion .....	69
6.3.	Future Works .....	70
References.....		71
Appendix.....		78
Appendix A:.....		78
Appendix B:.....		79



## LIST OF ACRONYMS

AI	Artificial Intelligence
BLEU	Bilingual Evaluation Understudy
CNN	Convolutional Neural Networks
EBMT	Examples-Based Machine Translation
GNMT	Google's Neural Machine Translation System
GRU	Gated Recurrent Unit
HMM	Hidden Markov Models
HMT	Hybrid Machine Translation
LM	Language Modeling
LSTM	Long Short-Term Memory
ML	Machine Learning
MT	Machine Translation
NL	Natural Language
NLM	Neural Language Model
NLP	Natural Language Processing
NMT	Neural Machine Translation
RBMT	Rule Based Machine Translation
RNN	Recurrent Neural Networks
Seq2seq	Sequence to Sequence
SMT	Statistical Machine Translation
TPUs	Tensor Processing Unit

## LIST OF FIGURES

Figure 2.1: General Steps of Rule-based Machine Translation .....	13
Figure 2.2: Major Tasks in Direct Machine Translation Approach.....	14
Figure 2.3: Machine Translation Pyramid .....	15
Figure 2.4: Statistical Machine Translation .....	19
Figure 2.5: The encoder-decoder architecture for NMT.....	24
Figure 2.6: Architecture of RNN .....	29
Figure 2.7: Gated Recurrent Units (GRU).....	31
Figure 2.8: Long Short-Term Memory (LSTM).....	32
Figure 3.1: Word Morphemes.....	44
Figure 4.1: Proposed Architecture for English-Wolaytta NMT .....	56
Figure 4.2: Sample English-Wolaytta Parallel Corpus .....	58
Figure 4.3: Sample word2index Representation.....	60
Figure 4.4: LSTM Encoder .....	61
Figure 5. 1 : English-Wolaytta NMT Translation Result .....	66

## LIST OF TABLES

Table 1.1: Hardware Tool.....	8
Table 3.1: Suffix Formation of Wolaytta Language.....	45
Table 3.2: List of Pronouns.....	46
Table 3.3: General Syllable Formulation of a Bi-radical Wolaytta Nouns.....	47
Table 3.4: Pluri-radical Wolaytta Nouns Form.....	47
Table 3.5: Case Markers .....	49
Table 3.6: List of Wolaytta Adjective with their Ending.....	49
Table 3.7: Derivational Morphology Nouns.....	51
Table 3.8: List of conjunctions .....	52
Table 3.9: Wolaytta Language Alphabet Letters .....	53
Table 5. 1 : Best Results using Encode-Decoder without Attention Mechanism.....	65
Table 5. 2 : Best Results using Encode-Decoder with Attention Mechanism.....	65

## ABSTRACT

Machine translation (MT) is one of the applications of natural language processing which involves using computers to translate from one source language to another target language. For many years, Statistical Machine Translation (SMT) dominated the field of machine translation technology. Long sentences are broken up into small pieces in classical statistical machine translation, which results in poor levels of accuracy. Neural Machine Translation (NMT) is a new paradigm that swiftly superseded SMT as the predominant method of MT, developed with the development of deep learning. NMT approach differs from SMT systems as all parts of the neural translation model are trained jointly (end-to-end) to maximize the translation performance. In an encoder-decoder design, the entire source sequence's input is condensed into a single context vector, that is then sent to the decoder to create the output sequence. The major drawback of encoder-decoder model is that it can only work on short sequences. It is difficult for the encoder model to memorize long sequences and convert it into a fixed-length vector. One realistic solution to this problem is the attention mechanism. The attention mechanism predicts the next word by concentrating on a few relevant parts of the sequence rather than looking on the entire sequence. Hence, the objective of this research work is to develop a neural machine translation system for English-Wolaytta using attention mechanism.

The English-Wolaytta machine translation system has been trained on parallel corpus covering the religious, and frequently used sentences or phrases which can be used in day to day communication. A total of 27351 parallel English-Wolaytta sentences were prepared and the system is trained and tested using 80/20 ratio. These data were preprocessed in the suitable format in way to be used in neural machine translation. For building the proposed English-Wolaytta NMT model, an LSTM encoder and decoder architecture with an attention mechanism has been proposed in the Sequence-to-Sequence concept. In order to evaluate the efficiency of the proposed system, BLUE score metrics is used, and for testing the efficiency of attention mechanism, we have developed non-attention model and compared it with the attention mechanism. Hence, we have proved that the attention mechanism has a better translation and has achieved a BLEU score of 5.16 and 88.65 accuracy.

**Keywords:** *Machine Translation, Neural Machine Translation, English, Wolaytta, Attention Mechanism, Encoder-Decoder Architecture, Natural Language Processing*

# CHAPTER ONE

## INTRODUCTION

### 1.1. Background of the Study

Communication is the main tie that binds our world community. Language is the primary means of communication among humans. Language also called Natural Language (NL) which refers to any human spoken or written symbol that has evolved naturally for human communication. Through natural language communication, we are able to share our ideas, opinions, views, and emotions with another person. The purpose of language is creating an understanding of complex and abstract thinking. It plays a vital role in helping people build a bridge of relationships. Various natural language is used by people residing in different areas or belonging to different communities.

To make communication possible in a multilingual environment, either people need to use a common language when conveying a message or they need to adapt the source language and culture to those of the people they want to communicate with. But language barriers are creating huge gaps. Globalization and the rise of the internet as a global medium of communication has led to an ever-increasing demand for translation systems. Translation systems have a significant role in bridging both the linguistic and cultural differences that has long-standing between people of different corner of the world. Recently, advances in technology have paved a way to positive changes in translation making possible Interlingua communications. Natural Language Processing (NLP) is one of these advancements.

NLP, or computational linguistics, is a fundamental area of machine learning (ML). It is the capacity of a computer software to comprehend, interpret, and work with spoken and written human language [1]. NLP contributes significantly to the survival and further development of languages by offering state-of-the-art tools and applications to the speakers. It is a considerable step forward in Artificial Intelligence (AI) [2]. NLP typically involves applications of Computer Science and computational linguistics in its efforts to fill the gap between human communication and computer understanding [1]. NLP has various applications including Machine Translation (MT), Text summarization, Information extraction etc. [3].

Human-human, human-computer, computer-human, and computer-computer communication via computing systems can be facilitated by NLP applications. Every language should be easily comprehended by the computer in the NLP environment. Machine translation is the procedure that enables a computer to comprehend the various languages spoken worldwide (MT).

Machine translation is a branch of computational linguistics which studies the use of computer software to translate text or speech from one natural language to another [4]. It can be defined as the capability of computers to automatically translate text between two natural languages while maintaining the intended meaning and producing fluid text in the target language. Without the use of human translators, these automatic translation systems transform one language into another using cutting-edge technology, extensive dictionaries, and a set of linguistic principles [5]. Despite being one of the oldest areas of study in artificial intelligence, machine translation has seen very significant advancements in terms of translation quality due to a growing interest in human-machine interactions, the availability of big data, improved algorithms, and a recent shift toward large-scale empirical techniques. The demand for translation services is currently expanding across numerous industries, including business, medicine, and the economy.

Machine translations can be classified according to their core methodology: the rule based approach also known as Knowledge based approach which is a linguistic rich approach where humans specify a set of rules to describe the translation process and the corpus based approach which is entirely corpus based where knowledge is extracted from a parallel corpus.

Rule based machine translation approach involves sub approaches like direct approach, transfer based approach and Interlingua machine translation approach. Sub approaches involved in corpus-based machine translation include statistical phrase-based approach and neural-based approaches. Example-based, Knowledge based approaches are termed as is less used approaches nowadays [3].

## **1.2. Statement of the Problem**

Ethiopia is a multilingual, multiethnic and culturally a pluralistic nation with more than 80 different languages with over 200 dialects spoken. Languages in Ethiopia can be classified within four major language groups, though the country is also home to several unclassified tongues. The four main language groups in Ethiopia are Semitic, Cushitic, Omotic and Nilo-Saharan. Most of

the languages in Ethiopia are included in the Afro-Asia language family. Of these Ethiopian languages' majority are Cushitic and Omotic languages and some are Semitic languages [6]. Nearly 30 languages collectively referred to as "Omotic languages" and they are spoken in the south-west of Ethiopia, close to the Omo River. The 28 Omotic languages are divided into southern and northern sub-families among these [7].

The Wolaytta language, which is spoken in the Wolaytta Zone and some other areas of Ethiopia's Southern Nations, Nationalities, and People's Region, is a member of the Northern Omotic language family. The number of speakers of this language is estimated about 15.5 million Populations (based on 1999E.C census) [8]); it is the native language of the Wolaytta people. It is also spoken in different neighboring areas and various cities throughout the country by people from Wolaytta region. In the region, the media of instruction in primary schools is Wolaytta language and is offered as a program in Wolaytta Sodo University. The number of articles and newspapers published in this language is increasing over the years and different Mass Medias are streaming their programs this language [6]. The need for effective translation has become a matter of urgency due to the ever-increasing amount of contents that are being created in English language. Various research results, teaching materials and information available on the Internet use these languages as their preferred language of communication.

Language constraints might still make it difficult to acquire information in the contemporary globalized environment. In some cases, it is impossible to meet the demand for translation by using solely human translators; as a result, tools like MT are becoming more and more popular since they have the ability to solve this issue. The importance of using technology in people's daily lives is also increasing. Not only in educational areas but also in social, financial, technological, entertainments and cultural fields. As the necessity to use technology is increasing, so does the demand for translation.

Despite the large number of speakers all over the country, there are very few computational natural language tools available for Wolaytta. It is a morphologically rich language having many other distinct linguistic characteristics. On the contrary it is still an under-resourced language. Just to mention a few of the works done so far: Attention Based Amharic-to- Wolaytta Neural Machine Translation by Workineh Wogasso [9], English- Wolaytta machine translation using Statistical Approach by Melaku Mara [10], Bidirectional Dictionary Based machine translation for Wolaytta

Amharic by Temesgen Mengistu [11], a Hybrid Machine Translation System for English-to-Wolaytta by Kidanemariam Firew [12]. This previous works uses traditional translational approaches. Over the course of machine translation development, MT has changed greatly, from systems that required hours and days of computing time to produce a translation of dubious quality, to the current neural machine translation (NMT) systems that can process the same content in mere seconds and with much more accuracy. Neural machine translation has many advantages over traditional machine translation including higher translation accuracy, less need for human input, quicker translation turnaround times and so on. However, to the researcher's knowledge the neural translation using attention-based model for English to Wolaytta has not yet been done.

This study uses neural machine translation (NMT) with attention approach to translate words from English to Wolaytta. The proposed model is chosen because attention-based neural machine translation has achieved significant performance in recent years and attentional neural machine translation is efficient and produces fluent translation. It is becoming the mainstream machine translation method in the current industry. For example, NMT with attention is the leading model behind the popular services like google translate. In 2016 google announce the launch of Google Neural Machine Translation system (GNMT), which utilizes state-of-the-art training techniques to achieve the largest improvements to date for machine translation quality. Human evaluations show that GNMT has reduced translation errors by 60% compared to the previous phrase-based system on many pairs of languages: English ↔ French, English ↔ Spanish, and English ↔ Chinese [13].

During translation, the attention mechanisms selectively focuses on sub-parts of the sentence to improve the performance of neural machine translations. To accomplish an efficient translation with respect to accuracy as well as quality and to reach a better BLEU score between the language pairs, the attention-based neural machine translation is proposed.

### **1.3. Research Motivation**

The primary objective of MT is to eliminate language barriers by developing a machine translation system that can translate one human language to another. It plays a vital role in strengthening communications between people residing in different areas and in enabling peoples to use documents and data produced in resource rich languages. There is a high need for translation due



to the requirement for information sharing among resource-rich and low resource languages. Languages with a limited amount of parallel corpus resources benefit less from such a system than languages with large translated resources. Among the languages with less translated documents are Ethiopian languages such as Wolaytta.

Despite being widely spoken language in southern Ethiopia, Wolaytta is considered as less resourced language. English is the universal language on the Internet and many documents are written in English. Because of that non-English speakers are faced with the problem of communication and limited access to resources and this problem is compounded in Wolaytta language. Because it falls under the category of a resource-rich language, English is the ideal choice to translate material from. Improving machine translation accuracy from resource rich language like English to resource scarce languages like Wolaytta will make a significant contribution. With the recent widespread use of Wolaytta language on publications and on social media in Wolaytta Zone, improving the performance of translation from English language is of great benefit to both the community, the private and public sector in the area. Since Wolaytta is the official working language of Wolaytta zone, applying machine translation on the translation of different educational books or other materials can contribute to different government institutions like elementary education. English- Wolaytta Machine Translation can solve the aforementioned problems. This has motivated us to work on Attention-based English-to- Wolaytta Neural MT.

#### **1.4. Research Questions**

The central questions of the study which is addressed by this research study are:

- What will the English-Wolaytta language pairs' performance of the attention-based NMT approach be?
- How well do attention-based NMT models perform in translation tasks?

#### **1.5. Objectives of the Study**

##### **1.5.1 General Objective**

The general objective of this study is to develop an English-to-Wolaytta neural machine translation using attention-based approach.

### **1.5.2 Specific Objectives**

The following specific objectives are outlined in order to fulfill the study's general objective.

- To prepare English- Wolaytta parallel corpus
- To review techniques and methodologies used for machine translation
- To review related literatures to identify the linguistic behaviors of English and Wolaytta languages.
- To design the architecture of the system
- To develop a model for English-Wolaytta machine translation
- To evaluate the performance of the developed model

### **1.6. Scope and Limitations of the Study**

The main aim of this study is to develop an attention based neural machine translation for English to Wolaytta. The study designed to operate only in one direction (unidirectional) i.e., from English to Wolaytta. Because of having insufficient sources, the data is collected manually from limited resources which makes the system limited to specific domains. The other limitation of this study is a lack of computationally powerful machine that helps us to make different experiments to get a more efficient model by changing different parameters setups and using different approaches.

### **1.7. Significance of the Study**

The main contribution of this research work includes:

- After this study, it is expected that there will be better translation between the two language pairs.
- This study can be used in the preparation of teaching materials in the areas in which Wolaytta is spoken.
- The study will contribute to increase digital literacy by removing language barrier
- It will help with the development of teaching materials in the Wolaytta region and the accurate translation of technical papers.

## **1.8. Methodology of the study**

To accomplish the research a lot of methods, tools, and techniques were applied.

### **1.8.1 Literature Review**

Reviewing several literatures on machine translation systems were conducted for other language pairs to gain deep understanding of the research area. A detailed literature review on Neural Machine translation approach in particular with regard to techniques used in the approach were done. For further understanding of the linguistic behavior of both language pairs, different related articles and books were reviewed.

### **1.8.2 Data Collection**

The translation system tries to generate translations using the English-Wolaytta corpus based on neural methods. To perform the experiment, English-Wolaytta parallel corpus were collected from sources like the Holy Bible and simple sentences commonly used for daily communication purpose. The simple sentences were prepared manually to conduct the experiment. Since Wolaytta is under-resourced language, the corpora are from limited sources.

### **1.8.3 Tools**

To implement the model of the English- Wolaytta neural machine translation system, Python programming language is used. Python is chosen because of its high-recital tool for NLP and it is open-source, with a variety of rich libraries, rich documentation, and support. Anaconda Navigator which is a desktop graphical user interface (GUI) is used to help us use different very efficient editors like Jupyter Notebook. The NMT was built using TensorFlow, one of the most popular data science and machine learning frameworks, which is an open-source deep learning library with Keras and NumPy library. Keras library was employed in this work on top of TensorFlow.

Table 1.1: Hardware Tool

Computer Type	Operating System	Processor	System Type	Installed RAM	Storage
Laptop, HP ENVY x360	Windows 11 Home	AMD Ryzen5 4500U with Radeon Graphics 2.38 GHz	64-bit Operating system x64-based processor	8 GB	238.46 GB SSD

### 1.9. Thesis Organization

This thesis report includes six chapters. Chapter two presents the literature review of Machine Translation, Types of Machine Translation and their advantage and disadvantage, Neural Machine Translation and the different architecture of NMT and algorithms. It also presents a review of works done in the area of Machine Translation in Ethiopian and Foreign languages.

Chapter Three presents the overview of Wolaytta Language. The chapter represents the inflectional and derivational morphology of the language.

Chapter Four discusses the steps that have been followed in the research work and the proposed architecture of the of English-Wolaytta NMT model.

Chapter Five discusses the experiment and implementation of English-Wolaytta NMT model by applying data processing and Encoder-Decoder with attention mechanism, models training, and testing. It also presents the experiment result, evaluation and comparison of performance of the model based on evaluation metrics.

Chapter Six presents the conclusion and future works for further research direction on this research topic/domain.

## **CHAPTER TWO**

### **LITERATURE REVIEW**

#### **2.1. Introduction**

In this chapter, review of literature in the field of machine translation has been made. The chapter covers overview of machine translation, the need for machine translation and different approaches of machine translation with special focus on neural machine translation approach. A review of related research works which were done in the area of Machine Translation in Ethiopian and Foreign languages is also provided.

#### **2.2. Why Machine Translation?**

In the context of modern globalization, people in various fields are now expanding their scope activities from local and regional to international levels. Globalization is playing a major role in this regard. It is combining social, cultural, economic, political and technological developments and is creating a unified overall environment. Globalization supported by information and communication technology has made the world a very small place. Keeping pace with the ever-increasing globalization is making the need for translation greater than ever. Translation fills the global communication gap between the communities speaking different languages. Difficulties in understanding foreign language have reduced significantly since MT has come to the field. MT has constantly been improved and upgraded through the enhanced programs of statistics, analysis and data processing. Such popular services as Babelfish and Google Translate are often used to meet the communication needs of the globalized world community, in particular on the Internet platform.

The Internet World Statistics Report describes that the content available on the internet in different languages varies, and the most dominant language on the internet is English [14] keeping in view this issue there is a dire need of machine translation system to make the web content available to everyone in their native language

Machine translation frameworks are expected to decode or translate creative works from any language to local language. Such machine translation frameworks can break the language obstruction by quickly making work accessible to the globe's masses. Numerous web pages may

contain information related to our interest in a foreign language, and with the help of machine translation, we understand the content present in those web pages. Machine translation can also help commercial product manufacturers prepare product manual in many languages that can be used by different countries. With the advancements in the internet, millions of users worldwide can get the information in their native language with the help of machine translation. In modern civilization, machine translations have growing need and importance in economics, business, health care and industrialization. The social and political urgency of machine translation rises in societies where more than one language is spoken. During the last decade, machine translation technology has improved. With the emergence of personal computers and the increasing use of computer-assisted tools, machine translation gained a strong momentum giving birth to commercially available software and hardware with translation tools and powerful dictionaries. Currently, machine translation is used by several companies and by governments. It is becoming more and more crucial for companies to remain relevant in the fast-changing global economy. It translates a vast amount of text in less time than a human translator, thus saving a lot of time.

### **2.3. An Overview of Machine Translation**

Without a human being involved, a computer program may translate text from one language to another using a process known as machine translation (MT), sometimes known as automated translation. It is a field of practical research that incorporates concepts and methods from statistics, artificial intelligence (AI), computer science, and linguistic programming [15].

The goal of machine translation is to create a system that can translate text from one language into another while maintaining the original text's meaning. Because natural languages are extremely complicated in terms of term word meaning, grammar rules, etc., MT requires a comprehensive understanding of both the source language and the target language, including both languages' grammar and semantic syntactic comprehension [16]. It begins the process by analyzing the input in the source language and building an internal representation. This representation is changed and converted into a format appropriate for the intended language. Finally, output is produced in the intended language. On a fundamental level, MT simply swaps out words from one natural language with words from another.

A translation may be done manually or automatically using machine translation. When using computer-based translation tools, human translators assist machine translators in performing the translation. When using human-aided machine translation, humans and computers work together to translate texts. Human involvement occurs either before the translation process, known as pre-editing, or after the translation process, known as post-editing. The distinction between machine aided and human aided machine translation is frequently ambiguous, and the term "computer-aided translation" might refer to either. However, the automation of the entire translation process is the essence of machine translation [4].

Systems for translating texts are either created for two specific languages (bilingual systems) or for multiple pairs of languages (multilingual systems). Bilingual systems can be created to work in both directions and only one direction (unidirectional), such as from English into Wolaytta language. When translation is unidirectional, just one direction from the source language to the destination language is supported. Systems that are bidirectional function in both directions, allowing one language to serve as the source and the other as the destination and vice versa. Most bilingual systems are unidirectional, although multilingual systems are typically meant to be bidirectional [4].

However, machine translation has advantages and disadvantages just like everything else. The speed of machine translation is its main benefit. It is accessible at all times, quick, affordable, and easily updatable. Utilizing it comes with certain trade-offs as well. It is unable to provide a complete and accurate translation on its own. Even so, it eventually needs assistance from people. Additionally, machine translation cannot understand the intricacies of culture and society or its substance. The main problem for the machine translation paradigm is developing a program that can comprehend text like humans do and produce new material in the target language that sounds like it was produced by a human [17].

