Hybrid Artificial Neural Machine Translation using Deep Learning Techniques English-to-Afaan Oromoo

By:

Dereje Saifu Kebede

A Thesis Submitted to the

Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment for the Degree of Masters of Science in

Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

January 2019

Adama, Ethiopia

Hybrid Artificial Neural Machine Translation using Deep Learning Techniques English-to-Afaan Oromoo

By:

Dereje Saifu Kebede

Name of Advisor: Mohammad Wazi (Ph.D.)

A Thesis Submitted to the

Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment for the Degree of Masters of Science in

Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

January 2019

Adama, Ethiopia

Declaration

I declare that the study has been on papered by myself and that no other submitted professional qualification for the research. I announce that the research paper succumbed is mine, with the exception of where the study involved in together-authored publications.

Name: Dereje Saifu Kebede

Signature:_____

The thesis presented for assessment with my authorization as a thesis advisor.

Name :- MOHD. WAZIH AHMAD Assistant Prof. (Ph.D.)

Signature :-_____

Submission Date :- _____

Approval of Board of Examiners

We, the members of the board of examiners of the final open defense by *Dereje Saifu*, have read out and assessed his/her thesis entitled "*Hybrid Artificial Neural Machine Translation using Deep Learning Techniques English-to-Afaan Oromoo*" and examined the candidate. This is, therefore, to certify that the thesis has been accepted in partial the fulfillment of the requirement of the Degree of Computer Science and Engineering.

_____	_____	_____
(Supervisor/Advisor)	(Signature)	(Date)
_____	_____	_____
(Chairperson)	(Signature)	(Date)
_____	_____	_____
(Internal Examiner)	(Signature)	(Date)
_____	_____	_____
(External Examiner)	(Signature)	(Date)

Acknowledgment

First and headmost, I would like to give thanks for **Eyesus Kirestose** and his mother, **Weladite Amlak Dingel Mariyam** Almighty, for giving me the ability, strength, knowledge, and opportunity to keep it up and accomplish this research study adequately. Without **Eyesus Kirestose** and **Weladite Amlak Dingel Mariyam** blessings, this achievement not thinkable.

I want to acknowledge the one my honest thankfulness to my consultant(advisor) **Dr. MOHD. WAZIH** for the continuous advice my research investigation for his immense knowledge, motivation, patience, and enthusiasm. I could not have an expression of having the best advisor and mentor for my thesis.

Next to my consultant, I acknowledge ASTU for the second-degree chance and the remaining thesis committee: **Dr. Mesfin Abebe**, 3S SIG members, and coordinator for their insightful comments, encouragement, and hard questions. In particular, I am grateful to **Dr. Mesfin Abebe** for enlightening me on the first glance of research.

I would also like to express special delivery my thankfulness to my sweetie wife without thanking the leading source of my strength, my wife. The blessings of my wife **W/ro. Erkeselam Gezahegn** and my children's **Dagim Dereje** and **Zemariyam Dereje** when I confused and for encouraging me to keep on my vision. Also, my niece friend **Ato Eyobe Yilma and Anteneh Tilaye** supported me never let things get boring, have all made an incredible contribution to my thesis.

I want to express the gratitude to my family: my parents **W/ro Masresha Belay** and **Ato Saifu Kebede**, for giving life to me in the first place and supporting me morally throughout my life and also my brothers **Asnake Saifu, Endale Saifu** and **Trariku Saifu** for potentially and economic support.

Last but not the least, I thank my class meets: **Asnake Ayele, Daniel Aschalew, Tadele Shimelis, Anteneh Tilaye, Eliyas Mohammed, Dureti Genemo, Habtamu Alemayehu, Amanueal Negash, Musa Ahmed, Moti Bacha and Yonas Kenenisa** for the sleepless nights, we were working jointly for discussions, and for everything the enjoyment we have had in the last two years.