Machines are not yet intelligent enough to distinguish a word meaning based on context, and may provide a bad translation. Also, the idioms, tone, cultural references upon which a language is built and impart any particular message; cannot be understood by any machines.

Humans are not exempt from difficulties. For instance, no two human translators can translate the same material into the same language pair in exactly the same way, and accurate translation may

need several rounds of revisions. Combining the efforts of the human mind and the machine results in a higher quality translation. Although the quality of machine translation is significantly inferior to that of human translation, text can be translated quickly, accurately, and effectively by integrating machine translation with other technologies and human translators. There have been a variety of methods developed to automate translation, each with advantages and problems of their own [18].

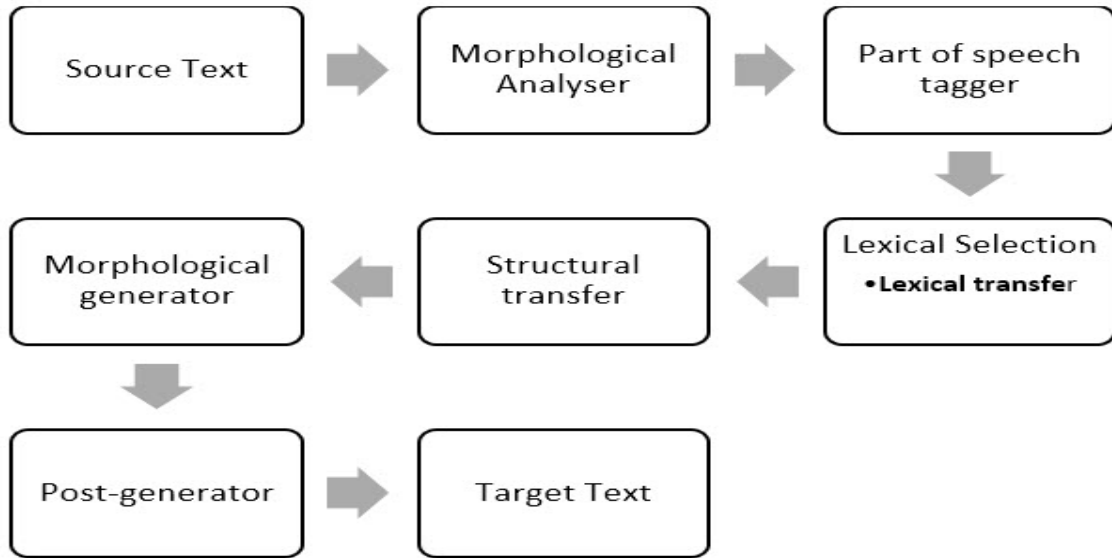
## **2.4. Approaches of Machine Translation**

Different strategies have been presented and put into effect since the idea of employing machines for language translation procedures was developed. Rule-based machine translation and corpus-based translation are the two basic paradigms that machine translation has historically followed. The integration of rule-based and corpus-based MT systems has lately led to the development of hybrid techniques and, most recently, neural machine translation approaches, which are currently dominating the paradigms of machine translation [19].

### **2.4.1 Rule Based Machine Translation (RBMT)**

The first method ever created in the field of machine translation is called Rule-Based Machine Translation (RBMT), sometimes known as Knowledge-Based Machine Translation. Machine translation systems based on linguistic data about the source and target languages are referred by this generic term. RBMT systems provide translation using bilingual and monolingual dictionaries, grammars, and transfer rules. Morphological, syntactic, and semantic information about the source and target languages are managed during translation. To create linguistic principles, this information is used. Rules are important at all phases of translation, including syntactic processing, semantic interpretation, and contextual language processing [18].



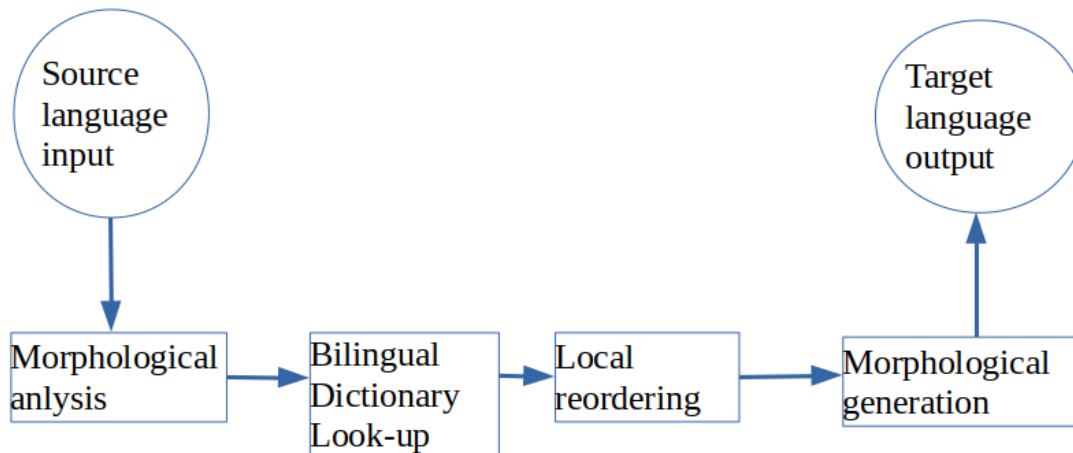


*Figure 2.1: General Steps of Rule-based Machine Translation [16]*

Converting source language structures to target language structures is the aim of RBMT. Direct, transfer-based, and Interlingua are the three sub-approaches that the methodology can use. The sub methods vary in how thoroughly they analyze the source language and how much they try to express meaning or intent between the source and target languages in a way that is not dependent on either language [16].

#### **2.4.1.1 Direct Approach**

In this approach literal translation involves transferring the source language into the target language without any intermediate languages. Without using an additional or intermediary representation, source language words are translated. Word-to-word translation is done with or without keeping the word's sense in this method. Systems for direct translation are essentially unidirectional and bilingual [16]. There won't be any complicated architecture involved in this approach. With the use of a bilingual dictionary, it performs word-by-word translation, usually followed by some syntactic rearrangement. Such systems are heavily reliant on both the source and target languages as a result of this direct mapping.



*Figure 2.2: Major Tasks in Direct Machine Translation Approach [16]*

#### **2.4.1.2 Interlingua Approach**

This method of machine translation (MT) involves converting the source language into a representation that is independent of all other languages used in the translation process. The term "neutral language" refers to this transitional tongue. Languages used as the source and target are irrelevant. The Interlingua is then used to produce the target language. From now on, this type of method just requires two modules: analysis and synthesis. For multilingual systems, the Interlingua technique is undoubtedly the most appealing. One of the main benefits of this technique is that as the number of target languages it may be converted into grows, the Interlingua gains in value. In 1992, Nyberg and Mitamura created KANT, the sole Interlingua machine translation system. It is not simple work to develop an Interlingua language. To create completely neutral language, too much work is needed [18].

#### **2.4.1.3 Transfer-based Approach**

Transfer-based machine translation produces a translation from an intermediate representation that mimics the meaning of the source sentence. Contrary to Interlingua MT, it is somewhat reliant on the language pair being translated. By examining the grammatical structures of both the source language and the target language, a set of linguistic rules are established in this translation in order to retain the meaning of a sentence. Translation can be divided into three phases in the transfer-based approach: Analysis, transfer, and generation come first. The source language text is initially examined using linguistic data to create a syntactic representation of the source language using a

source language parser. The second stage involves the transformation of the source syntactic representation into the target syntactic representation. The target language text is produced using the morphological analyzer in the last step of this translation methodology. Transfer-based systems require rules for lexical, semantic, and syntactic transfer. The source parse tree can be changed to resemble the target parse tree using syntactic transfer rules. Semantic role labeling is used in semantic transmission. On a multilingual dictionary, lexical transfer rules are based. Lexical ambiguity can be resolved by using the dictionary [18] [20].

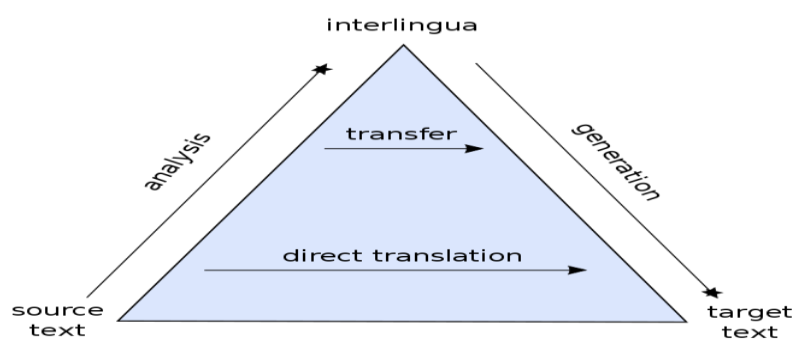


Figure 2.3: Machine Translation Pyramid [18]

#### 2.4.1.4 Advantages and Disadvantages of RBMT

Different approaches of MT have complementary pros and cons. Here are some of the advantages and disadvantages associated with RBMT approach.

##### Advantages

A bilingual corpus is not necessary to construct a rule-based machine translation system. This enables the development of translation systems for languages without any digital information or similarities. The RBMT system is domain neutral and provides excellent out-of-domain quality due to the rules' ability to be applied across a variety of domains. The addition of improved quality and consistency is RBMT's additional benefit. Errors are simpler to identify and troubleshoot since the system is more transparent. Even if the trigger scenario is extremely uncommon, a tailored rule can correct each inaccuracy. Another RBMT advantage is reusability. The foundation of RBMT systems typically consists of a robust source language analysis that is fed to a transfer step and target language generator; the source language analysis and target language generation

components can be shared by multiple translation systems, necessitating the specialization of only the transfer step. Additionally, a substantially related language analysis can be bootstrapped using the source language analysis for one language. The needs of a wide range of linguistic phenomena can be met by RBMT, which is extensible and capable of high accuracy in limited language sets.

### **Disadvantage**

The fundamental disadvantage of rule-based machine translation is that a huge number of rules are required to cover all language aspects, which necessitates extensive linguistic understanding. It is quite expensive because it not only calls for linguistic proficiency but also intensive coding on the part of the programmer, which takes a lot of time and linguistic resources. Large-scale rule interactions, ambiguity, and idiomatic idioms are challenging to handle in RBMT. Rules must be changed in order to increase the quality of an RBMT; but, changing just one rule does not ensure that the RBMT will perform more accurately overall. Additionally, rule updates are typically highly expensive. There are still certain linguistic details that need to be carefully set.

### **2.4.2 Corpus-based Machine Translation**

A key paradigm for creating MT systems is rule-based techniques. To address the extensive variances and temporal shifting properties of the actual text, such techniques struggle to acquire the necessary information. Some statistical translation models and assisting tools had been created to address this issue. A machine translation system with an architecture based on bilingual or multilingual corpora analysis is known as corpus-based machine translation. Data with text and its translation are called bilingual parallel aligned corpora. As an alternative to the rule-based method, it was introduced. A significant amount of raw data is gathered in parallel corpora when using a corpus-based technique, and this data is then used to extract translations for new sentences. The untranslated data includes literature, dictionaries, grammars, and other resources. The translation between the source and target languages is essentially the raw data [16].

The corpus-based method to machine translation has become one of the most extensively researched areas in machine translation since 1989. Examples-Based Machine Translation (EBMT), Statistical Machine Translation (SMT), and Neural Machine Translation are the three variants of the corpus-based technique that have been recently classified. These strategies are succinctly explained in the following subsections [16] [20].

### **2.4.2.1 Example-Based Machine Translation (EBMT)**

Example-based translation, often referred to as Memory-based translation, is a type of corpus-based machine translation that draws its primary information from bilingual corpora of parallel texts. Makoto Nagao proposed the example-based translation approach in 1984. It is based on analogical reasoning between two translation instances. A bilingual corpus is used by an example-based translation at runtime as its primary knowledge basis. These algorithms use the source text as input and search the corpus for the source examples that are most comparable to the source text. Retrieving equivalent translations is the next stage. Recombining the obtained translations with the final translation is the last step. The main principle is that if a sentence that has already been translated reappears, the same translation is likely to be accurate once more. The system uses the example that has already been translated as knowledge. This method extracts data from corpora for analysis, transmission, and translation production [21] [22].

In EBMT, the translation process is essentially a mechanism for matching the input sentence against the stored translated samples rather than utilizing explicit mapping rules to convert sentences from one language to another.

#### **Advantages of an EBMT**

EBMT uses parallel texts as its main knowledge which avoids the need for manually derived rules making quickly adaptable to many language pairs. Because EBMT a memory-based translation, the translation memory saves the user effort of re translating the sentence and this saves the processor time and user time. The other advantage of this model is it works well with small set of data and possible to generate output more quickly by training the translation program.

#### **Disadvantages of an EBMT**

Even while EBMT does away with the requirement for manually developed rules, it still needs analysis and generation modules to create the dependency trees required for both evaluating the sentence and the examples database. Although parallel processing techniques can be used, EBMT's computational efficiency, particularly for big databases, remains a concern. To create the dependency trees required for the examples database and for analyzing the sentence, analysis and

generation modules are required. Because there are no effective techniques for cleaning noisy corpora, the effectiveness of the EBMT system is hindered in cases where there are noisy corpora.

#### **2.4.2.2 Statistical Machine Translation (SMT)**

Statistical Machine Translation (SMT) is a type of machine translation in which translations are produced using statistical models whose parameters are obtained through the examination of bilingual text corpora. Warren Weaver first proposed the concept in 1949, but it wasn't until the late 1980s that researchers at IBM's Thomas J. Watson research center revived it [23]. SMT seeks to use statistical decision theory, which is based on the probability distribution function, to arrive at the best translation decision. Empirical Machine Translation (EMT) systems use statistical translation models that are built from the examination of monolingual and bilingual corpora to perform statistical machine translation [22].

In SMT, a bilingual text corpus is utilized to examine the parameters of the statistical model and statistical probabilities are employed to determine the likelihood of a translation. The availability of a statistical table, which can be created using supervised or unsupervised statistical machine learning techniques, is a key component of SMT. Statistics for phrases or languages are typically included in statistics tables. Instead of utilizing linguistic translation methods, SMT calculates the odds of a match statistically using two probabilistic models: The Language Model and the Translation Model.

The idea of SMT is that document can be translated on the basis of probability distribution function. And this function is generated easily by using Bayes theorem. In Bayes theorem probability distribution  $P(t/s)$  is obtained from the product of  $P(s/t)$  and  $P(t)$ , where  $P(s/t)$  is the probability that the source sentence is a translation of the target sentence, and  $P(t)$  is the probability of the target language. Three elements the language model, translation model, and decoder play a prominent role in the SMT architecture as seen in Figure 2.4.

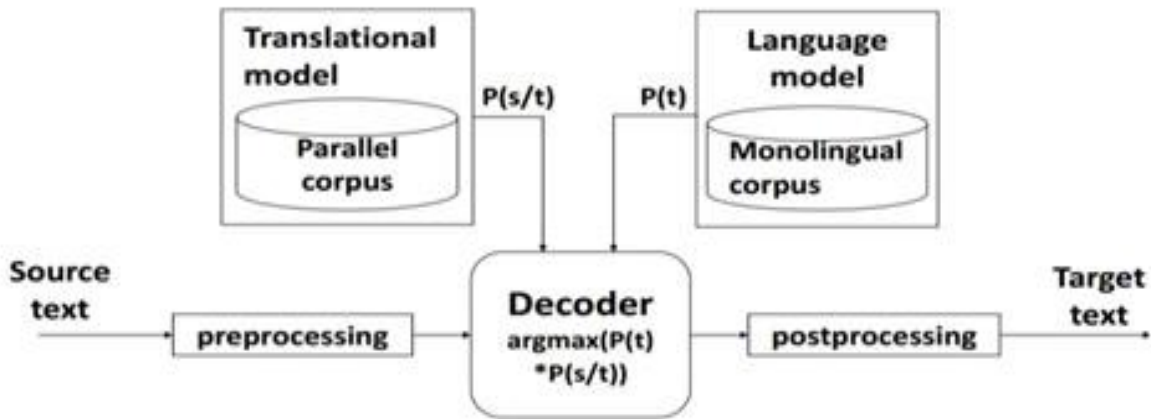


Figure 2.4: Statistical Machine Translation

The language model first determines  $P(t)$  using the monolingual corpus. The appropriate word combinations in the target language are also its responsibility. Thus, it guarantees the output is grammatically correct. Second, the translation model, which ensures that the target hypothesis produced by the machine translation system corresponds to the source phrase, estimates  $P(s/t)$  based on the parallel corpus. The Decoder, which actually does the translation, comes in third [24] [23] Using the following equation, the decoder selects the sentence in the SL that has the highest likelihood of being a feasible translation for a given sentence  $t$  in the target language.

$$P(t|s) = \text{argmax}(P(s|t) * P(t)) \quad (2.1)$$

Depending on the fundamental modeling unit, statistical machine translation can be divided into three groups. These three types of language modeling are word-based, phrase-based, and syntactical-based.

**Word-based Language Modeling:** It is the statistical-based MT system's initial attempt. The basic building block of this approach is words, as its name suggests. The target sentence is created by translating each word in the input sentence word by word, then arranging those translated words in a certain order. In word-based translation, the alignment of the words in the input and output phrases typically follows particular patterns [22].

**Phrase-based Language Modeling:** A phrase or series of words is the basic building block of this approach. Before translation, each source and target sentence are broken up into distinct phrases rather than words. There is a creation of a word order using both the source and target languages.

Based on the vector of items with coordinating properties for the dialect succession pair, decoding is performed [20] [22].

**Syntactical-based Language Modeling:** The translation rule serves as the basic building block of this approach. The source language's word and variable order makes up the translation rule. a target language syntax tree (having words or variables at leaves), The likelihood of the language pair is expressed as a vector of feature values.

### **Advantages of SMT**

The key benefit of this approach is that linguists don't need to customize the tool because it learns translation techniques through statistical analysis of bilingual corpora. When expensive and qualified corpora are readily accessible, SMT systems are simple to construct, simple to maintain, effective, and deliver excellent quality. Another benefit of the SMT model that promotes more natural transactions is better resource management. No specific pair of languages is the focus of SMT systems.

### **Disadvantages of SMT**

The main difficulty in SMT system is creating massive parallel corpus. SMT requires preparing parallel text which can be costly for users with limited resources. Training SMT systems requires high computational resources and it is difficult to perform error analysis. The other disadvantage in SMT is it does not work well between languages that have significantly different word orders.

### **2.4.3 Hybrid Machine Translation Approach**

Hybrid Machine Translation (HMT) is a new strategy created by utilizing both statistical and rule-based translation approaches, and it has proven to be more effective in the field of MT systems. By smoothly incorporating the advantages of both technologies, hybrid machine translation approaches have been developed with the goal of resolving the issues brought on by RBMT and SMT systems and producing translation output of greater quality [18]. It blends statistics and rules. The hybrid technique can be implemented in a variety of ways, and generally speaking, architectures having an SMT system or an RBMT system at their heart can be distinguished from each other. The drawbacks of both approaches were taken out of the hybrid approach to machine translation, which now offers a promising translation with high efficiency [22].



## **Advantages of Hybrid MT**

This approach has a lot more power, flexibility and control when translating. Many issues can be addressed at their root causes through rules that go beyond the capabilities on a statistical only approach.

## **Disadvantages of Hybrid MT**

But despite of its advantages, this approach has some demerits like: it requires a big dictionaries or corpora, complexity of the system, domain specific nature of system, lexicon and linguistic irregularities, etc. Hampers its commercial viability, etc.

### **2.4.4 Neural Machine Translation Approach**

For many years, SMT dominated the field of machine translation technology. Long sentences are broken up into small pieces in classical statistical machine translation, which results in poor levels of accuracy. Neural machine translation was developed and frequently used to address this issue. MT using a neural network that directly predicts the conditional probability of converting a given source sentence to a target sentence is known as Neural Machine Translation (NMT). Neural Machine Translation is a new paradigm that swiftly superseded SMT as the predominant method of MT, developed with the development of deep learning.

The NMT approach does not have a separate language model, translation model, or reordering model like the statistical technique does. Instead, one word at a time is predicted using a single sequence model. The task of estimating the likelihood of a string of words is carried out by an artificial neural network. NMT eliminates the need for wasteful feature engineering by modeling the complete translation process with a single, sizable neural network. As opposed to SMT's separately tuned components, NMT's training is end-to-end. Compared to rule-based and statistical machine translation (SMT) systems, NMT has recently emerged as a viable methodology that has produced notable advances [25].

Machine translation employing an "encoder-decoder" structure was first proposed by [26] in the year 1997. A language model based on neural networks was created in 2003 by a team of researchers at the University of Montreal headed by Yoshua Bengio [27], which solved the data scarcity issue of conventional SMT models. Their efforts set the foundation for the use of neural networks in machine translation in the future.