Dedication

The dedication of my research study is to my family. An extraordinary sense of thankfulness to my sweetie wife, **W/ro Erkeselam Gezahegn**, for my children's **Dagim Dereje and Zemariyam Dereje**, and for my parents **W/ro Masresha Belay** and **Ato Saifu Kebede** whose support of inspiration and guide me to work hard. My wife **Erkeselam Gezahegn** has never left my side, and she is very special to me, and whose dreams for me have resulted in this achievement and without her loving upbringing and nurturing. I beside commit this success to the rest of all my families, who have assisted me overall in the steps. I always going to be admitted the significance of all they performed, especially my mother, Masresha Belay, for helping me by taking care of my children for a long time. Both of you have been my best cheerleaders.

Table of Contents

List of Tables

List of Figures

List of Equations

List of Sample code

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
ANN	Artificial Neural Networks
BiRNN	Bidirectional Recurrent Neural Network
BLEU	Bilingual Evaluation Understudy
CGI	Common Gateway Interface
CNN	Convolutional Neural Networks
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
EBMT	Example-Based Mation Translation
EOS	End of Sentence
FDRE	Federal Democrat Republic of Ethiopia
GPU	Graphical Processing Unit
GRU	Gated Reccurnt Unit
HNMT	Hybrid Neural Machine Translation
HTML	Heypertext Markup Language
KohNN	Kohonen Neural Network
LSTM	Long Short Term Memory
MNN	Modular Neural Systems
MT	Machine Translation
NIST	NationaliInstituteiof StandardsiandiTechnology
NLP	Natural Language Processing
NMT	Neural Machine Translation
NN	Neural Networks

OOV	Out-Of-Vocabulary
OSNMT-tr	Open-Source Neural Neural Machine Translation-tensor-flow
PBMT	Phrase Based Machine Translation
ReNN	Recursive Neural Network
RFB	Radial-Basis Fun
RNN	Reccurent Neural Network-Search
RU	Recurrent Unit
SGD	Stochastic, Gradient Descent
SMT	Statistical Machine Translation
SOS	Source-of-sentence
TER	Translation Edit Rate
UNK	Unknown
UTF-8	Eight-bit Unicode Transformation Format
WMT	Conference for Machine Translation
XML	Extensible Markup Language

Abstract

Artificial neural networks are at the basis of the most state-of-the-art for a variety of activities and have enjoyed great success in a variety of tasks such as machine translation, image recognition, speech recognition, text summarization, and in another computation. Neural networks with recurrent model so-called NMT, have recently been applied to machine translation and begun to show successive results. However, as a recently appeared method, the approach has some restrictions. A neural machine translation system typically has to apply a vocabulary of some size to prevent the time-usage in training and decoding; consequently, it produces a critical out-of-vocabulary difficulty. Moreover, the decoder lacks a mechanism to assurance all the source words to be translated and usually resulted short translations, resulting in influent but insufficient translations. The problems, are addressed by modeling statistical machine translation, components with the basis of NMT model. The main aim of this research is to design a model that significantly reduces translation problems shown in the neural machine translation system. The proposed model is implemented by classifying the model into three layers namely; encoder layer, hybrid layer and decoder layer. Our experiment result shows that the proposed method significantly improved the translation quality of the state-of-the-art neural machine translation system on English-to-Afaan Oromoo translation tasks prepared dataset. The problems; out-of-vocabulary lead to unknown word output, lack of translation problem for long sentences, lack of decoding mechanism, which badly hurt the translation quality in the neural machine translation system is reduced by our proposed hybrid neural machine translation model. Our approach resulted in a gain of up to 2.4 BLEU score on test sets. The BLEU score evaluation of the translations obtained with this combination increased, and this also results in improved translation quality as observed in our experiments. Experiments on English-to-Afaan Oromoo translation tasks show that our system achieves significant improvements over the reference on a small amount of the training dataset collected from the web. The proposed method achieved better performance compared with the previous models (Recurrent Neural Network-search, and Statistical Machine Translation).

Keywords: Artificial neural network, Afaan Oromo, Hybrid, Natural Language Processing, Machine Translation, Neural Machine Translation, Machine Learning, and Deep learning

CHAPTER ONE

1. Introduction

This chapter described the introductory part of machine translation by stating the background of neural machine translation. What are the researchable problems, the research hypothesis used to answer the relevance of the problem? It is addressing the research problem, the purpose of general and specific objectives to solve the research problems stated in the statement of the problem, and the general research methodology carried out. The methods that are addressed used to achieve the research objectives.

1.1 Background of the Study

The human brain is the most significant fantastic organ in the human body. The mind commands the means we observe every sense organs. For a century, we've dreamed of developing smart machines. The machines with brains like our robotic helpers to clean our homes, self-deriving cars, tools that automatically detect diseases, a device that translates the human language [1][2]. But building these artificially intelligent machines helps us to solve some of the most complex computational problems. Toward to address these problems, researchers will have to acquire a radically another way of programming. The intelligent machine is an active area of artificial computer intelligence, named to deep learning [3].

Automated translation starts when the race for information and technology motivates researchers and scholars to find a way to translate information quickly. Difficulties, in an aware foreign language, have reduced significantly since machine translation (MT) has come to the field. The concept of MT refers to the artificial intelligence (AI) method of communicating with an intelligent system using automated translation(translation computed by a computer), without the interference of human's natural language [4]. Also, MT refers to build an intelligent machine that interacts and translates the human tongue to communicate just like human beings easily.

MT is a process, a subpart of natural language processing, referred to as NLP. MT is a way or a means of human language translation without the involvement of human beings from trained datasets(corpus). MT, which uses a bilingual, monolingual, and parallel dataset and other language

resources to develop language and phrase models make use of to translate text. Currently, different MT approaches are used for language translation purposes. The MT carried out using a single translation engine (i.e., SMT, NMT, Rule-based) or using multiple translation engines (i.e., Hybrid approach) [5][6][7][8][9][10].

Artificial neural networks (ANN) have enjoyed great success in a variety of tasks such as MT, image recognition, speech recognition, text summarization, and in another computation [11]. Neural networks (NN) have recently been applied to MT and begun to show successive results. Different researchers [5], [8], and [11] built NN to perform end-to-end translation, called NMT. NMT is the new methods developed on a deep NN system that contains two components, an encoder and a decoder based on the vector. NMT models accomplish the translation process by using an encoder-decoder architecture [11]. Those architectures widely used in sequence transformation tasks, e.g., image caption, text summarization, natural language conversation, time series, and MT. Specifically, the encoder-decoder architecture made up of two NN. An NN-based encoder translates (encodes) the source of the sentence into a vector representation based on which another NN-based decoder decodes the vector representation to generate a decoded target sentence word-by-word [5][11][8].

NMT has several advantages over the conventional SMT models, which exploit fancy features under the log-linear framework [5]. First, the NMT model, which is essentially a conceptually single vast NN. It entirely constructed by learning the translation knowledge from the training corpus with less effort of a feature designed by linguists or engineers. Second, gating and attention techniques, widely adopted in the field of NLP. The two methods proved to be effective in capturing potential long-distance dependencies and complicated word alignment information in the MT process, both of which pose a series of challenges for SMT. Lastly, the memory consumption of the NMT system model is much smaller than that of conventional SMT models, which maintain a large translation model, a reordering model, and a language model. From another point of view , NMT has some limitations compared with other SMT and other translation models. Primarily, the model usually has to apply a vocabulary of a specific amount to prevent time-consuming training and decoding [7]. However, the NMT system has some limitations in the structure of the model in the translation process. Firstly, it causes a series out-of-vocabulary (OOV) [12][10][13] problem; secondly, the decoder lacks a mechanism to produce all the given words

and characters to be translated [14][15]; and lastly, it offers short translations resulting in insufficient conversions [16]. With many approaches in hand to achieve a computer-translated sentence or paragraph, SMT and NMT approaches are playing the lead role for a translation system.

The current research result [5] shows that a promising result with the combination of NMT and SMT can produce promising results using machine learning compared with a single model [10][12]-[16]. Developing hybrid models that are capable of reading and generating words using multiple engines is good-looking for many possibilities. Primarily it opens the possible forms of models reason about obscure source words, such as morphological variants of observed words with the best integration of the two engines. It can handle the OOV problem. Next, the system permits the creation of hidden target words successfully recasting the translation as open vocabulary tasks that avoid a lack of decoding problems. Lastly, the system benefit from a significant reduction of the given source and out target vocabulary size as to be in the system modeled clearly to handle short translation[17][18].