For machine translation, a brand-new end-to-end encoder-decoder structure was put forth in 2013 by Nal Kalchbrenner and Phil Blunsom [28]. Convolutional neural networks (CNN) are used in this model to encode a given source text into a continuous vector, and recurrent neural networks (RNN) are used as the decoder to convert the state vector into the target language. Their research gave rise to the NMT, a technique that maps natural language using deep learning neural networks. In contrast to the linear SMT models, NMT's nonlinear mapping describes the semantic equivalence utilizing the state vectors that link the encoder and the decoder. Additionally, the RNN is meant to be able to comprehend the meaning behind phrases of any length and resolve the "long distance reordering" issue. However, the "exploding/vanishing gradient" problem makes it difficult for RNN to truly handle the long-distance dependencies; as a result, the NMT model did not initially attain a decent performance [29].

Sequence to sequence learning (seq2seq) was developed by [30] in 2014 using RNNs for both encoder and decoder, and the Long Short-Term Memory (LSTM, a type of RNN) for NMT was also introduced. The issue of "exploding/vanishing gradients" is controlled by the gate mechanism that allows for explicit memory deletes and updates in LSTM, which allows the model to capture "long-distance interdependence" in a phrase much better. The "long distance reordering" problem was resolved with the advent of LSTM, while the "fixed-length vector" problem became the main obstacle for NMT.

Since the "attention" method was first introduced by Yoshua Bengio's group [25], the "fixed-length vector" problem has begun to be solved. When performing a prediction job, the neural network is able to concentrate on relevant input components more thanks to the attention mechanism. Only a small percentage of the source phrase is significant when the decoder is creating a word for the target sentence; as a result, a content-based attention mechanism is used to dynamically construct a (weighted) context vector depending on the source sentence. Then, rather than using a fixed-length vector, the target word will be anticipated based on the context vectors. Since then, NMT has significantly increased in performance, and the "attentional encoder-decoder networks" model is now the most advanced one available.

## **Advantages of NMT**

Because all components of the neural translation model are trained together (end-to-end), unlike conventional translation systems, Neural Machine Translation exhibits greater performance in translation. It also exhibits simplicity in contrast to earlier paradigms because it uses fewer components, requires fewer processing steps, and uses less memory than SMT. It requires less knowledge related to the structure of source and target language as well as, it allows to use human and data resources more efficiently than RBMT. NMT systems can understand the broader context of words and phrases to produce more accurate and fluent translations. By contrast, conventional PBMT only considers the context of a few words on either side of the translated word. For morphologically rich target language, NMT can select more correct word forms than RBMT. Also, reordering errors in NMT are lesser than SMT.

## **Disadvantages of NMT**

The main disadvantages of NMT is that it is time consuming to train NMT model. NMT requires large training data to build competitive models. The other drawback of NMT system is: it performs poorly when it comes to the translation of rare words. One reason for that is the limited vocabulary; another is the unreliable training of rare word's embedding as the word embedding represent both source and target words. Also, it is difficult to fix errors in an NMT system because it uses a beam search with almost no constraint for searching target words.

### **2.5 Encoder-Decoder Architecture**

The encoder-decoder sequence to sequence network, commonly known as the standard algorithm for NMT, is an architecture that can be used to create RNNs or Transformers. Recurrent neural networks can be used to solve sequence-to-sequence prediction issues utilizing the encoder-decoder approach. The encoder-decoder framework, which almost all neural machine translation models use, typically consists of two recurrent neural networks (RNNs), one of which consumes the input text sequence and the other of which produces translated output text [13].

The encoder and the decoder are two connected networks in the original encoder-decoder structure, each serving a different purpose in the translation process. Each hidden state of the encoder network compresses the variable-length sequence into a fixed-length vector after reading the

source sentence word-by-word when it gets a source sentence. Encoding is the name of this procedure. The decoder then performs the opposite process by word-by-word translating the thought vector to the target sentence from the encoder's final concealed state also known as the thought vector. This procedure is also known as end-to-end translation since the encoder-decoder structure directly handles the translation task from the source data to the target result, meaning there is no discernible outcome in the middle process. The Encoder-Decoder structure of NMT works on the basis of a semantic space intermediate vector that maps the source sentence to the target sentence. In fact, both languages can use this intermediate vector to describe the same semantic meaning [25] [30].

The process of Encoding and Decoding can be illustrated in the below figure.



*Figure 2.5: The encoder-decoder architecture for NMT*

The source and target sentence pairs will be X and Y, respectively. The encoder RNN turns the source text  $X_1, X_2, \dots, X_n$  into fixed-dimension vectors. Using conditional probability, the decoder produces one word at a time.

$$P(Y|X) = P(Y|x_1, x_2, x_3, \dots, x_n) \quad (2.2)$$

The fixed size vectors encoded by the encoder are indicated in the equation by the  $x_1, \dots, x_n$ . The equation above is changed to the equation below using the chain rule, where the source phrase vectors and symbols predicted up to this point are used to forecast the following word as it is being decoded. The following expression then changes to:

$$P(Y|X) = P(y_1|y_0, y_2, \dots, y_{i-1}; x_1, x_2, x_3, \dots, x_n) \quad (2.3)$$

A softmax over the vocabulary terms is used to represent each term in the distribution. Different neural networks, including long and short memory neural networks and gated recurrent neural networks, can implement the encoder-decoder model because it is a general framework [31].

## **2.6 Attention**

This encoder-decoder method could have a problem because a neural network needs to be able to fit all the information from a source sentence into a fixed-length vector. Long sentences, especially those that are longer than the sentences in the training corpus, may be challenging for the neural network to handle as a result [32]. In 2015 Bahdanau, Cho, and Bengio [25] introduced a modification to the encoder-decoder model that learns to align and translate concurrently in order to address this problem. The suggested approach soft-searches for a collection of spots in a source phrase where the most pertinent information is focused each time a word in a translation is generated. The model then predicts a target word based on all previously generated target words as well as the context vectors connected to these source positions. The encoder typically converts the full source sentence into a fixed-length vector in the encoder-decoder architecture. The translated word is predicted using the most pertinent information from the source sentence and the previously created target words in the model with attention mechanism. By not having to encode all of the information in the source sentence into a fixed-length vector, searching for the most pertinent sections of the source phrase eases the encoder's workload.

The decoder receives information from each hidden state of the encoder through an interface (attention) that connects the encoder and decoder. With the help of this framework, the model may pick concentrate on the most important segments of the input sequence and then understand the associations between them. This enables the model to effectively handle lengthy input sentences. The attention mechanism lowers the cost of calculation for NMT.

## **2.7 Language Modelling**

The process of giving sentences in a language a probability is known as language modeling (LM). It explains the word order in a natural language. In order to provide a foundation for their word predictions, language models examine corpora of text data. It can be thought of as the computation of a single word's probability given every word that comes before it in a phrase. It makes an effort to imitate the natural language's built-in regularities (in word order). With applications in speech detection, text production, and machine translation, language modeling is a basic task in AI and NLP [33].

The most crucial piece of information in the MT job is actually provided by the language model: the likelihood that a specific word (or phrase), which is dependent on earlier words, will emerge. Therefore, enhancing the LM will undoubtedly enhance translation performance. Statistical language models and neural language models are the two main categories of language models (NLM). The Statistical Language Models learns the probability distribution of words using conventional statistical methods like N-grams, Hidden Markov Models (HMM), and specific linguistic rules. It typically entails creating an n-th order Markov assumption, calculating n-gram probabilities through counting, and then smoothing [33]. This language model is expressed as a probability distribution over a sequence of strings (words). Learning the joint probability function of word sequences in a language is one of the objectives of statistical language modeling. The curse of dimensionality, which refers to the requirement for enormous quantities of training data when learning very complicated functions, makes this inherently challenging. The number of instances needed can expand exponentially as the number of input variables rises. When a large number of distinct input variable value combinations must be distinguished from one another and the learning process requires at least one example for each significant value combination, this is known as the "curse of dimensionality". The issue with language models arises from the enormous number of word combinations that are feasible; for instance, there are 1050 different combinations for a sequence of 10 words drawn from a vocabulary of 100,000 words [27].

Although probability of rare n-grams can be poorly predicted due to data scarcity, statistical language models are easy to train (despite smoothing techniques). In order to address the n-gram problem, Bengio et al. first suggested the neural language model in 2003 [27]. Better language model development frequently leads to models that perform better on the NLP job they were designed for. The goal of better language model training is beneficial in and of itself since it frequently enhances the underlying metrics of the downstream work (such as the BLEU score for translation). Due to the ease of the modeling stages, neural language models (NLMs) became the preferred option.

## **2.8 Neural Language Model (NLM)**

Continuous representations or embedding of words are used by neural language models also known as continuous space language models to generate predictions. To simulate a language, these models employ several types of neural networks. In order to lessen the effects of the curse of

dimensionality, NLM make use of their capacity to learn dispersed representations. By encoding words in a distributed manner as non-linear combinations of weights in a neural net, neural networks are able to overcome the problem of the curse of dimensionality. By parameterizing words as vectors (word embedding) and using them as inputs to a neural network, this model can address the problem of n-gram data scarcity [27]. As part of the training process, the parameters are learned. Similar to non-neuronal approaches like Latent Semantic Analysis, word embedding produced by NLMs have the feature that semantically similar words are also similar to one another in the induced vector space. In both standalone LMs and when models are merged into bigger models for difficult tasks like machine translation, NLM approaches outperform conventional methods.

Although it has been demonstrated that NLMs perform better than count-based n-gram language models, they are blind to sub word information (e.g. morphemes). As a result, it is difficult to predict the embedding of unusual words, which causes significant perplexities for rare words (and words surrounding them). This is particularly problematic for languages with complex morphology and long-tailed frequency distributions, as well as for domains with dynamic vocabularies [34]. To solve the difficulty of learning long-term dependency problem various improvements were proposed. Some novel, effective methods, including Long Short-term Memory RNN Language Model, character-aware models, factored models, bidirectional models, caching, attention, etc., are proposed. Recently, attention mechanisms have been introduced to improve NNLMs, which achieved significant performance improvements.

## **2.9 Network Models in Neural Machine Translation**

The precise design of the neural networks utilized for machine translation varies between neural machine translations. Recurrent neural networks (RNNs) and convolutional neural networks (CNNs) are the two main strategies. RNNs translate texts by reading a sentence in one language and forecasting an orderly series of words with the same meaning in another language, either strictly left-to-right or right-to-left. RNNs are currently used by Google Translate and other programs to search through databases of texts, statistically analyze them, and then offer the results that are most plausible. RNNs therefore process information methodically and linearly. Despite traditionally outperforming CNNs at language translation tasks, RNNs have a design flaw that may be recognized by considering how they handle data. Because RNNs process words one at a time

in a strict left-to-right or right-to-left order, they don't naturally fit with the extremely parallel GPU hardware that drives contemporary machine learning. Due to the requirement that each word wait until the network has finished processing the preceding word, the computation cannot be fully parallelized. RNN based NMT models however, differ in three main terms:

- a) Some studies have chosen a straightforward unidirectional RNN in terms of directionality—unidirectional or bidirectional. To accept the input sentence, Luong et al., for instance, employed unidirectional RNN directly. A different popular option that can improve translation quality is bidirectional RNN. This is so that the model's ability to accurately forecast the current word depends on how well it "knows" the information in the context word. Clearly, this capability might be strengthened by a bidirectional RNN.
- b) When comparing single layer RNN with multi-layer RNN in terms of depth, single layer RNN typically performs worse than multi-layer RNN. Nearly all models with competitive performance in recent years have used deep networks, indicating a tendency of adopting a deeper model to produce the most up-to-date result and
- c) In terms of type, frequently either an LSTM, a Gated Recurrent Unit, or a vanilla RNN (GRU). Both LSTM and GRU are more resilient than a plain RNN in dealing with the gradient exploding and vanishing problem. Another job for sequence processing has similarly shown improved results for GRU and LSTM [35].

CNNs, on the other hand, have the ability to process data hierarchically, enabling them to search for non-linear relationships in the data. This has implications for translation because it makes it simpler for CNN to understand context and translate appropriately. Today, Facebook adopts CNN's strategy for its translation services. Convolutional architecture is currently being taken into account for machine translation. They are consequently more effective computationally. The fact that information is processed hierarchically by CNNs, which makes it simpler to capture complicated relationships in the data, is another benefit [36]. CNN-based models outperform RNN-based NMTs in terms of training speed because of CNN's inherent structure, which enables parallel computations for its various filters when processing input data. Additionally, the gradient vanishing issue in CNN-based models is now simpler to overcome because to the model structure. However, their translation quality suffers from two catastrophic flaws. The long dependency of words can only be identified in high-level convolution layers because the initial CNN based model



can only capture word dependencies within the width of its filters; this unnatural nature frequently results in a worse performance than the RNN based model. Second, when the sentence lengthens, performance suffers significantly because the original NMT model compresses a sentence into a fixed size of the vector. This results from the vector's finite capacity for representation. Early RNN-based model proposals also exhibit a similar issue, which is eventually mitigated by the Attention Mechanism [35].

### 2.9.1 Recurrent Neural Network

Recurrent neural networks (RNNs) are a subclass of neural networks that are particularly well adapted for handling sequential input, such as text sentences where words depend on one another. With hidden states, RNNs enable the use of prior outputs as inputs. This is ideal for modeling languages because each word in a language is dependent upon the ones that come before it. The hidden state is updated and used to process the subsequent word at each time step as a sequence is processed one word at a time [37]. In order to analyze sequences, a recurrent neural network (RNN) iterates through the sequence's elements while keeping track of its current state, which contains data about what it has seen so far.

RNN is made to extract contextual data by identifying the relationships between different time stamps. It is made up of a great deal of successive recurrent layers that are successively modeled in order to map the sequence with other sequences. RNN is quite good at extracting contextual information from the sequence. However, the network structure's contextual cues are reliable and useful for achieving the data classification procedure. The length of the sequences doesn't matter while using RNN.

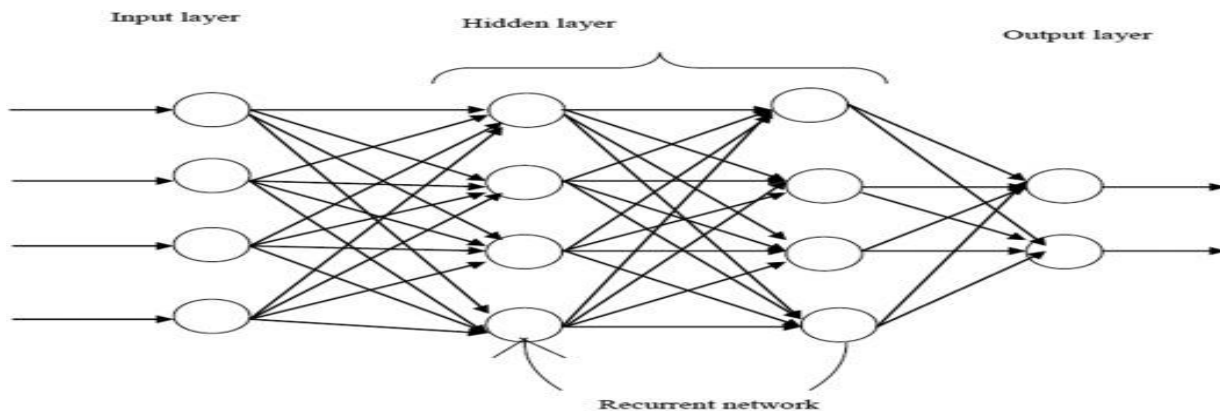


Figure 2.6: Architecture of RNN

Using a pure deep RNN model, [30] proposed the first successful RNN-based NMT and obtained performance that is comparable to the best SMT result. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation is one example of a large-scale system that swiftly adopted the RNN-based NMT technique, or RNMT, as the de-facto standard for NMT [13].

Following RNMT, convolutional neural network-based approaches [38] to NMT have recently drawn research attention due to their ability to fully parallelize training to take advantage of modern fast computing devices such as GPUs and Tensor Processing Units (TPUs). Well known examples are ByteNet and ConvS2S. The ConvS2S model was shown to outperform the original RNMT architecture in terms of quality, while also providing greater training speed.

Most recently, the Transformer model [39], which is based solely on a self-attention mechanism and feed-forward connections, has further advanced the field of NMT, both in terms of translation quality and speed of convergence. NMT has already been widely deployed in production systems by Google, Microsoft, Facebook, Amazon, SDL, Yandex, and many more [40].

Although RNNs are often regarded as the standard text architecture, they come with their own set of issues, including the inability to retain past material for extended periods of time and difficulty producing lengthy relevant text sequences due to explosion or vanishing gradient issues. Due to these factors, new architectures were created and established as the most advanced method for various language production tasks, including Long Short-Term Memory (LSTM) [41] and Gated Recurrent Units (GRU) [42].

### **2.9.2 Long Short-Term Memory**

To stop the error gradient from declining over time and either disappearing entirely or growing exponentially, Hochreiter and Schmidhuber created the Long Short-Term Memory (LSTM) in 1997 [43]. The LSTM is a memory cell, as its name suggests. Cells and gates both play a role in memory modification and information retention. From the first to later time steps, the knowledge is carried by the Cell States without disappearing.

The sigmoid activation, also known as tanh activation, is used by gates. Tanh activation ranges from 0 to 1. The input gate, forget gate, and output gate are the three gates that make up an LSTM. Forget Gate: The forget gate eliminates information that is no longer relevant in the cell state. The

gate receives two inputs,  $c_t$  (input at the current time) and  $h_{t-1}$  (prior cell output), which are multiplied with weight matrices before bias is added. The output of the activation function, which receives the outcome, is binary. If a cell state's output is 0, the piece of information is lost, however if it is 1, the information is saved for use in the future. Input gate: The input gate is responsible for adding important information to the cell state. The inputs  $h_{t-1}$  and  $c_t$  is used to control the information first using the sigmoid function, which filters the values that need to be remembered in a manner similar to the forget gate. Then, using the tanh function, a vector is produced that contains all possible values for  $h_{t-1}$  and  $c_t$  and has an output range of -1 to +1. To extract the useful information, the vector's values and the controlled values are finally multiplied. The output gate's job is to gather pertinent data from the current cell state and display it as output. The tanh function is first used to the cell to create a vector. The data is then filtered by the values to be remembered using the inputs  $h_{t-1}$  and  $c_t$ , and the information is then controlled using the sigmoid function. The vector's values and the controlled values are finally multiplied and supplied as input and output to the following cell, respectively [44].

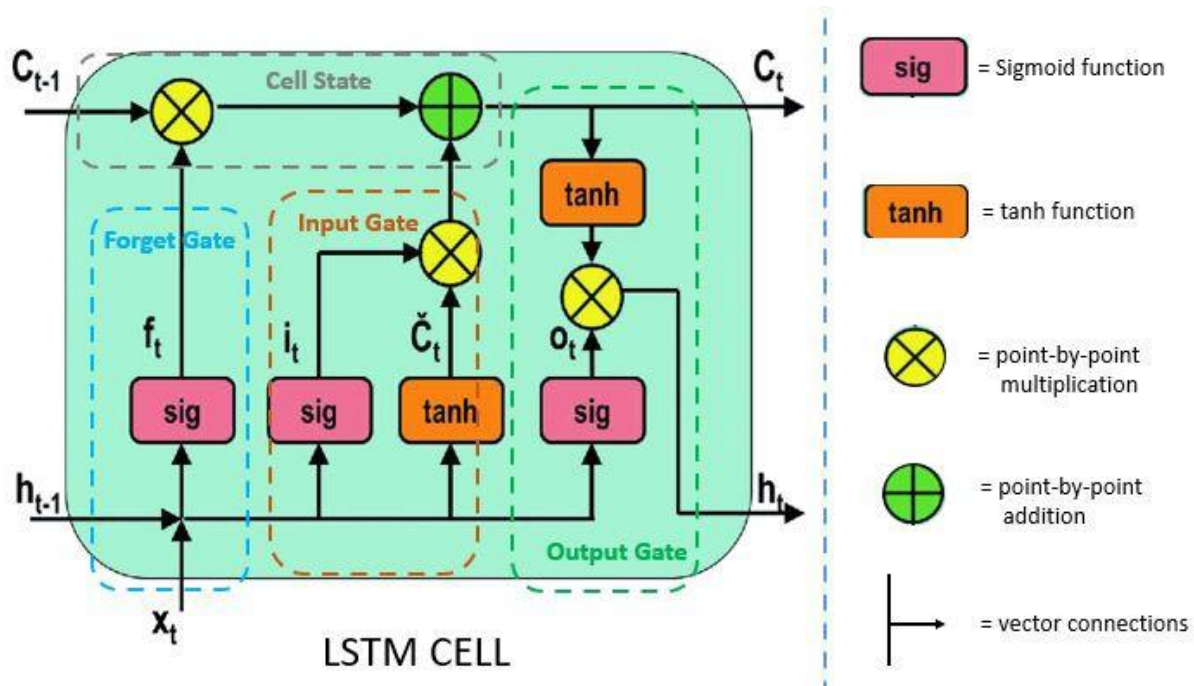


Figure 2.7: Gated Recurrent Units (GRU)

The GRU, a more straightforward version of the LSTM, was released in 2014 [45]. This unit has two gates that regulate how much information is remembered or forgotten: a reset gate and an update gate. Each recurrent unit is produced by a gated recurrent unit (GRU), which was developed to capture dependencies on various time scales. The GRU has gating units that regulate the flow of information inside the unit without the use of separate memory cells, much like the LSTM unit does. It only has three gates, unlike LSTM, and it doesn't keep track of the internal state of the cell. The data that is kept in an LSTM recurrent unit's internal cell state is incorporated into the gated recurrent unit's hidden state. The next Gated Recurrent Unit receives this group of data.

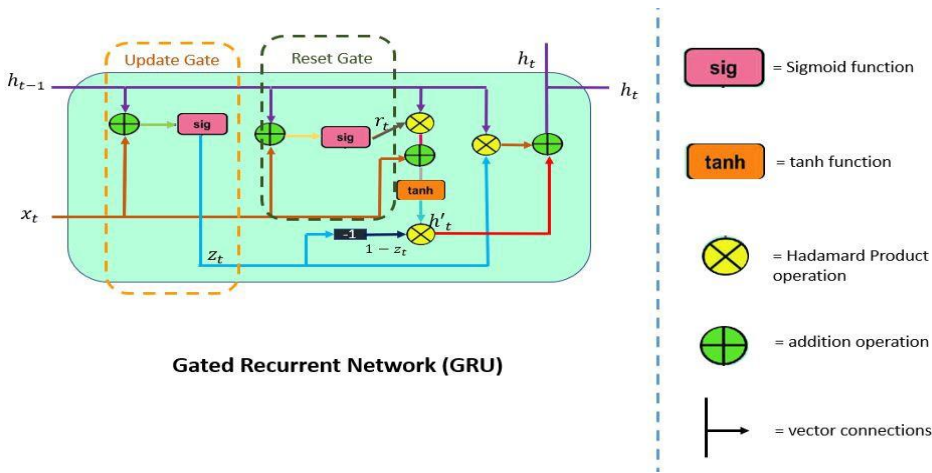


Figure 2.8: Long Short-Term Memory (LSTM)

**Update Gate:** The update gate assists the model in deciding how much historical data from earlier time steps should be transmitted to the future. Forget Gate and Input Gate are combined to create Update Gate. It is comparable to an LSTM recurrent unit's Output Gate.

**Reset Gate:** In order to prevent gradient explosion, this gate resets the previous information. The amount of prior knowledge that should be forgotten is determined by Reset Gate. It is comparable to how the Input Gate and Forget Gate work together in an LSTM recurrent unit.

**Current Memory Gate ( $h_t$ ):** Similar to how the Input Modulation Gate is, a component of the Input Gate is utilized to provide some nonlinearity into the input as well as make the input Zero-mean and it is incorporated into the Reset Gate. Making it a component of the Reset gate also lessens the impact that knowledge from the past has on information that is being sent into the future [44].

## **2.10 Related Work on Machine Translation**

Many attempts were made to create a machine translation between various languages in the past. This subsection examines some of the related works discovered while reading up on machine translation research involving the Wolaytta language as well as other Ethiopian and foreign languages.

### **2.10.1 Machine Translation Involving Wolaytta Language**

Few studies have been done on the Wolaytta language using various techniques and resources. This section will review existing research on machine translation, with a focus on the Wolaytta language.

#### **Bidirectional Dictionary Based Machine Translation for Wolayitegna-Amharic by Java**

To help Amharic speakers use Wolayitegna and vice versa, Temesgen Mengistu [46] created a bidirectional dictionary-based machine translation to convert Wolayitegna to Amharic. In order to provide an exact translation for the experiment, 5400 dictionary entries were created in a MySQL database, and Java was used to design the user interface. The bilingual dictionary served as the foundation for a machine translation of these two languages in this research project. An arrangement of source language words and their associated target language words is defined. Dictionary-based translation uses bilingual corpus, which is defined in the form of a dictionary, as its database throughout run time. The translation memory houses this database. The study adopted the dictionary-based machine translation strategy since it is the most advised for languages with comparable structures and little linguistic resources, like Wolaytta (Wolaytta and Amharic). An Amharic definition for each word from the Wolaytta source language was provided in a bilingual dictionary.

Words with multiple meanings were some of the difficulties the researcher encountered during this research. Some words in Wolayitegna have the same spelling and pronunciation but a different meaning depending on the context of the sentence. In order to translate between two languages, dictionary-based machine translation uses a database-stored word-based dictionary. Another difficulty encountered during this investigation was that Wolayitegna only accepts postfix while Wolayitegna has no proposition and requires both prefix and postfix. This distinction made it difficult to translate between two languages since, from a single root Wolayitegna word, we may

construct a number of words with postfixes that might or might not be derived from the same Amharic word.

The researcher suggested that by incorporating more corpora and contextualizing grammatical translation for both languages, the dictionary-based system may be made even better.

### **English-Wolayita Machine Translation using Statistical Approach**

Melaku Mara [10] conducted the experiment with the goal of translating English text into Wolayita text using a statistical machine translation method. 30,000 bilingual corpora from the spiritual realm and 39,893 monolingual corpora from other sources were collected to meet the research project's goal. In the parallel corpus, the researcher preprocessed data. Normalization, tokenization, lower-case and clean, and sentence alignment are a few of the preprocessing task. A variety of freely accessible tools were utilized, including the SRILM toolkit for language modeling, MGIZA++ to align the corpus at the word level using IBM models (1–5), Moses for decoding, and the Ubuntu operating system, which is appropriate for Moses environments. Additionally, Wolaytta text is segmented using the unsupervised morpheme segmentation tool Morfessor, and the BLEU score is used for evaluation.

In this study, both un segmented and segmented experimental groups were run in order to develop SMT for the English-Wolaytta language pair. Six distinct corpora were used for each experiment group. The parallel sentences were divided into groups of 5, 10, 15, 20, 25, and 30. 95 percent of the sentences in each corpus were utilized for training, 2 percent were used for tuning, and 3 percent were used for testing. The un segmented corpus uses the previously divided parallel sentences to perform BLEU scores of 4.91 percent, 6.30 percent, 7.21 percent, 7.60 percent, 7.96 percent, and 8.46 percent. The segmented corpus uses the previously divided parallel sentences to perform BLEU scores of 9.83 percent, 11.38 percent, 12.70 percent, 12.77 percent, 12.93 percent, and 13.21 percent.

By recording the results of each experiment, the researcher was able to determine that the segmented approach had a superior BLEU score than the un segmented English-Wolaytta combination, which was 8.46 percent. Based on the results of the studies, the researcher concluded that larger corpora and morphological segmentation would result in greater performance. The

researcher also suggested that future studies should concentrate on morphological segmentation and expanding the corpus size in order to further enhance the system's performance.

### **Attention based Wolaita-Amharic Neural Machine Translation**

Workineh Wogasso conducted the study with the aim of creating an attention-based neural machine translation strategy for an Amharic-to-Wolaita machine translation system [9]. He gathered datasets from various sources, primarily religious books, totaling 9280 Amharic-Wolaita parallel phrases. 80 percent of the total data from the corpus is used for the training set, while the remaining 20 percent is used to test the system. Thus, the parallel corpus that makes up the training set totals 7424 sentences, whereas the test set only has 1856. To implement the system, various tools were employed. He employed the Python programming language along with a selection of open-source deep learning libraries, including Keras, TensorFlow, and NumPy.

The Sequence to Sequence (Seq2Seq) model, based on Encoder-Decoder architecture, was used to build the system by fusing Recurrent Neural Networks (RNN) with Gated Recurrent Units (GRU). He used two separate methods in two trials to test the system's accuracy. The first experiment had a BLEU score of 0.5960 and was carried out using a non-attention-based methodology. The attention-based technique was used in the second trial, which had a BLEU score of 0.6258.

The studies' findings indicate that the attention-based system performs better in translations, with a BLEU score improvement of +0.02978, and requires less training time than the non-attention-based system. The experiment's findings also shown that, as sentence length increases, the attention-based model performs better than the non-attention-based approach.

## **2.10.2 Machine Translation Involving other Ethiopian Languages**

### **Amharic-Arabic Neural Machine Translation**

Ibrahim Gashaw and HL Shashirekha conducted a research to create a neural machine translation between Amharic and Arabic [47]. The researchers created a modest size Amharic-Arabic parallel text corpus using Quranic text corpora that were available on Tanzile in order to conduct the experiment because Amharic and Arabic lack parallel corpora for the purpose of building NMT.

They manually divided the verses into separate Amharic-language source sentences and Arabic-language target sentences. 13,501 Amharic-Arabic parallel phrases totaling 3.2 MB in size were prepared, and they are divided into training (80%) and the remaining 20% for testing. Both the Amharic and Arabic scripts were preprocessed, the sentences were manually separated and aligned, and all punctuation was then deleted from the texts.

The researchers used an open-source OpenNMT system to create two LSTM and GRU-based NMT models utilizing an attention-based encoder-decoder architecture. Bilingual Evaluation Understudy is used to test the models (BLEU). They compared Google Translation System, a free multilingual translation tool developed by Google to translate multilingual text, with the two recurrent units LSTM and GRU based OpenNMT translation algorithm, finding that LSTM based OpenNMT outperforms the other, with BLEU scores of 12 percent, 11 percent, and 6 percent for LSTM, GRU, and GNMT, respectively. The outcome shows that LSTM-based NMT performs better than GRU-based NMT.

It was regarded as a good performance for a small size corpus because their experiment was the first one performed on an Amharic and Arabic parallel text corpus. Finally, the researchers suggested that for improved performance, a lengthy experiment with lots of training data may be used.

### **A Parallel Corpora for bi-directional Neural Machine Translation for Low Resourced Ethiopian Languages**

In this study, a team of researchers used neural machine translation to create parallel corpora for English and Ethiopian languages such Wolaita, Gamo, Gofa, and Dawuro [48]. A parallel dataset is gathered from the internet and pre-processed before being used in an NMT experiment. The dataset was separated into a train, validation, and test set for the experimental purpose. 80 % of the dataset was utilized for training, while 20 % was used for testing. The training set was further split into training set and validation set, each comprising 70% of the training set. As a starting point for neural machine translation, a bi-directional neural machine translation experiment has been carried out using the gathered corpus. The test findings demonstrate that neural machine translation performs well when compared to a baseline experiment with BLEU scores of 13.8 for



Wolaita-English and 8.2 for English-Wolaita. Comparatively speaking, the Wolaita-English translation performs better than those of the other Ethiopian language pairs.

The experiment's findings demonstrate that the performance of neural machine translation depends on the size of the dataset and improves as it grows. In addition to these factors, the morphological diversity of the Ethiopian language also played a role in the poor results of neural machine translation when Ethiopian was the target language. The researchers proposed that to improve the performance of the NMT model, more datasets should be used, and alternative domains should be used with more linguistic variables for Ethiopian languages.

### **Bi-Directional English-Afan Oromo Machine Translation Using Convolutional Neural Network**

By using convolutional neural networks on translations between these language pairs, Arfaso Birhanu [49] hopes to improve the prior work on machine translation from English to Afan Oromo by making the translation bidirectional. He gathered a total of 5550 parallel phrases to accomplish his goal, including passages from the Bible, published conversational novels, regional and federal Ethiopian governmental constitutions, Oromia regional revenue, and Oromia health sectors. A total of 20% of the dataset was used for testing, while the remaining 80% was used for training. In order to accomplish the goal, he put three study-designed systems into use and similarly trained the systems to obtain an accurate comparison of their performance. For the bidirectional translation of Afan Oromo and English, three different approaches were used: a word-based statistical technique as a baseline, the RNN method as a competitive model, and convolutional neural networks.

The BLEU scores from English to Afan Oromo and Afan Oromo to English were 20.51 and 19.86 for the Baseline (STM) model, 22.79 and 21.67 for the RNN-based model, and 24.37 and 23.18 for the CNN-based model. In comparison to the baseline system, the CNN translations from English to Afan Oromo and vice versa improved by 3.86 and 3.32 BLEU values, respectively. In addition, the BLEU score improved over the RNN technique in the translations from English to Afan Oromo and from Afan Oromo to English, respectively, by 1.58 and 1.51. When the results of the CNN-based model were compared to those of the RNN-based model and the STM model, the CNN-based model outperformed both in terms of translation quality and training requirements. Finally, when the results of translation in the two directions are compared, translation from Afan Oromo to English yields a higher BLUE score.

The researcher suggests that future study include more datasets to improve translation quality even further and test the systems on GPU-based computers to further cut down on training time for the systems.

### **Bidirectional Tigrigna – English Statistical Machine Translation**

Mulubrahan Hailegebreal [50] conducted a study in order to create a statistical machine translation-based translation system from Tigrigna to Amharic. The Holy Bible, the FDRE Constitution, and basic sentences made up the corpus. The corpus was divided into five groups, referred to as Corpus I, Corpus II, Corpus III, Corpus IV, and Corpus V, and prepared in a format appropriate for use in the development process. Baseline (a phrase-based machine translation system), morph-based (based on morphemes collected using an unsupervised method), and post-processed segmented systems are the three sets of experiments that are carried out (based on morphemes obtained by post-processing the output of the unsupervised segmented).

The language modeling, word alignment, segmentation goal, and automatic evaluation technique were built using IRSTLM, GIZA++, Morfessor 1.0, and BLEU, respectively. 90% of the corpus data were utilized for training, and 10% were used for testing. The experiment's findings indicate that the post-processed segmented system outperforms the other two for the Tigrigna-English language pair. The researcher found that using Tigrigna and English as the source and target sentences, respectively, resulted in higher translation accuracy in each experiment. As a result, the post-processed experiment employing corpus II produced a better result, with a BLEU score of 53.35 % for Tigrigna-English translations and 22.46 % for English-Tigrigna translations.

The researcher concludes by advising that segmenting only prepositions and conjunctions has greatly improved the BLEU score. The translation quality may be further enhanced by carefully segmenting various Tigrigna language derivational and inflectional morphs. This may be a topic for research in order to enhance the functionality of a translation system for this language combination.

### **2.10.3 Machine Translation Involving Non- Ethiopian (Foreign) Languages**

#### **Google's Neural Machine Translation system: Bridging the Gap between Human and Machine Translation**

The design and implementation of GNMT, a Google production NMT system, are presented by Yonghui Wu, Mike Schuster, et al [13]. The system aims to address many NMT-related problems, such as slower training times, inefficiency in handling rare words, and occasionally failing to translate all words in the source sentence. To avoid delayed training, their model has a deep LSTM network with 8 encoder layers and 8 decoder levels. They employ an attention mechanism that joins the bottom layer of the decoder and the top layer of the encoder in order to increase parallelism and hence shorten training time. They use low-precision arithmetic for their inference computations to speed up translation. Additionally, they used sub-word units, also referred to as word-pieces for inputs and outputs, to address unusual terms. The model was able to translate all of the inputs by using a beam search strategy. Their model carefully adheres to the conventional sequence-to-sequence learning architecture. An encoder network, a decoder network, and an attention network make up its three parts. The WMT'14 English-to-French (WMT EnFr) and English-to-German (WMT EnDe) corpora, which are widely used as benchmarks for Neural Machine Translation systems, were employed in the experiment. They used a beam search method to implement their plan.

They evaluate GNMT using Google's translation production corpora in addition to publicly accessible corpora, which are two to three decimal orders of magnitude larger than the WMT corpora for a particular language pair. They contrast the accuracy of their model with that of human translators and Google Translate's top production system for phrase-based machine translation (PBMT). The training sets on WMT En Fr and En De have 36M and 5M sentence pairs, respectively. In all instances, the 2014 News Test was used as the test sets to contrast with earlier works.

The BLEU score metric was used to assess their system. The single model performs 38.95 BLEU on WMT'14 English-to-French, which is an improvement of 7.5 and 1.2 BLEU from the single model without an external alignment stated, respectively. Additionally, their models were totally independent. The single model performs similarly on WMT'14 English-to-German, scoring 24.17 BLEU, which is 3.4 BLEU higher than a previous competitive baseline. Last but not least, when

compared to Google's previous phrase-based translation system on the aforementioned language pairs, the GNMT model reduces translation errors by an average of 60%.

### **Sequence to Sequence Learning with Neural Networks**

The creation of Sequence to Sequence Learning using Neural Networks for English to French Machine Translation is presented by Ilya Sutskever, Oriol Vinyals, and Quoc V. Le [30]. They provide a generic, end-to-end method for sequence learning in this study that places the bare minimum of presumptions on the sequence structure. The input sequence is first translated into a vector with a fixed dimensionality using a multilayered Long Short-Term Memory (LSTM), and the target sequence is then extracted from the vector using a deeper LSTM. They used a subset of 12 million phrases with 348 million French words and 304 million English words from the WMT'14 English to French dataset to train their models. With an input vocabulary of 160,000 and an output vocabulary of 80,000, they used deep LSTMs with 4 layers, 1000 cells per layer, and 1000-dimensional word embedding.

The key output of their work is that, on the WMT'14 English to French translation problem, they directly extracted translations from an ensemble of 5 deep LSTMs (each with 380M parameters) using a straightforward left-to-right beam-search decoder, resulting in a BLEU score of 34.81. This is unquestionably the best outcome that direct translation using big neural networks has ever produced. They used the 33.30 BLEU score of an SMT baseline from the same dataset for comparison. The LSTM used to generate the 34.81 BLEU score had an 80k word vocabulary, hence any terms in the reference translation that were not in this vocabulary reduced the score. This finding demonstrates that a phrase based SMT system outperforms a somewhat under-optimized neural network architecture with lots of space for advancement.

Although the LSTM is capable of addressing issues with long-term dependencies, they found that in the training and test sets, the LSTM learns far better when the source sentences are reversed but not the target sentences, and they achieved a 36.5 BLEU score by doing so. Since doing so created several short-term dependencies between the source and the target sentences and simplified the optimization problem, they found it to be incredibly beneficial to reverse the order of the words in the input phrase, which enhanced the LSTM's performance.

In conclusion, there have been numerous studies done in Ethiopian and foreign languages on statistical, rule-based, hybrid, and neural machine translations between various languages. In the majority of cases, statistical machine translation was employed, while several studies have combined statistical and rule-based approaches. Machine translations frequently employed by the statistical MT methodology; however, the goal of this work is to demonstrate how implementing encoder-decoder architecture NMT with attention mechanism can be and implementing a new machine translation system for English-Wolaytta Language.

## CHAPTER THREE

### THE WOLAYTTA LANGUAGE

#### 3.1 Introduction

In this Chapter, a brief overview on Wolaytta language is provided. The chapter concerned on the inflectional and derivational morphology of the language. The major Wolaytta word classes, which are nouns, verbs, adjectives and conjunctions, also described in this chapter.

#### 3.2 Overview of Wolaytta Language

Ethiopia's Wolaytta or Wolaytta is an administrative region. The Wolaytta people, whose ancestral home is in the zone. The name "Wolaytta" is a representation of the people, place, and language [51]. Gamo Gofa borders Wolaytta on the south, the Omo River on the west, which divides it from Dawro, Kembata Tembaro on the northwest, Hadiya on the north, the Bilate River on the east, which divides it from Sidama Region, and the Lake Abaya on the south-east, which divides it from Oromia Region. Sodo serves as the administrative hub for Wolaytta. Areka, Boditi, Tebela, Bele, Gesuba, Gununo, Bedessa, and Dimtu are further significant towns.

The native peoples refer to their language as "Wolaitta" (Wolayttattuwa in their language). The Wolaytta Zone and neighboring areas of Ethiopia's Southern Nations, Nationalities, and People's Region are home to Wolaytta, a North Omotic language of the Ometo group. It is the Wolaytta people's native tongue [52].

The language is also referred to as Wolaitta doonna (literally mouth of Wolaytta) or Wolaitta Kaalaa (literally word of Wolaytta). These words are said to relate to the Wolaytta verb root walakk- to mix (v.t.)<sup>4</sup> and their derivatives such as waláh-étt- to be mixed, to mingle with<sup>4</sup>, waláh-aa mixture<sup>4</sup>, waláh-ett-aa mixing<sup>4</sup>, this naming reflects the history or origin of Wolaytta people: they claim that many races or tribes mixed with each other to form Wolaytta and so on [51].