The proposed model uses hybrid neural machine translation (HNMT) technique by experimenting on English-Afaan Oromoo prepared parallel dataset. The model implemented to translate English written text into Afaan Oromo text using a deep learning approach. HNMT system takes the three components from SMT, and it depends more on the NMT model. To alleviate the problems shown in NMT described above, the proposed paper incorporates SMT components, such as a translation model, word reward feature, and an n-gram language model, with the phrase-based NMT model. The RNN encoder-decoder is trained on small-sized parallel corpora dataset and performs an end-to-end translation.

On the other hand, under the existing architecture, it is not an easy task to get better translation quality by incorporating other translation components. Specifically, as for many different NLP tasks, the model is capable of incorporated using long short term memory so-called (LSTM) and attention-based bidirectional recurrent neural network (BiRNN). HNMT model is an architecture of the MT model, which implemented on the bases of the NMT system model. The proposed model automatically translates text from the English language to the Afaan Oromo language, with the integration of few components from SMT on the bases of the NMT model. The critical architecture

they have is that they work on sequences: given an input sequence, they produce an output sequence.

1.2 Motivation of the study

The existing Internet resources that show from the world population, only twenty percent of the world population speaks English. Also, from the current Internet resource, fifty percent is available in the English language. The text data are written in the English language into the actuality world, to translate vast amounts of accurate MT tools required. Contrarily, according to different research findings from the world languages, African languages, including Ethiopian languages, which contribute around thirty percent. But, the words profoundly hurt from the lack of adequate language resources [19]. For example, Afaan-Oromoo is the regional language of the in Oromia and spoken in different parts of Ethiopia and Africa. Therefore, a set of translation demands among the English document to the Afaan-Oromoo language is needed.

Reasoning from this fact for users to offer MT tools of the country, the materials written in English languages, the resources need translating to the tongue they understand. To share the resources, and to make the society more reliable for information. Since manual translation is expensive, a hopeful alternative is the use of MT. Particularly HNMT, as Ethiopian languages suffer from a lack of primary linguistic resources such as Afaan-Oromoo. The significant and essential corpus required for MT is parallel corpora, which are not available popularly for Afaan-Oromoo languages. The preparation of the HNMT model for Afaan-Oromoo is, therefore, an essential part of facilitating future MT research and development. Still, there is a need and translation demand for Afaan-Oromoo language document translation. For the translation demand, there are no highly accurate systems in translation quality and speed software and systems because of a lack of well-designed translation model.

1.3 Statements of the Problem

In the area of MT, NMT has recently accomplished much improvement in academic and industrial fields. It advances the potential of spoken translation and written translation. NMT has numerous benefit over the SMT models. Primarily, NMT model, which is totally built by training the translation information (knowledge) from the training dataset with minimum effort of components

designed for the model [5][8][11][20][21]. Additionally, attention [9] and gating [22] methods, which are commonly accepted in the area of NLP, have been verified to be applicable in catching possibility of long distance needs and complex word-alignment knowledge in the MT process, both of them cause critical difficulties for SMT. Finally, the memory usage of NMT model is significantly less than SMT models, which maintain reordering model, language model and translation model [23].

Different researches proposed a solution to reduce the problems shown in NMT model; most of them are proposed a solution by combining SMT with the NMT approach. The hybridization of recent studies has some limitations (gap) incase of combining SMT and NMT models. First, NMT system has limitations due to smaller vocabulary size compared to SMT system since many words in the source language are marked as unknown words; as a result, many words are translated as OOV, words problem leads to strange word output and unable to perform well in case of terminology translation, fixing incorrectly translated non-translatable token [12][22]. In other hands, to moderate model complexity, an NMT system usually uses the uppermost common words in the training dataset and which considers other words that are unobserved leads to out of terminology difficulties. When OOV words occur, the model cannot handle a larger vocabulary, lead to the unknown word problem in the sentences to be translated and the translation quality would be poorly affected [23]. Moreover, the NMT decoder deficiencies of a system to undertaking that all the given words are translated and frequently prefer short translations. This short translation occasionally results in an insufficient translation that does not convey the full meaning of the source sentence [16][14][7].