The language is known by the common names Wolaytta and Wolaittattuwa. Other names for it include Wolaita Doanaa and Wolaitta Kaalaa, which translate to mean "mouth of Wolaytta" (lit word of Wolaytta). This language has been written in a number of different ways using the Latin alphabet. This includes the terms Wolaytta, Wolaitta, and Welaita that [52] and others use. In this

instance, "Wolaytta" is used in the paper. Of the 20–25 languages/dialects that make up the Afroasiatic Phylum, Wolaytta is an Omotic language. The Omotic language family is divided into western and eastern Omotic by [53]. A part of western Omotic is Wolaytta. Then, the Eastern Branch is separated into the south-eastern Banna, Hamar, and Karo languages, and the north-eastern Ari and Dime languages. The Western Omotic language further splits out into the maji and Kafa Gimojan languages. Gimojan and Kafa languages make up the first group, while Nao, Sheko, and Maji languages make up the second. Ometo, Jenjero, and Gimira are the three subgroups of the Gimojan. Since the Sudan Interior Mission originally developed it in the 1940s, the Wolaita language has been written in Latin script on official documents. A team led by Dr. Bruce Adams later changed the writing method. In 1981, the New Testament was completed, and the entire Bible followed in 2002.

The 1985 E.C.-published doctoral research on the Wolaytta language [51] and other studies utilize a separate notation from Wolaytta called "Wolaitatto pitaliyaa xaafiyo wogaa," which translates to "the custom in writing the Wolaytta letters". Wolaytta language is the fourth most extensively used in the nation after Oromifa, Tigrigna, and Amharic.

### **3.3 Morphology of Wolaytta Language**

The study of word creation and structure is known as morphology. It deals with how words are constructed out of morphemes, which are smaller meaning-bearing units. The smallest meaningful unit in a language that has a meaning and cannot be further broken down into a meaningful unit is known as a morpheme. Inflectional and derivational morphology are the two different types. When word stems join with grammatical markers for things like person, gender, number, tense, case, and mode, the process is known as inflectional morphology. Morphemes can be divided into two categories: bound and free. In Wolaytta, there are two of them. In contrast to bound morpheme, which cannot stand alone as a word, free morpheme may. Free morphemes can stand alone, such as town and dog. Only other morphemes are found with bound morphemes, such as "un-." Generally speaking, prefixes and suffixes make up bound morphemes.

While some morphemes are affixes, some are roots. Affix is a morpheme that joins with roots (or stems), altering their meaning in predictable ways. Prefixes or suffixes are the two most common types of affixes. The words re-read, unloved, and eta-agaa in Wolaytta are examples of prefixes

that come before roots. Eta-agaa, eta-asa, and eta-ayyo are prefixes in English. An affix that follows a root is a suffix. Like -est, -er, and -s (quick-est, quick-er, read-s, book-s) [6]. The affixes we just discussed are unique in yet another aspect. When they are connected to the base, they are behaving in a specific way. They are either providing grammatical details or generating a new word [52].

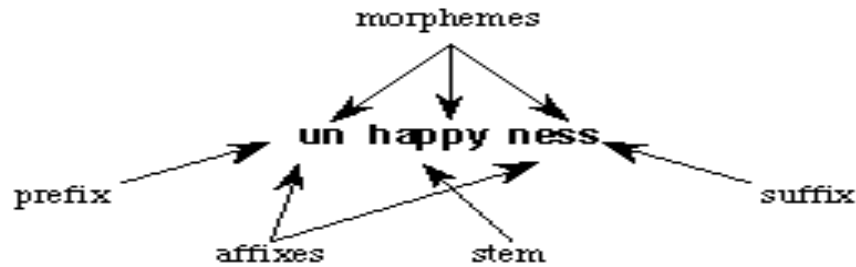


Figure 3.1: Word Morphemes

Among the Ethiopian languages Amharic and Tigrigna uses both types of affixes, Wolaytta language does not have prefix and infix. Instead, Suffixation is the basic way of word formation in Wolaytta. Most words in Wolaytta consist of a lexical stem and a grammatical ending. As stated by [51] [10], the lexical stem has the only suffix but not prefix. For example:

Table 3.1: Suffix Formation of Wolaytta Language

Word	Stem	Suffix
Keexiis	Keexa	-iis
Keexissiis	Keexa	-issiis
Keexidoogaapekka	Keexa	-idoogaapekka



Keexisisidaakaappekka	Keexa	-isisidaakaappekka
-----------------------	-------	--------------------

### 3.3.1 Inflectional Morphology and Derivational Morphology

Wolaytta, like the other Ethiopian languages, has an extremely rich morphology. Inflectional and derivational morphology are the two different types.

When word stems join with grammatical markers for things like person, gender, number, tense, case, and mode, the process is known as inflectional morphology. Parts of speech do not alter as a result of inflectional modifications. Wolaita is extremely inflectional; a given word's root can take on various forms. The Wolaytta language has the following inflection morphologies: siiqa, siiqaasa, siiqaasu, siiqada, siiqadasa, siiqadii, siiqaidda, siiqais, siiqoosona, and siiqida.

Derivational morphology examines the modifications that lead to the change in word classes (changes in the part of speech). For instance, a verb can be used to create a noun or an adjective.

The Wolaytta language's derivation morphology is "siiqakka," "siiqasa," which is derived from the root word "siiqa." It is the act of creating new words from existing ones or their roots, frequently by affixing suffixes like -kka and -asa [6].

#### 3.3.1.1 Inflectional morphology of Wolaytta Language

**A. Personal Pronouns:** A personal pronoun is a pronoun that is primarily used with a specific grammatical person, such as the first person (I), second person (you), or third person (as he, as she). According to number (often solitary or multiple), grammatical or natural gender, case, and formality, personal pronouns can also take on diverse forms. Basic distinctions based on personal pronouns are common in most languages. These distinctions can be found in the fundamental group of independent personal pronouns [10]. Some examples are shown in the following table.

*Table 3.2: List of Pronouns*

Wolaytta	English
Tani	I
A	She
I	He

Etti	They
Nuuni	We
Tana	Me
Nuna	Us
Eta	They

Sentences are made less repetitious by the use of pronouns. Just like in English, a noun or another pronoun can be replaced with a Wolaytta pronoun. They bear markings for their gender and number. For instance, pronouns like "Ta/ tani" that denote "me" in the singular and can be either masculine or feminine, "a" that denotes "she" in the singular and can either be masculine or feminine, I that denotes "he" in the singular, "etti" that denotes "they" in the plural and can either be masculine or feminine, and "nu/nuni" that denotes "we" in the plural Wolaytta pronouns.

**Common Nouns:** The character sequence for Wolaytta nouns is C1V1C2V2, where C and V stand for a consonant and a vowel, respectively. This indicates that the majority of Wolaytta stems are in fact bi-radical. A glottal stop can also be represented by C1. While V1 can be either a short or long vowel or, less frequently, a diphthong, C2 can be either a plain or geminated consonant. Last but not least, V2 denotes either the typical absolute case ending or a thematic vowel that denotes the endings necessary for the grammatical function of the noun that are connected to the noun stem. Although the majority of Wolaytta words are bi-radical (based on the number of consonants in the word), there are a few that are Pluri-radical [52] [54].

*Table 3.3: General Syllable Formulation of a Bi-radical Wolaytta Nouns*

Sequence of Character	Example	Meaning
CVCV	Kaisoi	Thief
CVC: V:	Shappa	River
CV:CV:	Keettaa	House
CV:C: V:	Keettaa	Milk

Pluri-radical forms are also common in Wolaytta nouns. The following table illustrate the pluri-radical Wolaytta nouns form.

Table 3.4: Pluri-radical Wolaytta Nouns Form

Noun	Meaning
Gisttiyaa	Wheat
Gallassatuppe	Day

According to the endings that Wolaytta nouns adopt in their inflection grouped into four primary types. The first class of nouns are those with an absolute case ending in -a and stress on the final syllable. Asa means "person," aawa means "father," and tuma means "darkness." The second class of nouns includes those whose absolute case ends in -iya and which stress their penultimate letter. Morkkiya means "enemy," Penggiya means "the door," and Siya means "listen." The third type of nouns are those with an absolute case ending in -uwa. Examples include "story," "trouble," and "cotton," among others. Nouns in the absolute case that end in -(i)yu belong to the fourth class. These phrases primarily apply to living things that are female. An illustration would be naiyu 'girl' bollotiyu 'mother-in-law' [6] [52].

Wolaytta nouns have gender, number, and case markings.

**I) Gender:** The Wolaytta language uses the masculine and feminine genders. As mentioned above, the fourth-class nouns are feminine, while the first, second, and third-class nouns are masculine. The endings of feminine and masculine words are different from one another. In absolute case, masculine ends in -a whereas feminine ends in -u [6].

**Example:** *dorsa* 'sheep', masculine vs. *dorsiyo* 'sheep', feminine, *dessa* 'goat', masculine vs. *deshiyo* 'goat', feminine, *hage* 'this, masculine' vs. *hanna* 'this, feminine, *taagaa* 'he/it is mine' vs. *taaro* 'she/it is mine'

**II) Numbers:** Wolaytta word comprises both singular and plural forms, according to [52]. The fundamental form of the noun is found in the single, and suffixes are used to make the plural. The basic noun form often makes up the single, whereas a special suffix is employed to create the plural. Wolaytta uses the morpheme -tv to make nouns plural, where -v stands for a terminal vowel that alters according on the case inflection of the plural. The plural marker is -ta in the absolute case, where -v equates to -a. The morpheme -tv is used by all four classes of nouns to create their multiple forms, but a noun's class membership also manifests in its plural form.

Accordingly, the -a-ta ending is used to make **First** class nouns plural.

**Example:** Asa 'person' -- Asata 'persons', Achcha 'tooth'--- Achchata 'teeth'

When making plurals, nouns of the **Second** class have the -eta ending.

**Example:** hariya 'donkey'--- hareta 'donkeys', Orgiya 'he-goat' --- Orgeta 'he-goats'

The -o-ta ending designates nouns belonging to the **Third** class.

**Example:** Oduwa 'tale' --- Odota 'tales', Worduwa 'lie' --- Wordota 'lies'

Finally, **fourth** class nouns that refer to feminine beings use the plural form of the corresponding masculine word.

**Example:** 7imattiyo 'female guest' ---7imatt-a-ta 'female guests'

Nouns of the second class and ending in -e-ta are similar to feminine nouns that lack a masculine counterpart.

**Example:** Machchiyo 'wife' ---machchota 'wives', mishshiriyo 'married woman' ---mishshireta 'married women'

**III) Case:** Wolaytta has a very difficult noun inflection. There are various cases in by adding case endings to the noun stem or the absolutive case form, the inflection is accomplished. As a result, the subject case terminates in -y (first three classes), -i (plural), and -(i)ya, but the absolutive case is distinguished by the endings -a (1st class and plural), -iya (2nd class), -uwa (3rd class), and -(i)yu (4th class), as we have previously seen above in the common noun section (4th class). The genitive is either conveyed just by the noun stem or, more frequently, is identified by the extension of the absolutive form's terminal vowel. The respective absolutive case and the object case of the noun inflection are consistent [6] [52] [54].

The endings listed in the table below serve as a marker for the other cases.

Table 3.5: Case Markers

Case Marker (Morphemes)	Function	Example
-ssi, -w or -yoo	Dative case	Garssassissi
-kko or -mati	Directive case	Garssakko
-ni	Locative	Garsani
-ppe	Ablative	Garsaappe
-ra	Commutative case	Garssara

**B. Adjectives:** A word that reveals more information about a noun is an adjective. It can also change a noun or pronoun by adding phrases that describe, name, or quantify something. Typically, an adjective will come after the noun or pronoun it modifies. In the Wolaytta language, adjectives finish in -a, -iy, or -uwa.

Table 3.6: List of Wolaytta Adjective with their Ending

Adjectives ending in –a	Adjectives ending in –iy	Adjectives ending in –uwa
Geessha 'clean'	Mal’iya 'sweet'	Lo7uwa 'good/nice'
Cinca 'clever'	Yashisiya ‘fierce’	yuushuwa 'round'

Since adjectives do not need to agree with their ruling noun in Wolaytta in terms of gender, number, or case when they are used in the attributive position, they frequently remain unmodified [52]. However, most adjectives ending in -uwa and a small number ending in -iya are replaced by the endings -o and -e respectively when employed in the attributive position.

**Example:** Lo7-uwa 'good/nice' → lo7o asa 'a good person'

haahuwa 'wide/far' → haaho sohuwa 'a far place'    luuliyaa 'straight' → luule 7ogiyaa 'a straight road'

The ending -uwa of an adjective will become -o when it is employed in the predicative position. However, not every word that comes before a noun is an adjective. For instance, the demonstrative determiner Ha plays the role of an adjective in the phrase Ha dorsay taga, "This sheep is mine."

**C. Verbs:** Similar to the majority of Ethiopian languages, Wolaytta has an extremely sophisticated verbal system. Most verbs have a consonant-vowel-consonant pattern [51].

For example: Gela - 'enter'

Mooga - 'bury'

Kera - 'split'

According to mood, tense, type of action, and aspect, Wolaytta verbs display a fairly complicated inflection system [6]. Take the verb *imma* (give), for instance. Its past tense inflection is as follows:

Imm-asi --- I gave

Imm-dasa – you gave

Imm-su – she gave

Imm-isi ---- he gave

Imm-ida --- we gave

Imm-ideta – you gave (plural)

Imm-idosona – they gave

Similar to the Amharic language, Wolaytta verbs are placed at the conclusion of phrases, and the sentences below use suffix-bound morphemes to assist identify the sentence's subject.

Tani 7osuwa wursasi 'I finished my job'

Verbs include *shamasu*, *madasa*, and *wursasi* in the three sentences above. The bound morphemes "-su," "-dasa," and "-si" indicate that the third person has a feminine tag and that the second and first-person pronouns are employed as the sentence's subjects. The Wolaytta language uses suffixes to alter the form of verbs at any moment for person, gender, and number [6] [52] [54].

### 3.3.2 Derivational Morphology in Wolaytta

**A. Verbs:** Wolaytta uses several morphemes, like other Cushitic and Semitic languages of Ethiopia, to create additional stems from roots or stems. In Wolaytta, this derivation process is explicitly used by appending one or more morphemes to the verbal stem [6] [54]. They also mentioned that there are three main types of verbal stems that can be formed from secondary sources: iterative (or intensive), causative, and passive (or reflexive) [52] [54]. The Wolaytta language uses the same morpheme, *-erett-*, which is frequently suffixed to its verbal stem, to signify iteratives and intensive stems. Examples are "to break" and "to break many times or in numerous pieces," respectively. *Shishh-erett* means "to gather many times or in many things" and means to collect something. Only the morpheme *-is-s*, which is productive and causative, is used in Wolaytta.

gel- 'enter' → gel-iss- 'let someone enter/put into'.

When creating the causative form of a verb with a primary stem ending in -y or -y-y, all instances of -y are replaced with -sh-.

**Example:** uy-y- 'drink' → ush-sh- 'let someone drink.

yuuy-y- 'turn, intr' → yuush-sh- 'turn, tr

**B. Noun:** Wolaytta nouns are created by combining the class suffixes -a and -uwa [6] [52] [54], they can be used to convey action nouns as well as very concrete or abstract words as shown in the following table.

*Table 3.7: Derivational Morphology Nouns*

<b>Noun</b>	<b>Suffix</b>	<b>Derived Noun</b>
hassay- 'speak'	-a	Hassaya/conversation
harg- 'sick'	-ya	harg-iya 'sickness'
gulba- 'knee'	-ta	gulba-ta 'knee'
wurse- 'end'	-tta	wurse-tta 'end'
eeyya- 'stupidity'	-tetta	eeyya-tetta 'stupidity'
kaawo-tetta- 'kingome'	-tetta	kaawo-tetta 'kingdom'

**C. Adjectives:** Adjectives in the Wolaytta language can be created from verbal roots by adding morphemes like -ta. Imma+ -ta imota, as an illustration. Additionally, it can come from nouns, stems created by adding bound morphemes, and compound words.

**D. Conjunctions:** The word conjunction is used to join words together into phrases, clauses, and sentences as well as to coordinate words. There are various terms that are used as conjunctions in Wolaytta. Below is a list of some Wolaytta terms for conjunctions [10].

Table 3.8: List of conjunctions

English Conjunction	Wolaytta Conjunction
And	Nne
Or	Woikko
So, Therefore	Hegaa gishshau
For	Aissi giikko
But	Shin
Because	Gishsha (do gishsha)
Even if	Hanikkokka
Whenever	Awudekka
Wherever	Awannikka

### 3.4 Word Formation in Wolaytta

Words in Wolaytta can be created through compounding and affixation. Affixes are morphemes that can function by themselves as a word. Out of the three affix types (prefix, suffix, and infix), Suffixation is the only method of word creation used by Wolaytta. In Wolaytta, a single word can take on numerous forms. This is true because Wolaytta words are relatively long due to the repeated addition of suffixes [6].

**Example:** fayda ‘count’           → fayduwa ‘number’  
                   Be’’a ‘see’               → be’’uwa ‘action of seeing’  
                   Na’’a ‘boy/son’       → na’’iyo ‘girl/daughter’

Compounding is the second step in Wolaytta word construction. Combining two language forms with distinct functions is known as compounding. Despite Wolaytta abundance in compounds, compound morphemes are uncommon there and their generation is erratic [52]. As a result, it might be challenging to identify the root of the compounds that form words.



### 3.5 Wolaytta Language Writing System

A Tagmemic Analysis of the Wolaytta Language by B. Adams states that the Wolaytta language employs a Latin-based alphabet with twenty-nine basic letters, of which five ('i', 'e', 'a', 'o', and 'u') are vowels, twenty-four ('b', 'c', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'm', 'n', 'p', 'q', 'r', 's', 't', 'v', 'w', 'x', 'y', 'z' and '7'). In addition, seven pair letters which are a combination of two consonant characters such as ('ch', 'dh', 'ny', 'ph', 'sh', 'ts' and 'zh'). The Latin alphabet has now been included into the mother tongue educational system, and numerous textbooks are now produced in Latin [52].

*Table 3.9: Wolaytta Language Alphabet Letters*

Latiine		Latiine	
Woggaa	Qeerra	Woggaa	Qeerra
A	A	R	r
B	B	S	S
C	C	T	T
D	D	U	U
E	E	V	V
F	F	W	W
G	G	X	X
H	H	Y	Y
I	I	Z	Z
J	J	CH	Ch
K	K	DH	Dh
L	L	NY	Ny
M	M	PH	Ph
N	N	SH	Sh
O	O	TS	Ts
P	P	ZH	Zh
Q	Q	7	7

### **3.6 Punctuation Marks in Wolaytta Language**

Similar to how it is done in English, white spaces are used to separate words in Wolaytta writings. The punctuation used in English and other languages that employ the Latin writing system is the same throughout all punctuation marks. For instance, the full stop (.) in a statement, the question mark (?) in an interrogative sentence, and the exclamation mark (!) in a command or exclamatory sentence all serve as markers for the conclusion of a sentence.

However, there are some apostrophe exceptions. Some books use the apostrophe in the alphabet at glottal place 7. For instance, "lo77o" uses two apostrophes, "lo"o," whereas "de7iya" uses one, "de'iya."

### **3.7 Wolaytta Language Sentence Structure**

The fundamental sentence structures, or SVO, SOV, OSV, OVS, VSO, and VOS, are all feasible depending on the locations of the Subject, Object, and Verb, or their permutations. The vast majority of Afro-Asian languages have SOV word order. Consider Amaharic and Afan Oromo. Since Wolaytta is an Omotic family language, it uses the SOV grammatical structure found in the Afro-Asian language phylum as opposed to English's SVO (subject verb object) sentence structure.

"Addaami kattaa immiis," for instance, is a statement in the Wolaytta language. A subject is "Addaami," an object is "Kattaa," and a verb is "Immiis." The sentence structure in English is subject-verb-object. For instance, the English translation of the Wolaytta line above would be "Adam given food," where Adam is the subject, "given" is the verb, and food is the object [10].

## CHAPTER FOUR

### PROPOSED ARCHITECTURE AND RESEARCH METHODOLOGY

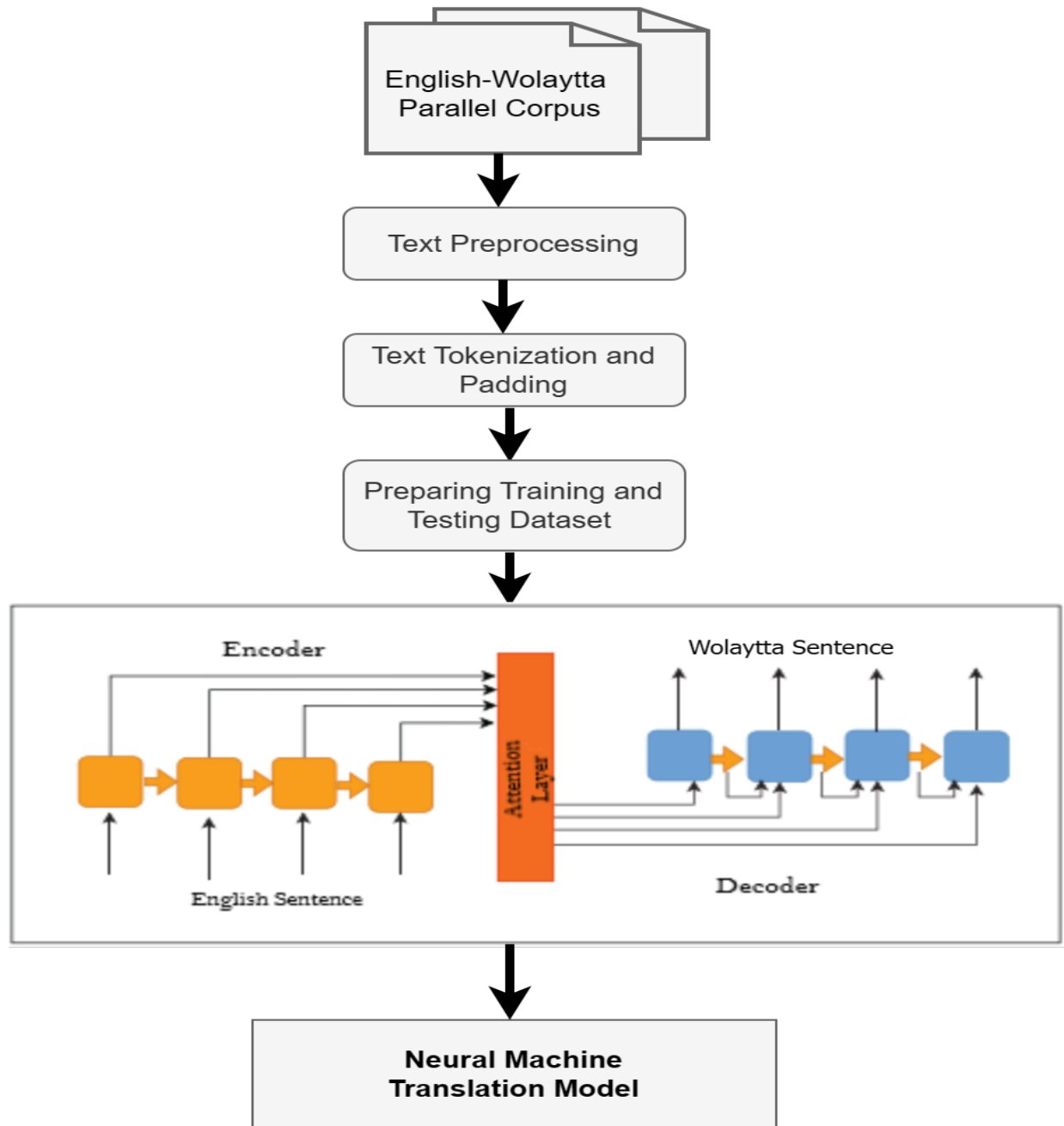
#### 4.1. Introduction

The objective of this study is to develop an English-Wolaytta neural machine translation with attention mechanism. In order to achieve the objective appropriate approaches were studied and chosen. Parallel English-Wolaytta corpus were collected and prepared in the way to be used for NMT model. This chapter briefly discusses the proposed architecture, how the study has been made, the tools and techniques that have been used for development of English-Wolaytta NMT model.

#### 4.2. Proposed Architecture of English-Wolaytta NMT

Neural machine translation is the state-of-the-art in machine translation that surpasses all existing machine translation methods. It uses word vector representation that utilizes numerous amounts of neural networks to predict the probability of word sequence. It translates sentences at once, that other translation approaches could not do. In NMT the input sentence passes through the encoder that designates the meaning of the input sentences, known as “thought vector/context vector” or a sentence vector, which passes through a decoder that process the input to provide a translation. This model is known as an encoder-decoder architecture [13].

Just as there is a successive change in other translation approaches, through time, there are also a promising improvement in neural machine translations. Initially it was implemented by sequence to sequence RNNs methods like LSTM, GRU and CNN. Later, attention mechanism added. Now it has been brought forth with supper improvements “Transformer with attention”. In this study encoder-decoder/sequence to sequence (seq2seq) with attention mechanism is implemented to develop the English-Wolaytta NMT. The following Figure 4.1 shows the proposed architecture of the study.



*Figure 4.1: Proposed Architecture for English-Wolaytta NMT*

While working with NMT, there must have parallel corpus that will be used to train and test the NMT model. The corpora that are pre-processed must also be split into the training, validation and test data sets. From these data the source and target validation data; the source and target train data are used to preprocess the first phase. The training steps takes the training and validation files to create the models for the translation step. This is where the best performing NMT architecture is

chosen and tuned with an appropriate parameter like batch size, epoch and so on. At the end of the training, there will be certain models created by which the translation will take place. The model created here, together with source test data set predicts an appropriate target translation. The predicted target translation is tested with target test data set (reference) to check the performance of the proposed model. The following section and subsection discuss all the steps and the methods followed to build the proposed English-Wolaytta NMT model.

### **4.3. Corpus Collection and Preparation**

To create an effective translation model, machine translation research heavily relies on parallel corpora of the source and target languages. Hence, the first step of any machine translation task is collecting a source and target language parallel corpus. The corpus which was used in this research study was collected and prepared from religious domain and some common words which are used in daily communication purpose. The religion domain corpus mainly constitutes bible contents which is extracted from the [online source](#)<sup>1</sup> for both English and Wolaytta language of bible scriptures. A total of 27351 parallel English-Wolaytta sentences were collected.

After collecting the corpus of the source (English) and target (Wolaytta) languages, the next task was preparing the parallel corpus manually for both languages because of the scarcity of parallel corpus. All of the data in the collected corpus was subsequently converted to plain text, cleaned up from the blank lines and noisy characters, and its encoding was converted to UTF-8 automatically to make it ready for training of the system. There is a number of preprocessing activities which were carried out to get a cleaned corpus and prepared it in a format that needs to be applicable for the system. These preprocessing includes lower-case, cleaning, normalization, tokenization, and padding. The following Figure 4.2 shows sample collected English-Wolaytta corpus.

---

<sup>1</sup> <https://ebible.org/bible/>

```
lang = pd.read_table("eng-wol.txt", names = ['English', 'Wolaytta'])
#printing sample data from lines
lang.sample(10)
```

	English	Wolaytta
15709	And it came to pass, when their hearts were me...	Eti keehippe ufaittido gishshau, "Kaa7idi nuna...
16101	Now therefore up by night, thou and the people...	Hegaa gishshau, neeninne nenaara de7iya asai q...
1075	I would that ye all spake with tongues, but ra...	Intte ubbai dumma dumma qaalan haasayanaagaa t...
3334	This [is] a faithful saying and worthy of all ...	Ha haasayai asai ubbai ekkanaa bessiyi tumu ha...
11345	Then said he unto me, The north chambers [and]...	Hegaappe guyyiyan, he bitanee tana hagaadan ya...
21126	And the flesh that toucheth any unclean [thing]...	"Tunabaa aibanne bochchida ashoi meetettoppo;...
24892	[Seemeth it but] a small thing unto you, that ...	Israa7eela Xoossai inttena Israa7eela maabaraa...
17625	Why doth thine heart carry thee away? and what...	Ne wozanai nena aissi wora efii? Ne aifeekka a...
19385	So when they continued asking him, he lifted u...	Eti a oichchidi aggennan ixxin, pude xooqqu gii...
15929	And the LORD discomfited Sisera, and all [his]...	GODAI Sisaara, a paraa gaareta ubbaanne a olan...

Figure 4.2: Sample English-Wolaytta Parallel Corpus

#### 4.4. Text Preprocessing

After preparing the manually collected English-Wolaytta corpus, the text preprocessing task has been performed before using it on the neural network machine translation model development. There are different activities which are undertaken in text preprocessing to prepare a cleaned corpus that helps to build an efficient translation model. These activities are described as follows in the following subsections.

##### 4.4.1. Normalization

Normalization is a method used to clean noise from unstructured text or sentence [55]. In this study, normalization is used to convert the English word or phrase with the same pronunciation and different writing representation to same word or phrase which are called contraction. In English language phrases like 'can't' and 'cannot', 'she's' and 'she is' represent the same meaning with different pronunciation and it can be used interchangeably without any meaning differences, therefore, those type of phrases or words are normalized by converting from its contraction/ short form of representation to its full representation. In this study, texts or phrases of the English language is performed by using the code in Appendix A.

#### **4.4.2. Data Cleaning**

The collected parallel English-Wolaytta corpus is not cleaned. Hence, the corpus needs to be cleaned by removing irrelevant characters, numbers, punctuation marks, blank values, special characters and so on. To preprocess and get a cleaned dataset from the prepared parallel corpus a python script is created by using a regular expression (RE) from NLTK to remove the unwanted characters, punctuation marks including spaces between paragraphs and numbers in the corpus. Appendix B demonstrates the implementation code used to clean the prepared parallel English-Wolaytta corpus.

#### **4.5. Tokenization and Padding**

Tokenization is a process in which a word or a sequence breaks up into keywords, words, symbols, etc. A word or a group of words may be used to do tokenization. In addition, a paragraph or sentence may be divided into phrases or words. Tokenization simply split strings or tokenize words in a sentence. The splitting can be performed by using space delimiter and it is performed in whole document [55]. In this study, word tokenization is applied to split the English and Wolaytta sentences into words to better use them for further machine translation activities. In order to find the boundaries of the sentence, <start> and <end> tag was added as an indicator of the start and end of a sentence so that the model knows as it reached the start and end of words in input sentence and output sentence.

Padding is a way of making each sequence of word ids in the same length in order to batch them together. Padding can be added at the end of the sequences to make them the same length because sentences can vary in length. This type of data is given to the encoder component, which creates a contextual association between words. When a sequence contains a longer sentence, an improvement method is required to aid in the storage of the longer phrase's context. In order to fix the length of a text, padding is the process of taking a value and adding pad characters to the left, right, or both sides of the string [56]. We did that by using a built-in function called `tf.keras.preprocessing.sequence.pad_sequences([inputs])`, which results in padding being added to the end of each English and Wolaytta sequence to make them all the same length. The implementation code is demonstrated inside preprocess function in Appendix B.

## 4.6. Word Representation

The original sequential data in NMT is not adequate for the neural network to read. It requires representation in a suitable format in some way. The input data must be numerical since a neural network consists of a sequence of addition and multiplication operations. Each word of the sentence in the data must be recognized and represented by a distinct index in order to transform it into appropriate format. One method of transforming each unique word into individual numbers is known as one-hot representation. To transform vocabulary words into a representation of a specific id, we used a built-in technique called `index_word` method in our study.

After identifying the unique tokens, the initial index is represented by 0 (zero), and the last index is represented by the sum of all the unique tokens minus one. Hence, only the words given in vector representation will be able to be processed by the neural network that uses this vocabulary as showed in Figure 4.3.

```
#creating a dictionary for index to word for English and Wolaytta vocabulary
english_word2idx= dict([(i, word) for word, i in english_word2idx.items()])
print(english_word2idx)
wolaytta_word2idx =dict([(i, word) for word, i in wolaytta_word2idx.items()])

{'<END>': 1, '<START>': 2, 'a': 3, 'aaron': 4, 'aaronites': 5, 'aaron's': 6, 'abagtha': 7, 'abana': 8, 'abanim': 9, 'abase': 10, 'abased': 11, 'abasing': 12, 'abated': 13, 'abba': 14, 'abda': 15, 'abdeel': 16, 'abdi': 17, 'abdiel': 18, 'abdon': 19, 'abednego': 20, 'abel': 21, 'abelbethmaachah': 22, 'abelmaim': 23, 'abelmeholah': 24, 'abelmizraim': 25, 'abelshittim': 26, 'ab ez': 27, 'abhor': 28, 'abhorred': 29, 'abhorrest': 30, 'abhorreth': 31, 'abhorring': 32, 'abi': 33, 'abia': 34, 'abiah': 35, 'abialbon': 36, 'abiasaph': 37, 'abiathar': 38, 'abiathars': 39, 'abib': 40, 'abida': 41, 'abidah': 42, 'abidan': 43, 'abid e': 44, 'abideth': 45, 'abiding': 46, 'abiel': 47, 'abiezer': 48, 'abiezrite': 49, 'abiezrites': 50, 'abigail': 51, 'abihai l': 52, 'abihu': 53, 'abihud': 54, 'abijah': 55, 'abijam': 56, 'abilene': 57, 'ability': 58, 'abimael': 59, 'abimelech': 60, 'abimelechs': 61, 'abinadab': 62, 'abinoam': 63, 'abiram': 64, 'abishag': 65, 'abishai': 66, 'abishalom': 67, 'abishua': 68, 'abishur': 69, 'abital': 70, 'abitub': 71, 'abiud': 72, 'able': 73, 'abner': 74, 'abners': 75, 'aboard': 76, 'abode': 77, 'ab odest': 78, 'abolish': 79, 'abolished': 80, 'abominable': 81, 'abominably': 82, 'abomination': 83, 'abominations': 84, 'aboun d': 85, 'abounded': 86, 'aboundeth': 87, 'abounding': 88, 'about': 89, 'above': 90, 'abraham': 91, 'abrahams': 92, 'abram': 9
```

Figure 4.3: Sample word2index Representation

## 4.7. Data Splitting for Model Training and Testing

Once the prepared corpus is preprocessed, the preprocessed data are used to train and test the neural network machine translation model. To do this, the prepared corpus should be classified as source train, source validation, source test, target train, target validation and target test. In this study, 80/20 corpus splitting were applied. The 20 percent testing data has also further split into validation set.



## 4.8. Encoder

Encoder is one component of NMT architecture that accepts a single element as an input sequence, processes it, gathers data about the element, and propagates it [32]. When reading a phrase with  $m$  words or an input sequence of length  $L$ , it will take  $L$  time steps because it only reads one element or word at a time. A thought vector or context vector representing the meaning of the source language must be created by the encoder. Following are some examples of notations used in the encoding process: The RNN's internal states at time step  $t$  are  $h_t$  and  $c_t$ ; the input at step  $t$  is  $x_t$ ; and the output at step  $t$  is  $y_t$ .

Take the question, "What is your name?" as an example. This string of letters can be thought of as a four-word statement. Here, the letters  $x_1$  stand for "What,"  $x_2$  for "is,"  $x_3$  for "your," and  $x_4$  for "name." Four-time steps will be read from this sequence, as shown in figure below. When the time  $t = 1$  comes around, it remembers that the RNN cell has read "What," when the time  $t = 2$  comes around, it remembers that the RNN has read "What is," and when the time  $t = 4$  comes around, the final states  $h_4$  and  $c_4$  remember the full sequence "What is your name." Initial vectors for the initial states  $h_0$  and  $c_0$  are zero. The encoder determines the thought vector using a list of words as input.

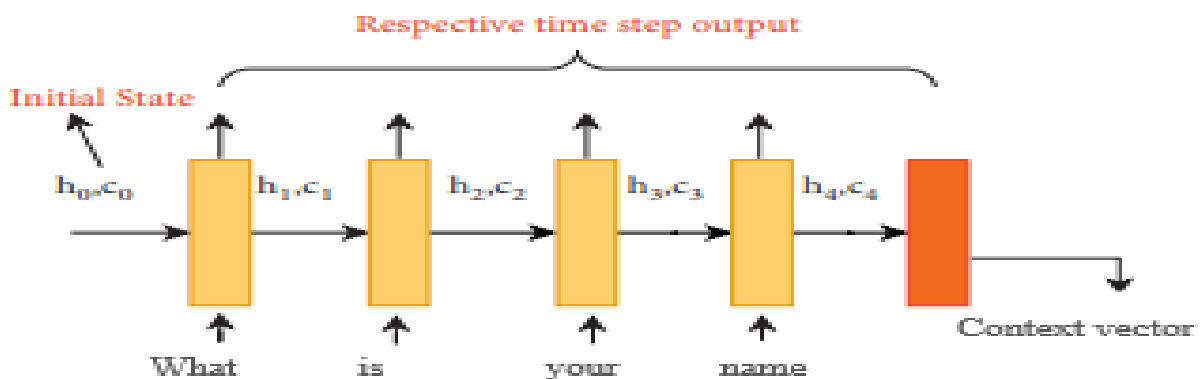


Figure 4.4: LSTM Encoder

A sentence from a specified source language is transformed into a thought vector using a high-dimensional vector of real numbers or components. Concisely representing the source language sentence and choosing how to initialize the initial states of an encoder with zeros are the major goals of the context vector ( $v$ ). The context vectors serve as the decoder's initial state. The context vector is used as the beginning state by the RNN decoder instead of starting with zero.

#### **4.9. Decoder**

The decoder is another component of NMT. The decoder's purpose is to convert the context vector into the desired translation [32]. The most recent input data and the system's previous outputs, the encoder produces a context vector as its final output. This context vector will be sent into the decoder either with or without the use of the attention-based mechanisms. The last layer of the encoder network will be connected to the attention layer of the network if the attention mechanism is used [32]. RNN network serves as the encoder and decoder. In our model, we have used one of the special kinds of RNN network called LSTM since it is most familiar for both Encoder and Decoder part of NMT on text-based translation [57].

#### **4.10. Attention Mechanism**

One of the major advances in machine translation that helped the neural machine translation systems is the attention mechanism [58]. As discussed in previous subsection in a basic Encoder-Decoder based RNN architecture, the encoder encodes the entire source language sequences of data into a single real-valued vector which is the context vector, and then passes it to the decoder to create the output sequence. Consequently, the Decoder only has access to the Encoder's output context vector. Because of this, representing the full input sequence as a single vector is inefficient and unable to represent for increasingly complex sentences and expansive vocabulary. The attention mechanism, which has lately gained prominence in neural network training, is one practical approach to solving such an issue. Encoder-Decoder architecture with attention mechanisms predicts a target word based on the context vectors associated with the source language position and the previously generated target words, as opposed to Encoder-Decoder architecture without attention, which uses the source representation only once to initialize the decoder hidden state.

By incorporating an attention mechanism, we were able to resolve the fundamental Encoder-Decoder difficulty. An attention mechanism to the Encoder-Decoder reduces the higher dimension vector into the lower dimension vector, to address the increasing number of vocabulary size problems of the basic Encoder-Decoder based technique [25]. It is helpful to focus on the source sequence's most pertinent information. So, it is helpful to focus only on the relevant information other than taking whole information at once. Self-attention is represented by the following formula.

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V \quad (4.1)$$

Where  $Q$  represents the matrix that contains the query result,  $K$  represents all keys, and  $V$  are values. Each symbol represents the vector representations of all words in the sequence [59].

#### **4.11. Model Evaluation Metrics**

Various techniques have been proposed to evaluate any machine translations accuracy of translation performance. Bilingual evaluation understudy (BLEU), which compares the system's output with reference sentences that have been translated by humans, is one of these metrics. The core concept of BLEU is the measurements of this proximity. A high-quality translation is one that is more similar to a competent human translation. The BLEU score value is between 0 and 1. Most often, it is expressed as a percentage from 0 to 100. The more similar the translation is to a human translation, the higher the BLEU score [60].

## CHAPTER FIVE

### EXPERIMENTAL RESULT AND DISCUSSION

#### 5.1. Introduction

In this chapter, the experimental result gained from the English-Wolaytta NMT model training and testing are presented in detail. Finally, the chapter concludes with a review of the study's findings and details on evaluation metrics.

#### 5.2. English-Wolaytta NMT Model Building and Training

For building the proposed English-Wolaytta NMT model LSTM encoder and decoder architecture with an attention mechanism has been proposed in the Sequence-to-Sequence concept. Simple RNNs have a hidden state where they keep the details of the data they have seen up to that point. However, simple RNNs cannot preserve long-term dependencies because they have vanishing gradient difficulties for extended sequences as we have seen in the literature part [61]. Long-term dependencies, which are frequently present in time series data, are preserved by LSTMs. Given that a word toward the end of the sentence may frequently be extremely dependent upon a word toward the beginning of the sentence, this is a highly desired quality in a MT model. The LSTM architecture was chosen to carry out the work of NMT in this research due to its numerous benefits.

For better understanding of the efficiency of attention mechanism we have employed both attention and non-attention mechanism to compare their result. We have also trained the model numerous times by adjusting the number of hyper parameters, including the epoch, batch size, optimization functions and learning rate. Thus, the model is trained with parallel English-Wolaytta corpus by using batch size of 100 and 128, epoch number of 5, 10, 15, 30, 38 and 50 and two optimization function which are Adam and RMSProp with learning rate of 0.1 and 0.01. The following subsection describes the result gained from the experiment.

##### 5.2.1. English-Wolaytta NMT using Non-attention Mechanism

In this experiment, the English-Wolaytta NMT model is built with an Encoder-Decoder architecture without attention mechanism and trained with the training corpus set. As we have discussed earlier different hyper parameter values are used interchangeably to check their effect on the performance of the English-Wolaytta NMT translation accuracy. By setting the different

parameters, the best performing hyper parameter combination are recorded. Hence, Table 5.1 shows top three best results gained from the encoder- decoder without attention mechanism based on the different hyper parameter setups.