The proposed model uses an HNMT technique by experimenting on English-Afaan Oromoo prepared parallel dataset. The model implemented to translate English written text into Afaan Oromo text using a deep learning approach. The three SMT models usually do not make a mistake in translating low-frequency words. Because they perform phrase mapping which produces a target word that has been observed as a translation in the training data at least once. HNMT system takes the three components from SMT, and it depends more on the NMT model. To alleviate the difficulties shown in NMT described above, the proposed paper incorporates SMT components, such as an n-gram language model, word reward feature, a translation model, on the basis of the

NMT model. The proposed model uses the HNMT model that learns to encode-decode the given text, and the model is capable of incorporating using LSTM and attention-based BiRNN.

1.4 Research Questions

Based on the above problems, the following are some questions in the proposed research that will be addressed:

- How to handle the OOV translation problems lead to unknown word output using a hybrid translation approach?
- How can models be incorporated with the parallel dataset to enhance the lack of translation for long sentences to improve translation quality for fluent translation?
- How can be alleviated lack of decoding translation problems occurred in NMT system using a hybrid approach?

1.5 Objectives of the Study

1.5.1 General (Main) Objective

The main aim of this study is to design a hybrid NMT model that reduces OOV and translation quality problems shown in the NMT system by conducting an experiment on English to Afaan Oromoo.

1.5.2 Specific Objectives

The specific objectives of the HNMT model are as follows:

- Reviewing and analyzing the methodologies used in hybrid machine translation.
- Designing a general architecture model for the proposed HNMT.
- Developing a better language translator algorithm using the hybrid approach.
- Preparing a parallel corpora dataset for both English-Afaan Oromo language pairs.
- Designing and modeling a prototype for the Afaan-Oromoo text machine translation system with the hybrid model.
- Testing and evaluating hybrid machine translation with previous machine translation.

- Discussing the evaluation of multiple adaptation ways and the proposed method by comparing with the RNN and SMT existing model on the basis of bilingual evaluation understudy (BLEU-4)-metrics

1.6 Research Methodology

We conducted experimental type of research method [24] to achieve the research aim(objectives). Different stages of this research named as discussion with NLP experts, data collection, and preparation, literature review, analyzing written documents, selecting the approach, preparing parallel corpus for dataset, training, validating, and testing the proposed model, develop a prototype, and evaluation of the proposed model. The following steps are used to address the research objective.

i. **Data Collection and Preparation**: to show a morphological behavior of a language, a well collected, sized, and defined text data is required. The collection of text data is, therefore, an essential component in developing a hybrid neural MT. Thus, the corpus used in this research collected from online religious documents, Federal Democrat Republic of Ethiopia (FDRE) constitution, FDRE criminal code, Council of Oromia regional state, and English-Afaan Oromo language education materials. Afaan Oromoo-English dataset of limited vocabulary developed by applying the different rules, and the researcher prepared small-sized datasets more than 13000 corpus sentences. The need for parallel dataset preparation is because there is no well organized and prepared standard dataset for the proposed model and English-Afaan Oromoo language pairs. Additionally, the need for dataset preparation is that, some of the prepared parallel dataset have lack of corpus clarity. The parallel dataset preparation is needed to experiment the model on English-Afaan Oromoo language pairs in model training and testing phases.
ii. **Model selection:** This section shows model training, validating, and testing data preparation(pre-processing) steps. One difficulty in NMT is, it requires a very large amount of corpus preparation. It is challenging to train, validate, and test the model using HNMT for those under corpus resourced language like Afaan Oromo. To overcome massive amounts of corpus preparation problems, the researcher used zero resources[25] machine learning strategies to train and test the model [26]. We followed hold-out cross-validation machine

learning approach. From the prepared dataset, we used 70% for model training and 30% for model testing. Once the English-Afaan Oromo datasets ready, then it passes through the training phase, tuning phase, and finally testing step.

iii. **Tools and techniques:** Beforehand choosing a toolkit, a researcher needs to gain a general idea of the various open-source NMT toolkits that are available at the time of writing. The system model uses open-source NMT tensor flow (OSNMT-tr) [27] toolkit; tensor flow is a large-scale, general-purpose, open-source machine learning toolkit, for the implementation of these deep artificial NMT [28]. OSNM-tr is not a language-dependent, and preferable tool kit for morphologically abundant languages like Afaan Oromo and under-resourced dataset assets in the state-of-art NMT approaches [29].

iv. **System Prototyping Tool:** To proposed system model prototype is developed using a Python programming language used. The system used a python anaconda environment for model experimentation and implementation, and the proposed model used python libraries and packages [27]. Python programming language included libraries and packages for scientific computing and technical computing. Also, through python, the system can easily import OSNMT-tr and open-source MT apllication programming interface(API) function and to access translation components layers [29].

v. **Evaluation Metrics:** The researcher is used BLEU score evaluation metrics to evaluate the proposed HNMT model by experimenting English-Afaan dataset source and target language with reference language. The cornerstone of the BLEU score metric is a precision-oriented measure that counts the n-gram matches between the MT output. The gold standard references, quick, language-independent, numerical translation affinity(closeness) metric, a dataset of well quality human cross-reference translations, and correlates highly with human evaluation, BLEU score evaluation metric is the leading metric currently available in the MT state-of-art [30][31].

1.7 Scope and Limitation

1.7.1 Scope

This study focuses on developing HNMT using deep learning approach by implementing LSTM and attention enabled encoder-decoder architecture. We concentrate on the OOV problem leads to unknown word output, lack of translation problem for long sentences, and lack of decoding

mechanism problem that badly degrade translation quality indicated in section 1.3. The proposed model only experimented on the English-Afaan Oromoo dataset and translate sentences written in English to Afaan Oromo with a minimal amount (more than 13 thousand sentences) of text dataset and dataset trained by the proposed model. The proposed model only centers on SMT's three best components to handle the problem as shown in section 1.3 and the proposed model implemented on the bases of the NMT model.

1.7.2 Limitations

Basically, due to the scope of the research as proposed in the solution, to properly avoid and handle the problems shown in section 1.3. The limitations of this paper include; the experimentation is not implemented in more than two language dataset. Translation in different translation levels, speech recognition (i.e. speech to speech, speech to text vis verse) not investigated in this research, and in the proposed model. On the other hand, because of an availability of research done on English to Afaan Oromoo language using NMT approaches its difficult to compare the findings of the study. An other difficulties are lack of well organized and prepared standard open access English-Afaan Oromoo parallel dataset(corpus), dataset size that fit with the proposed model, NLP tool that directly helps MT for Afaan Oromoo language and computational hardware problem.

1.8 Significance

According to research study [32], 72.1 percent of the clients waste most, or all of their time-on web sites in their language, 72.4 percent answered they would likely to market an item for consumption with communication in their tongue, and 56.2 percent says that the need to get hold of communication in their language(understanding) is more significant than cost. These are just only some of the massive reasons that translation has become necessary in the modern-day and ever more globalized world that we live in. In the technology age, the rate of MT is exponentially quicker than that of human translation[33]. The average human converter can translate approximately 2,000 words in a day. The performance compared with MT, the output of MT is not in its useable closing form right away, but in specific scenarios, it can be quite useful.

1.9 Application of Result

The application result of the HNMT model is applicable in different fields of language translations in the world, and in the national aspect, used by incorporating this research result with their system, in research areas, for applications and software developers, and for those who work on Afaan Oromo NLP. Moreover, the research result is used for researches based on the findings stated in the research study and in the deferent area that doesn't mentioned above such as:

- In Expert Systems
- In Speech recognition
- In NLP
- In Computer vision
- In Robotics
- It is useful for software's in grammar checking and writing platform

1.10 Thesis Structure

The thesis organized into seven chapters. *Chapter* one gives the overall introductory information of the argument. In this chapter, we try clear up what are the researchable problems, the research hypothesis used to answer relevance of the problem it is addressing the research problem, the purpose of general and specific objective to solve the research problems stated in section 1.3 and the general research methodology carried out to accomplish research objectives are.

Chapter two is a literature review. This chapter describes the literature part and source of the research area, morphology structure of the languages and to learn from others' work, and to evidence, the hypothesis of the research, different kinds of literature on the problem domain are reviewed. Additionally, this chapter identifies the gap of related paper by establishing a systematic review (systematic literature) approach.