*Table 5.1: Best Results using Encode-Decoder without Attention Mechanism*

<b>Rank</b>	<b>Epoch</b>	<b>Batch Size</b>	<b>Optimization Function</b>	<b>Learning Rate</b>	<b>BLUE Score</b>
<b>1</b>	50	100	Adam	0.01	2.35
<b>2</b>	5	128	Adam	0.01	1.78
<b>3</b>	15	100	Adam	0.01	0.63

### **5.2.2. English-Wolaytta NMT using Attention Mechanism**

In this experiment, the English-Wolaytta NMT model is built with an Encoder-Decoder architecture with attention mechanism and trained with the training corpus set like we did in the non-attention. Table 4.2 illustrate top three best results gained from the encoder- decoder with attention mechanism English-Wolaytta NMT using different hyper parameter settings.

*Table 5.2: Best Results using Encode-Decoder with Attention Mechanism*

<b>Rank</b>	<b>Epoch</b>	<b>Batch Size</b>	<b>Optimization Function</b>	<b>Learning Rate</b>	<b>BLUE Score</b>
<b>1</b>	15	128	Adam	0.01	5.16
<b>2</b>	50	100	Adam	0.1	4.59
<b>3</b>	10	100	RMSProp	0.01	3.14

Based on the findings of the above different experiments, English-Wolaytta NMT built with attention mechanism has got a better result than non-attention. This is because Encoder-Decoder without attention mechanisms cannot handle large number of sentences. As the length of the sentence increases, the inter-dependency of words is loosely related and unable to handle a large number of vocabulary sizes. In addition to that, we have seen that the performance of the NMT model depends on the hyper parameter settings like batch size, epoch, and optimization function and so on. The Encoder-Decoder model with LSTM are tuned with these hyper parameters in the translation as seen in the above sections, and accordingly the BLEU score shown above confirms

right. All the above model has shown that Encoder-Decoder model with attention mechanism can achieve better translation accuracy.

### 5.3. English-Wolaytta NMT Testing and Translation

After the training step is over at some points the succeeding step is to run a model test and start the encoder and decoder processes. By providing the data to the encoder and receiving the output at the decoder, we can make predictions. For the neural architecture, translation is done in a step-by-step, end-to-end process which predicts the possible best translations taking a model with test data and it gives best possible translations. The best possible translation predicted is used to evaluate the performance of that model. This translation attempt is somehow close to Wolaytta sentence but it lacks the exact translation because of the data size. The smaller is the data size the poor is the result. The following figure illustrates the translation result gained from the experiment.

```
translate(model_attention)
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 3ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
26/26 [=====] - 0s 4ms/step
Model BLEU score: 4.59
example:1
Source sentence: wherefore they are no more twain but one flesh what therefore god hath joined together let not man put asunder
Predicted translation: heгаа gishshau eti naa77u qofan issuwaa issuwaa gidiyo gishshau attumaasa xayidi ainne
Actual translation: heгаа gishshau eti issi asa gidiyoogaappe attin naa77u asa gidokkona heгаа gishshau xoossai issippe asi yaagiis
```

Figure 5.1: English-Wolaytta NMT Translation Result

#### 5.4. Discussion of the Result

The general objective of this thesis work is to develop English-Wolaytta machine translation model using deep learning neural machine translation approach. For achieving this, different preprocessing steps were done. After that the model is built and trained with Encoder-Decoder architecture with and without attention mechanism using LSTM. Based on our experimental findings, better outcome is obtained when an attention mechanism is used instead of a non-attention mechanism. However, it is difficult to obtain a translation model that is more effective given the short amount of data employed in this study. Even if a small dataset is used, the maximum BLEU score of 5.16 and accuracy of 88.65 is recorded by playing with this small corpus and changing various hyper parameters. From our experimental finding we have seen that NMT has become a successful machine translation technique as a result of increased processing capacity. Sentences can be translated by NMT with remarkable accuracy using the Encoder-Decoder architecture. Machine translation is also becoming more effective with the addition of attention mechanisms which solves the problem of Encoder-Decoder model to handle a larger number of vocabulary sizes available within the data.

One can see that the English-Wolaytta translation result from our experimental result score demonstrates that the attention-based approach is better to the non-attention-based approach. However, the outcomes from both experiments were only based on smaller corpora. The accuracy improves together with the corpus size, and the BLEU score result can likewise increase proportionally.

Our experimental findings indicate that English-Wolaytta non-attention-based NMT has interesting translation results, with a BLEU score of 2.35 and accuracy of 86.91 in relation to our small datasets. Additionally, utilizing an attention-based approach has greatly improved our results, with a 2.81 BLEU improvement over a non-attention-based system. Additionally, when we looked at training time efficiency, the attention-based system outperformed the non-attention-based system. Additionally, when compared to non-attention-based systems, attention-based systems can store longer contextual relevancies of terms found in larger sentences of datasets.

Therefore, in this research work an attempt has been made to answer the research questions set out at the beginning of the research. The first research question was “What will be the performance of

attention-based NMT approach to English-Wolaytta language pairs?” It is answered by doing different experiments with limited resource and the best result we have got from the various experiments has achieved BLUE score of 5.16. The second research question which is “How well do attention-based NMT models perform in translation tasks?” is answered by implementing both attention and non-attention mechanism in order to evaluate the efficiency of the attention mechanism in the translation task and we have proved that it has a better translation accuracy than the non-attention based model.



## CHAPTER SIX

### CONCLUSION AND FUTURE WORKS

#### 6.1. Introduction

In this chapter, the researcher concludes the overall work of the study and provides recommendation for other problems to be investigated by summarizing the research, research findings, and significant contributions of the study. It also highlights the outcomes that resulted from the research experiments as well as the work that will need to be done in the future by anyone or any group interested in working on tasks connected to machine translation between the English and Wolaytta language pairs or any other language pairs.

#### 6.2. Conclusion

The design and implementation of an English-Wolaytta Neural Machine Translation (MT) system is the main emphasis of this research work. A highly important use of natural language processing is machine translation, which uses devices like computers to automatically translate vast amounts of text from a source natural language to a destination natural language. Machine translation is now a particularly difficult research topic in the fields of computational linguistics and natural language processing on a global scale. The employment of a multilingual machine translation system as a language learning aid can help one become familiar with both known and undiscovered natural languages. Natural language is a crucial component of human life. One of the most significant and efficient forms of communication is natural language. It helps in connecting individuals from other communities and civilizations. Even though Wolaytta is one of a recognized language in Ethiopia, there are relatively few electronic resources available for it and there no much MT systems for the Wolaytta language. Therefore, the researcher encouraged to prepare English-Wolaytta parallel text corpus and the development of an English-Wolaytta machine translation system.

For comparing the translation outcomes of the proposed English-Wolaytta MT system, the Neural Machine Translation (NMT) which is the state-of-art approach has been used to develop the English-Wolaytta MT system. For doing so, a Deep Learning Encoder-Decoder (seq2seq) with and without attention model for translating from English-Wolaytta was proposed. The parallel

English Wolaytta parallel corpus was prepared from some common sentences and religious domain to train the English-Wolaytta NMT system. The parallel English-Wolaytta corpus of 56900 tokens has been used, and out of these total tokens, 12,120 were English tokens, and 44,780 Wolaytta tokens. This corpus was used to train and test the seq2seq with attention and non-attention mechanism by using LSTM and the translation accuracy of the English-Wolaytta NMT system has been assessed using the BLEU score. After extensive experimentation, the suggested system achieves a BLEU score of 5.16 and 88.91 accuracy.

### **6.3. Future Works**

The followings are some further future works or recommendations based on the findings of this study:

- Recommend to train the NMT model with a powerful processing machines by adding more datasets to improve the translation quality since neural networks require more data and powerful processing machines to function more effectively and get accurate translation.
- Recommend to add and expand the corpus from various domains including business, tourism, health, entertainment and so on to make it work for most purposes.
- Recommend future researchers who are interested in this work to expand this system by making it bidirectional in the future so that resources written in both languages can be translated with ease.
- Recommend to increase the performance of the NMT model by using the most recently used architecture called transformer model.
- Recommend to develop and implementing the English-Wolaytta NMT system in web based or mobile application to make it more operational and useful for the users.
- Additionally, this research could be very beneficial for future study to work on speech to text or text to speech translation.

## References

- [1] G. G. Chowdhury, "Natural language processing," *Annual Review of Information Science and Technology*, pp. 51-89, 2003.
- [2] A. Karakanta, J. Dehdari and J. van Genabith, "Neural machine translation for low-resource languages without parallel corpora," *NLP in Low-Resource Languages*, vol. 32, no. 1, pp. 167-189, 2018.
- [3] M. Singh, R. Kumar and I. Chana, "Neural-Based Machine Translation System Outperforming Statistical Phrase-Based Machine Translation for Low-Resource Languages," in *Twelfth International Conference on Contemporary Computing (IC3)*, Noida, India, 2019.
- [4] W. J. Hutchins, "MACHINE TRANSLATION: A BRIEF HISTORY," in *Concise History of the Language Sciences*, Oxford: Pergamon, From the Sumerians to the Cognitivists, 1991, pp. 431-445.
- [5] N. Ashraf and M. Ahmad, "Machine Translation Techniques and their Comparative Study," *International journal of computer applications*, vol. 125, no. 7, pp. 25-31, September 2015.
- [6] L. L. FEREDÉ, *DEVELOPMENT OF STEMMING ALGORITHM FOR WOLAYTTA TEXT*, ADDIS ABABA: Masters Thesis, ADDIS ABABA UNIVERSITY, JULY, 2003.
- [7] M. L. Bender, J. D. Bowen, R. L. Cooper and F. C. A, *Language in Ethiopia*, London: Oxford University Press, 1976.
- [8] E. Commission, "BRTE Project," 1 November 2019. [Online]. Available: <http://brte.futureplan.ie/about/overview-wolaita/>. [Accessed 15 January 2022].
- [9] W. W. Gaga, *Attention-based Amharic-to-Wolaita Neural Machine Translation*, Addis Ababa: Masters Thesis, Addis Ababa University, October 2020.

- [10 M. Mara, *English-Wolaytta Machine Translation Using Statistical Approach*, Addis Ababa: Masters Thesis, Addis Ababa uNIVERSITY, July 2018.
- [11 T. M. Helana, "Bidirectional Dictionary Based Machine," *International Journal of Innovative Science and Research Technology*, vol. 5, no. 4, April – 2020 .
- [12 K. Firew, *A Hybrid Machine Translation System for English to Wolaytta Language*, March ] 2020.
- [13 Y. Wu, M. Schuster, Z. Chen and Q. V. Le, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," *arXiv:1609.08144v2 [cs.CL]*, October 2016.
- [14 J. Johnson, "Statista," 26 January 2020. [Online]. Available: ] <https://www.statista.com/statistics/262946/share-of-the-most-common-languages-on-the-internet/#:~:text=As%20of%20January%202020%2C%20English,percent%20of%20global%20internet%20users>. [Accessed 05 Feb 2022].
- [15 D. Arnold, L. Balkan and S. Meijer, *Machine Translation: an Introductory Guide*, UK: NCC ] Blackwell LTD, November 27, 2000.
- [16 M. Irfan, "Machine Translation," Bahria University, Islamabad, October 2017. ]
- [17 J. Daba, *Bidirectional English – Afaan Oromo Machine Translation Using Hybrid Approach*, ] Addis Ababa: Masters Thesis, Addis Ababa University, November 2013 .
- [18 M. D. Okpor, "Machine Translation Approaches: Issues and Challenges," *IJCSI International Journal of Computer Science*, vol. 5, no. 2, September 2014.
- [19 I. R. Trigueros, "Machine translation systems and quality assessment: a systematic review," ] *Language Resources and Evaluation* 56(2):1-27, pp. 1-27, June 2022.

- [20 S. Abdul Basit Andrabia and A. Wahid, "A Review of Machine Translation for South Asian  
] Low Resource Languages," *Turkish Journal of Computer and Mathematics Education*, vol. 12, no. 5, pp. 1134-1147, 5 April 2021.
- [21 R. Sinhal and K. Gupta, "Machine Translation Approaches and Design Aspects," *IOSR  
] Journal of Computer Engineering*, vol. 16, no. 1, pp. :22-25, January 2014.
- [22 A. P. J., "Machine Translation Approaches and Survey for Indian Languages," *The  
] Association for Computational Linguistics and Chinese Language Processing*, vol. 18, no. 1, pp. 47-78 , March 2013.
- [23 S. Dubey, "Survey of Machine Translation Techniques," *International Journal of Advance  
] Research in Computer Science and Management Studies*, vol. 5, no. 2, pp. 39-51, February 2017.
- [24 R. A. Sinhal and K. O. Gupta, "Machine Translation Approaches and Design Aspects," *IOSR  
] Journal of Computer Engineering (IOSR-JCE)*, , vol. 16, no. 1, p. Jan 2014, 25-28.
- [25 D. Bahdanau, K. Cho and Y. Bengio, "NEURAL MACHINE TRANSLATION BY  
] JOINTLY LEARNING TO ALIGN AND TRANSLATE," in *in Proceedings of the 3rd International Conference on Learning Representations (ICLR)*,, Montreal, 2015.
- [26 R. P. Neco and M. L. Forcada, "Asynchronous translations with recurrent neural nets," in *in  
] Computer Science Proceedings of International Conference on Neural Networks (ICNN'97)*, Alacant, Spain, June 1997.
- [27 Y. Bengio, R. Ducharme, P. Vincent and C. Janvin, "A neural probabilistic language model,"  
] *Journal of Machine Learning Research*, vol. 3, no. 1, p. 1137–1155, 2003.
- [28 N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," in *in  
] Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA, January 2013..

- [29 R. Pascanu, T. Mikolov and Y. Bengio, "On the difficulty of training Recurrent Neural Networks," in *In International Conference on Machine Learning*, Montreal, February 2013.
- [30 I. Sutskever, O. Vinyals and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *in Proceedings of the 27th International Conference on Neural Information Processing Systems*, December 2014..
- [31 K. Revanuru, K. Turlapaty and S. Rao, "Neural Machine Translation of Indian Languages," in *in COMPUTE 2017: 10th Annual ACM India Conference*, Bangalore, India, November 2017.
- [32 K. Cho, B. van Merriënboer and D. Bahda, "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches," in *in Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, Doha, Qatar, October 2014.
- [33 S. F. Chen and J. Goodman, *An Empirical Study of Smoothing Techniques for Language Modeling*, Cambridge, Massachusetts: Harvard University, August 1998.
- [34 Y. Kim, Y. Jernite, D. Sontag and Alexander , "Character-Aware Neural Language Models," in *in Proceedings of the AAAI Conference on Artificial Intelligence*, California USA, 2016.
- [35 S. Yang, Y. Wang and X. Chu, *A Survey of Deep Learning Techniques for Neural Machine Translation*, Hong Kong Baptist University, 18 Feb 2020.
- [36 S. S. Hosseini, *Recurrent VS Convolutional Neural Machine Translation: Translating Persian Verbal Inflections into English*, 2018.
- [37 C. C. Aggarwal, *Neural Networks and Deep Learning*, USA : IBM T.J Watson Research Center, 2018.
- [38 Y. LeCun, L. Bottou , Y. Bengio and a. P. H, "GradientBased Learning Applied to Document Recognition," in *On Proceedings of the IEEE*, 1998.

- [39 A. Vaswani, N. Shazeer and N. Parmar, "Attention is All you Need," in *in 31st Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 2017.
- [40 F. Stahlberg, "Neural Machine Translation: A Review and Survey," *Journal of Artificial Intelligence Research (JAIR)*, 4 December 2019.
- [41 A. Graves, *Generating Sequences With Recurrent Neural Networks*, arXiv:1308.0850, August 2013.
- [42 J. Chung, C. Gulcehre and K. Cho, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," in *in NIPS 2014 Workshop on Deep Learning*, December 2014.
- [43 S. Hochreiter and J. Schmidhuber, "LONG SHORT-TERM MEMORY," *Neural Computation*, vol. 9, no. 8, p. 1735–1780, 1997.
- [44 S. A. Zargar, *Introduction to Sequence Learning Models: RNN, LSTM, GRU*, North Carolina, April 2021.
- [45 J. Chung, C. Gulcehre, K. Cho and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS 2014 Workshop on Deep Learning*, December 2014.
- [46 T. Mengistu, "Bidirectional Dictionary Based Machine Translation for Wolayitegna-Amharic by Java," *International Journal of Innovative Science and Research Technology*, vol. 5, no. 4, April – 2020..
- [47 I. Gashaw and H. L. Shashirekha, *Amharic-Arabic Neural Machine Translation*, December 2019.
- [48 A. L. Tonja, M. M. Woldeyohannis and M. G. Yigezu, "A Parallel Corpora for bi-directional Neural Machine Translation for Low Resourced Ethiopian Languages," in *2021 International Conference on Information and Communication Technology for Development for Africa (ICT4DA)*, Bahir Dar, Ethiopia, November 2021.

- [49 A. Birhanu, *Bi-Directional English-Afan Oromo Machine Translation Using Convolutional Neural Network*, Addis Ababa, Ethiopia: Master's Thesis, Addis Ababa University, October 2019.
- [50 M. Hailegebreal, *A Bidirectional Tigrigna – English Statistical Machine Translation*, Addis Ababa, Ethiopia: Master's Thesis, Addis Ababa University, June, 2017.
- [51 M. WAKASSA, *A Descriptive Study of the Modern Wolaytta Language*, Japan: Doctoral Dissertation the University of Tokyo, May 2008.
- [52 M. Lamberti and R. Sottile, "The Wolaytta Language," *Journal of African languages and linguistics*, vol. 23, pp. 79 - 87, 2002.
- [53 F. a. H. C., "Non-Semitic languages: Cushitic and Omotic," *in Language in Ethiopia, London, Oxford Univ. Press*, pp. 34-53, 1976.
- [54 D. BELDADOS, *Automatic Thesaurus Construction from Wolaytta Text*, Addis Ababa: Master's Thesis, Addis Ababa University, JUNE 2013.
- [55 D. S. Vijayarani, M. J. Ilamathi and A. P. M. Phil, "Preprocessing Techniques for Text Mining.," *International Journal of Computer Science & Communication Networks*, 2015.
- [56 K. Mo, "Hands-on NLP Deep Learning Model Preparation in TensorFlow 2.X," 17 Aug 2020. [Online]. Available: <https://towardsdatascience.com/hands-on-nlp-deep-learning-model-preparation-in-tensorflow-2-x-2e8c9f3c7633>. [Accessed 2021 20 Dec].
- [57 G. Genthial, . L. Liu, . B. Oshri and . K. Ranjan., "Natural Language Processing with Deep Learning [Lecture notes]," 2019. [Online]. Available: [https://web.stanford.edu/class/cs224n/readings/cs224n2019-notes06-NMT\\_Seq2Seq\\_attention.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n2019-notes06-NMT_Seq2Seq_attention.pdf)..
- [58 "Analytics Vidhya," An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec, 19 October 2020. [Online]. Available: <https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/>. [Accessed 04 June 2022].



[59 Y. Liu a, L. Ji, R. Huang, T. Ming a and C. Gao, "An attention-gated convolutional neural network for sentence classification," *Intelligent Data Analysis journal*, vol. 3, pp. 1-19, 28 Dec 2018.

[60 Y. Zhang and S. Vogel , "Significance tests of automatic machine translation evaluation metrics," *Machine Translation*, vol. 24, no. 1, pp. 51-65, March 2010.

[61 M. Phi, "Medium," 24 Sep 2018. [Online]. Available:  
] <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>. [Accessed 16 June 2022].