Chapter three is about the research methodology. This chapter discusses more detail about how research methodologies are carried out and more descriptions and the logical reasoning of the morphology described in this chapter.

Chapter four is about the proposed(planned) solution of the HNMT for the English-Afaan Oromo language. Step by step description of how the proposed solution address the problems stated in the research problem using the HNMT model is developed and how it works is detailed in this chapter.

Chapter five is about the design and implementation of the HNMT for the English-Afaan Oromo language. Step by step description of how the HNMT is developed and how it works is detailed in this chapter.

Chapter six explain the discussion and result of the proposed solution compared with previous result findings drawn from this research. The effectiveness, of the proposed solution(model) of this conducted research is also described in this discussion and result chapter.

Chapter seven concludes the research and provide an insight into the recommendations and valuable suggestions from the author. The impacts of this research are also described in this concluding chapter.

CHAPTER TWO

2. Literature Review and Related Work

This chapter provides detailed background domain of machine translation using hybrid neural machine translation, type of machine translation and their properties, neural networks in the application of machine translation, subparts of machine translation; artificial intelligence, and deep learning and the requirement needed for machine translation in which the resource collected from preliminary source of machine translation review of books, journals, articles, websites and from native speaker of the language. Also, this chapter describes related works in the integration of neural machine translation and statistical machine translation using different similar work review strategies to identify their gaps.

2.1 Overview of Machine Translation

Description 1: MT is the translation of one source language to the understandable target language without human intervention of the machine[34].

Definition 2: The concept of MT is automated translation (translation computed by a computer). MT is a subfield of NLP, which uses a monolingual, bilingual or parallel dataset and additional linguistic properties named as corpus to construct language and catchphrase models used to translate text[35]. NLP is a field of computer science, AI and computational linguistics concerned with the interactions between computers and human (natural) languages, and, in particular, concerned with programming computers to fruitfully process large natural language corpora. NLP is used in many applications to provide capabilities that were previously not possible. It involves analyzing text to obtain intent and meaning, which can then be used to support an application[34].

The idea that computers can translate human languages is as old as computers themselves. The first attempts to build such technology in the 1950s in the USA were accompanied by a lot of enthusiasm and significant funding. However, the first decade of research failed to produce a usable system and the now-famous report by Automatic Language Processing Advisory Committee in 1966 found that the ten-year-long effort failed to fulfill expectations. The next time the general public heard of MT was likely in the late 1990s when the internet portal AltaVista

launched a free online translation service called “Babelfish”. Although the quality was often lacking, it became immensely popular and brought MT into the limelight again. Other internet giants presented similar services soon after, the most well-known of which is now Google translate[36].