## Appendix

### Appendix A: Normalization

```
def clean_text(text):
```

```
    """Clean text by removing unnecessary characters or contractions and altering the format of words."""
```

```
    text = text.lower()
```

```
    text = re.sub(r"i'm", "i am", text)
```

```
    text = re.sub(r"he's", "he is", text)
```

```
    text = re.sub(r"she's", "she is", text)
```

```
    text = re.sub(r"it's", "it is", text)
```

```
    text = re.sub(r"that's", "that is", text)
```

```
    text = re.sub(r"what's", "that is", text)
```

```
    text = re.sub(r"where's", "where is", text)
```

```
    text = re.sub(r"how's", "how is", text)
```

```
    text = re.sub(r"\ll", " will", text)
```

```
    text = re.sub(r"\ve", " have", text)
```

```
    text = re.sub(r"\re", " are", text)
```

```
    text = re.sub(r"\d", " would", text)
```

```
    text = re.sub(r"\re", " are", text)
```

```
    text = re.sub(r"won't", "will not", text)
```

```
    text = re.sub(r"can't", "cannot", text)
```

```
    text = re.sub(r"n't", " not", text)
```

```
text = re.sub(r"n", "ng", text)

text = re.sub(r"bout", "about", text)

text = re.sub(r"til", "until", text)

return text

lang.English=lang.English.apply(clean_text)
```

## **Appendix B: Implementation Code**

```
# **Attention Based English-Wolaytta Neural Machine Translation**

## Load important libraries

import re

import matplotlib.pyplot as plt

import time

import numpy as np

import tensorflow as tf

from keras.layers import Embedding,LSTM,Dropout,Dense,Layer

from keras import Model,Input

from keras.utils import plot_model

from keras_preprocessing.sequence import pad_sequences

from nltk.translate.bleu_score import corpus_bleu

from tensorflow.keras.optimizers import Adam

import collections

import keras.backend as K
```

```

## **Preprocess dataset**

source = "English.txt"

target = "Wolayita.txt"

def preprocessing(source,target, max_num_input=26000):

    # open and read source and target datasets

    source_data = open(source,'r',encoding='utf8').readlines()

    target_data = open(target,'r',encoding='utf8').readlines()

    # check if source and target datasets are equal

    assert len(source_data) == len(target_data)

    print("Length of source Language :",len(source_data))

    print("Length of target Language :",len(target_data))

    # limit input data

    if max_num_input > 0:

        max_num_input = min(len(source_data), max_num_input)

        source_data = source_data[:max_num_input]

        target_data = target_data[:max_num_input]

    # changing to lowercase,remove punctuation, strip trailing/leading whitespaces and tokenize
    each sentence.

    source_sen = [[re.sub('[\W]', "", str(token.lower() )) for token in sen.strip().split(' ')] for sen in
source_data]

    # target

    target_sen = [[re.sub('[\W+0-6+8-9]', "", str(token.lower() )) for token in sen.strip().split(' ')] for
sen in target_data]

```

```

# # remove punctuation

# target_sen = re.sub('[\W+0-6+8-9]', "", str(target_sen))

# for the target sentences, add <sos> and <eos> tokens to each sentence

for sent in target_sen:

    sent.append('<end>')

    sent.insert(0, '<start>')

# create the common_words objects for each file

source_dict = common_words(source_sen)

target_dict = common_words(target_sen)

# For the source sentences. we'll use this to split into train/dev/test

split_size = len(source_sen)//10

# get the sents-as-ids for each sentence

source_words = [[source_dict.word2ids.get(token,source_dict.UNK) for token in sen] for sen in
source_sen]

# Use 8 parts (80%) of the sentences for training and pad upto maximum sentence length

source_train = pad_sequences(source_words[:8*split_size], padding='post')

# Use 1 parts (10%) of the sentences for evaluation and pad up to maximum sentence length

source_eva = pad_sequences(source_words[8*split_size:9*split_size],padding='post')

# Use 1 parts (10%) of the sentences for test and pad upto maximum sentence length

source_test = pad_sequences(source_words[9*split_size:],padding='post')

eos = target_dict.word2ids['<end>']

# for each sentence, get the word index for the tokens from <start> to up to but not including
<end>,

```

```
target_words = [[target_dict.word2ids.get(tok,target_dict.UNK) for tok in sent[:-1]] for sent in target_sen]
```

```
# select the training set and pad the sentences
```

```
target_train = pad_sequences(target_words[:8*split_size],padding='post')
```

```
# the label for each target word is the next word after it
```

```
target_train_labels = [sent[1:]+[eos] for sent in target_words[:8*split_size]]
```

```
# pad the labels. Dim = [num_sents, max_sent_lenght]
```

```
target_train_labels = pad_sequences(target_train_labels,padding='post')
```

```
# expand dimensions Dim = [num_sents, max_sent_lenght, 1].
```

```
target_train_labels = np.expand_dims(target_train_labels,axis=2)
```

```
# get the labels for the dev and test data. No need for inputs here. no need to expand dimensions
```

```
target_eva_labels = pad_sequences([sent[1:] + [eos] for sent in target_words[8 * split_size:9 * split_size]], padding='post')
```

```
target_test_labels = pad_sequences([sent[1:] + [eos] for sent in target_words[9 * split_size:]], padding='post')
```

```
# we have our data.
```

```
train_data = [source_train,target_train,target_train_labels]
```

```
eva_data = [source_eva,target_eva_labels]
```

```
test_data = [source_test,target_test_labels]
```

```
return train_data,eva_data,test_data,source_dict,target_dict
```

```
## **common_words**
```

```
# Extract common words from dataset
```

```
class common_words():
```

```

def __init__(self, sents):

    word_counter = collections.Counter(tok.lower() for sent in sents for tok in sent)

    self.vocab = []

    self.vocab.append('<pad>') #zero paddings

    self.vocab.append('<unk>')

    # add only words that appear at least 4 times in the corpus

    self.vocab.extend([t for t,c in word_counter.items() if c > 10])

    self.word2ids = {w:id for id, w in enumerate(self.vocab)}

    self.ids2word = dict([(value, key) for (key, value) in self.word2ids.items()])

    self.UNK = self.word2ids['<unk>']

    self.PAD = self.word2ids['<pad>']

```

## Load the datasets

Let's load the datasets using the load function defined earlier.

```

train_data,eva_data,test_data,source_dict,target_dict = preprocessing(source,target,
max_num_input=26000)

```

let's now quickly check the data structure.

```

eva_data[0]

```

```

test_data

```

## **\*\*The Neural Translation Model (NMT)\*\***

For the NMT the network (a system of connected layers/models) used for training differs slightly from the network used for inference. Both use the seq-to-seq encoder-decoder architecture.

```

class AttentionLayer(Layer):

```

```

def compute_mask(self, inputs, mask=None):
    if mask == None:
        return None
    return mask[1]

def compute_output_shape(self, input_shape):
    return (input_shape[1][0],input_shape[1][1],input_shape[1][2]*2)

def call(self, inputs, mask=None):
    encoder_outputs, decoder_outputs = inputs
    decoder_outputs_transpose = K.permute_dimensions(decoder_outputs,pattern = (0,2,1))
    luong_score = K.batch_dot(encoder_outputs,decoder_outputs_transpose)
    luong_score = tf.nn.softmax(luong_score,axis = 1)
    encoder_vector = tf.math.multiply(tf.expand_dims(encoder_outputs,axis = -2) ,
    tf.expand_dims(luong_score,axis = -1) )
    encoder_vector = tf.reduce_sum(encoder_vector,axis=1)
    # [batch,max_dec,2*emb]
    new_decoder_outputs = K.concatenate([decoder_outputs, encoder_vector])
    return new_decoder_outputs

class seq2seqModel(object):
    def __init__(self,source_dict,target_dict,use_attention):
        # the number of hidden units used by the LSTM
        self.hidden_size = 200
        # the size of the word embeddings being used
        self.embedding_size = 100

```



```

# the dropout rate for the hidden layers
self.hidden_dropout_rate=0.2

# the dropout rate for the word embeddings
self.embedding_dropout_rate = 0.2

# batch size
self.batch_size = 100

# the maximum length of the target sentences
self.max_target_step = 100

# vocab size for source and target; we'll use everything we receive
self.vocab_target_size = len(target_dict.vocab)
self.vocab_source_size = len(source_dict.vocab)

# instances of the dictionaries
self.target_dict = target_dict
self.source_dict = source_dict

# special tokens to indicate sentence starts and ends.
self.SOS = target_dict.word2ids['<start>']
self.EOS = target_dict.word2ids['<end>']

# use attention or no
self.use_attention = use_attention

print("number of tokens in source: %d, number of tokens in target:%d" %
(self.vocab_source_size,self.vocab_target_size))

def build(self):
    #-----Train Models-----

```

```

source_words = Input(shape=(None,),dtype='int32')

target_words = Input(shape=(None,), dtype='int32')

# The train encoder

# Create two randomly initialized embedding lookups, one for the source, another for the
target.

# Creating the embedding lookups...

embeddings_source = Embedding(self.vocab_source_size,self.embedding_size)

embeddings_target = Embedding(self.vocab_target_size,self.embedding_size)

# Look up the embeddings for source words and for target words. Apply dropout to each
encoded input

# Looking up source and target words...'

source_word_embeddings = Dropout(0.3)(embeddings_source(source_words))

target_words_embeddings = Dropout(0.3)(embeddings_target(target_words))

# An encoder LSTM() with return sequences set to True

# Creating an encoder'

encoder_outputs,          encoder_state_h,          encoder_state_c          =
LSTM(self.hidden_size,recurrent_dropout=self.hidden_dropout_rate,return_sequences=True,retu
rn_state=True)(source_word_embeddings)

encoder_states = [encoder_state_h,encoder_state_c]

# The train decoder

decoder_lstm              =
LSTM(self.hidden_size,recurrent_dropout=self.hidden_dropout_rate,return_sequences=True,retu
rn_state=True)

```

```

        decoder_outputs_train,_,_ =
decoder_lstm(target_words_embeddings,initial_state=encoder_states)

if self.use_attention:

    decoder_attention = AttentionLayer()

    decoder_outputs_train = decoder_attention([encoder_outputs,decoder_outputs_train])

decoder_dense = Dense(self.vocab_target_size,activation='softmax')

decoder_outputs_train = decoder_dense(decoder_outputs_train)

# compiling the train model.

adam = Adam(learning_rate=0.01,clipnorm=5.0)

self.train_model = Model([source_words,target_words], decoder_outputs_train)

self.train_model.compile(optimizer=adam,loss='sparse_categorical_crossentropy',
metrics=['accuracy'])

# at this point you can print model summary for the train model

print('\n\t\t\t\t\t Train Model Summary.')

self.train_model.summary()

print(plot_model(self.train_model,      to_file='model_plot.png',      show_shapes=True,
show_layer_names=True))

#-----Inference Models-----

# The inference encoder

self.encoder_model =
Model(source_words,[encoder_outputs,encoder_state_h,encoder_state_c])

# at this point you can print the summary for the encoder model.

print('\n\t\t\t\t\t Inference Time Encoder Model Summary.')

```

```

self.encoder_model.summary()

# The decoder model

# specifying the inputs to the decoder

decoder_state_input_h = Input(shape=(self.hidden_size,))

decoder_state_input_c = Input(shape=(self.hidden_size,))

encoder_outputs_input = Input(shape=(None,self.hidden_size,))

# create decoder for inference

# Get the decoded outputs

# print("\n Putting together the decoder states')

# get the initial states for the decoder, decoder_states

# decoder states are the hidden and cell states from the training stage

decoder_states = [decoder_state_input_h,decoder_state_input_c]

# use decoder states as input to the decoder lstm to get the decoder outputs, h, and c for test
time inference

decoder_outputs_test,decoder_state_output_h,          decoder_state_output_c          =
decoder_lstm(target_words_embeddings, initial_state=decoder_states)

# Task 1 (b.) Add attention if attention

if self.use_attention:

    decoder_outputs_test = decoder_attention([encoder_outputs_input,decoder_outputs_test])

# Task 1 (c.) pass the decoder_outputs_test (with or without attention) to the decoder dense
layer

decoder_outputs_test = decoder_dense(decoder_outputs_test)

```

```

# put the model together

self.decoder_model =
Model([target_words,decoder_state_input_h,decoder_state_input_c,encoder_outputs_input],
      [decoder_outputs_test,decoder_state_output_h,decoder_state_output_c])

# you can now view the model summary

print("\t\t\t\t\t Decoder Inference Model summary')

print(self.decoder_model.summary())

def time_used(self, start_time):

    curr_time = time.time()

    used_time = curr_time-start_time

    m = used_time // 60

    s = used_time - 60 * m

    return "%d m %d s" % (m, s)

def train(self,train_data,dev_data,test_data, epochs):

    start_time = time.time()

    # for epoch in range(epochs):

    # print("Starting training epoch {}/{}".format(epoch + 1, epochs))

    epoch_time = time.time()

    source_words_train, target_words_train, target_words_train_labels = train_data
self.history=self.train_model.fit([source_words_train,target_words_train],target_words_train_labels,batch_size=self.batch_size,epochs=epochs)

    # print("Time used for epoch {}: {}".format(epoch + 1, self.time_used(epoch_time)))

    dev_time = time.time()

```

```

# print("Evaluating on dev set after epoch {}/{}".format(epoch + 1, epochs))

# self.eval(dev_data)

# print("Time used for evaluate on dev set: {}".format(self.time_used(dev_time)))

self.train_model.save('aeng_wol_model.h5')

self.encoder_model.save('aeng_wol_model_e.h5')

self.decoder_model.save('aeng_wol_model_d.h5')

print("Training finished!")

print("Time used for training: {}".format(self.time_used(start_time)))

print("Evaluating on test set:")

test_time = time.time()

self.eval(test_data)

print("Time used for evaluate on test set: {}".format(self.time_used(test_time)))

def get_target_sentences(self, sents, vocab, reference=False):

    str_sents = []

    num_sent, max_len = sents.shape

    for i in range(num_sent):

        str_sent = []

        for j in range(max_len):

            t = sents[i,j].item()

            if t == self.SOS:

                continue

            if t == self.EOS:

```

```

        break

    str_sent.append(vocab[t])

    if reference:

        str_sents.append([str_sent])

    else:

        str_sents.append(str_sent)

    return str_sents

def eval(self, dataset, print_outputs = False):

    # get the source words and target_word_labels for the eval dataset

    source_words, target_words = dataset

    vocab = self.target_dict.vocab

    # using the same encoding network used during training time, encode the training

    encoder_outputs, state_h, state_c =
self.encoder_model.predict(source_words, batch_size=self.batch_size)

    # for max_target_step steps, feed the step target words into the decoder.

    predictions = []

    step_target_words = np.ones([source_words.shape[0], 1]) * self.SOS

    for _ in range(self.max_target_step):

        step_decoder_outputs, state_h, state_c =
self.decoder_model.predict([step_target_words, state_h, state_c, encoder_outputs], batch_size=self.
batch_size)

        step_target_words = np.argmax(step_decoder_outputs, axis=2)

```

```

        predictions.append(step_target_words)

    # predictions is a [time_step x batch_size x 1] array. We use get_target_sentence() to recover
    the batch_size sentences

    predicted = self.get_target_sentences(np.concatenate(predictions,axis=1),vocab)

    Actual = self.get_target_sentences(target_words,vocab,reference=True)

    # score using nltk bleu scorer

    score = corpus_bleu(Actual,predicted)

    print("Model BLEU score: %.2f" % (score*100.0))

    #Modification

    if print_outputs:

        sources = self.get_target_sentences(np.array(source_words[0:len(source_words)]),self.source_dict.vocab)

        return sources, predicted, Actual

## **Training Without Attention**

### Architecture

#Clear session prior to creating the architecture

tf.keras.backend.clear_session()

model = seq2seqModel(source_dict, target_dict,False)

model.build()

### Training and test evaluation

model.train(train_data,eva_data,test_data,50)

### plo model history

model.history.history['loss']

```



```

model.history.history['accuracy']

model.history.history.keys()

model.history.history.keys()

# summarize history for accuracy

plt.plot(model.history.history['accuracy'])

# plt.plot(model.history.history['val_accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# summarize history for loss

plt.plot(model.history.history['loss'])

# plt.plot(model.history.history['val_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

def eval1( dataset, print_outputs = False):

    # get the source words and target_word_labels for the eval dataset

    source_words = dataset

```

```

vocab = model.target_dict.vocab

loaded_model = tf.keras.models.load_model('eng_wol_model.h5', compile=False)

# using the same encoding network used during training time, encode the training
encoder_outputs, state_h, state_c =
loaded_model.predict(source_words, batch_size=model.batch_size)

# for max_target_step steps, feed the step target words into the decoder.

predictions = []

step_target_words = np.ones([source_words.shape[0], 1]) * model.SOS

loaded_modeld = tf.keras.models.load_model('eng_wol_modeld.h5', compile=False)

for _ in range(model.max_target_step):

    step_decoder_outputs, state_h, state_c =
loaded_modeld.predict([step_target_words, state_h, state_c, encoder_outputs], batch_size=model.batch_size)

    step_target_words = np.argmax(step_decoder_outputs, axis=2)

    predictions.append(step_target_words)

# predictions is a [time_step x batch_size x 1] array. We use get_target_sentence() to recover
the batch_size sentences

predicted = model.get_target_sentences(np.concatenate(predictions, axis=1), vocab)

#Modification

if print_outputs:

    sources =
model.get_target_sentences(np.array(source_words[0:len(source_words)]), model.source_dict.vocab)

    return sources, predicted

```

```

def proprocessing(source):

    # changing to lowercase,remove punctuation, strip trailing/leading whitespaces and tokenize
    each sentence.

    a=[]

    source_sen = [re.sub('[\W]', "", str(token.lower() )) for token in source.split(' ')]

    a.append(source_sen)

    # get the sents-as-ids for each sentence

    source_words = [[source_dict.word2ids.get(token,source_dict.UNK) for token in sen] for sen in
a]

    print('source_words',source_words)

    source_test = pad_sequences(source_words,padding='post')

    return source_test

def translate2( ):

    # Prints out a set number of translations from the test set

    data=input('Please enter the text input: ')

    data=proprocessing(data)

    sources, candidates = eval1(data,print_outputs=True)

    example_no = 2

    for i in range(example_no-1):

        print(f"Output:{i+1}")

        print(f"Source sentence: {' '.join(sources[i]).replace('<pad>', '').replace('<unk>', '')}")

        print(f"Predicted translation: {' '.join(candidates[i]).replace('<pad>', '').replace('<unk>', '')}")

translate2()

```

```

# load HDF5 format

#loaded_model = tf.keras.models.load_model('eng_wol_model.h5')

def translate(model, sentence = 15):

    # data=input()

    sources, Predicted, Actual = model.eval(test_data,print_outputs=True)

    for i in range(sentence-1):

        print(f"example:{i+1}")

        print(f"Source sentence: {' '.join(sources[i]).replace('<pad>', '').replace('<unk>', '')}")

        print(f"Predicted translation: {' '.join(Predicted[i]).replace('<pad>', '').replace('<unk>', '')}")

        print(f"Actual translation: {' '.join([l[0] for l in Actual][i]).replace('<pad>', '').replace('<unk>',
)})")

# load HDF5 format

# loaded_model = tf.keras.models.load_model('eng_wol_model.h5')

translate(model)

#Clear session prior to creating the architecture

tf.keras.backend.clear_session()

model_attention = seq2seqModel(source_dict, target_dict, True)

model_attention.build()

### Training and test evaluation

model_attention.train(train_data,eva_data,test_data,50)

model_attention.history.history['loss']

```

```

model_attention.history.history['accuracy']

model.history.history.keys()

# summarize history for accuracy

plt.plot(model.history.history['accuracy'])

# plt.plot(model.history.history['val_accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# summarize history for loss

plt.plot(model.history.history['loss'])

# plt.plot(model.history.history['val_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# plot_model(model_attention, to_file='model_plot.png', show_shapes=True,
show_layer_names=True)

translate(model_attention)

### Translate from user input

def eval1( dataset, print_outputs = False):
97

```

```

# get the source words and target_word_labels for the eval dataset

source_words = dataset

vocab = model.target_dict.vocab

loaded_model = tf.keras.models.load_model('aeng_wol_model.h5', compile=False)

# using the same encoding network used during training time, encode the training

encoder_outputs, state_h, state_c =
loaded_model.predict(source_words, batch_size=model.batch_size)

# for max_target_step steps, feed the step target words into the decoder.

predictions = []

step_target_words = np.ones([source_words.shape[0], 1]) * model.SOS

loaded_modeld = tf.keras.models.load_model('aeng_wol_modeld.h5', compile=False,
custom_objects={"AttentionLayer": AttentionLayer })

for _ in range(model.max_target_step):

    step_decoder_outputs, state_h, state_c =
loaded_modeld.predict([step_target_words, state_h, state_c, encoder_outputs], batch_size=model.b
atch_size)

    step_target_words = np.argmax(step_decoder_outputs, axis=2)

    predictions.append(step_target_words)

# predictions is a [time_step x batch_size x 1] array. We use get_target_sentence() to recover
the batch_size sentences

predicted = model.get_target_sentences(np.concatenate(predictions, axis=1), vocab)

#Modification

if print_outputs:

```

```

        sources = model.get_target_sentences(np.array(source_words[0:len(source_words)]),model.source_dict.vocab)

    return sources, predicted

def proprocessing(source):

    # changing to lowercase,remove punctuation, strip trailing/leading whitespaces and tokenize
    each sentence.

    a=[]

    source_sen = [re.sub('[\W]', '', str(token.lower() )) for token in source.split(' ')]

    a.append(source_sen)

    # get the sents-as-ids for each sentence

    source_words = [[source_dict.word2ids.get(token,source_dict.UNK) for token in sen] for sen in
a]

    print('source_words',source_words)

    source_test = pad_sequences(source_words,padding='post')

    return source_test

def translate2( ):

    # Prints out a set number of translations from the test set

    data=input('Please enter the text input: ')

    data=proprocessing(data)

    sources, candidates = eval1(data,print_outputs=True)

    example_no = 2

    for i in range(example_no-1):

```

```
print(f"Output:{i+1}")  
  
print(f"Source sentence: {' '.join(sources[i]).replace('<pad>', '').replace('<unk>', '')}")  
  
print(f"Predicted translation: {' '.join(candidates[i]).replace('<pad>', '').replace('<unk>', '')}")  
  
translate2()
```