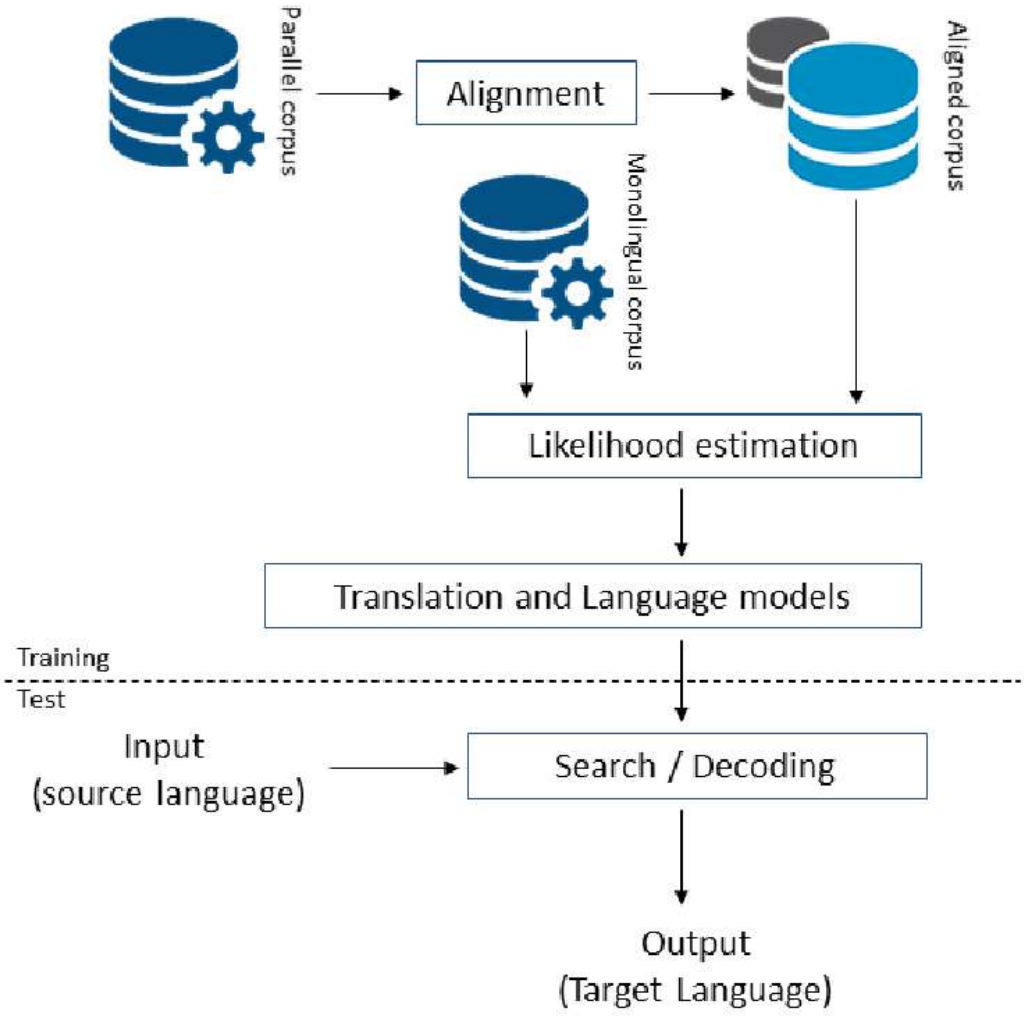

Figure 2.1: General machine translation architecture[36]

In a MT task, the input already consists of a sequence of symbols in some language, and the computer program must convert this into a sequence of symbols in another language. Given a sequence of text in a source language, there is no one single best translation of that text to another

language. This is because of the natural ambiguity and flexibility of human language. This makes the challenge of automatic MT difficult, perhaps one of the most difficult in AI. The fact is that accurate translation requires background knowledge in order to resolve ambiguity and establish the content of the sentence. Classical MT methods often involve rules for converting text in the source language to the target language. The rules are often developed by linguists and may operate at the lexical, syntactic, or semantic level [4] [36][37].

Some of the most MT Systems are[36]:

Generic MT: usually refers to platforms such as Google Translate, Bing, Yandex, and Naver. These platforms provide MT for ad hoc translations to millions of people. Companies can buy generic MT for batch pre-translation and connect to their own systems via API.

Customizable MT: refers to MT software that has a basic component and can be trained to improve terminology accuracy in a chosen domain (medical, legal, or a company's own preferred terminology). For example, WIPO's specialist MT engine translates patents more accurately than generalist MT engines, and eBay's solution can understand and render into other languages hundreds of abbreviations used in electronic commerce.

Adaptive MT: offers suggestions to translators as they type in their computer assisted tool, and learns from their input continuously in real time. Introduced by Lilt in 2016 and by SDL in 2017, adaptive MT is believed to improve translator productivity significantly and can challenge translation memory technology in the future.

A few different types of MT are available in the market today, the most widely use being example based MT, SMT, rule-based MT, NMT and hybrid Systems, which combine two different model. From the beginning, researchers concentrated almost exclusively on the translation of scientific and technical documents, where the difficulties of cultural differences and variable contexts are less acute than in the more 'culture-bound' translation of literature, legal texts, and many areas of sociology. In science and technology, the demand for translation has almost always exceeded the capacity of the translation profession, and these demands are growing rapidly. In addition, the coming of the Internet has created a demand for immediate online translations, which human translators cannot possibly meet[35][38][39].

In general, the process of translation has two levels: metaphrase and paraphrase. In metaphrase means “word-to-word” translation. It relates to “formal equivalence”, i.e., the translated version will have “literal” translation for each word in the text. However, the translated text may not necessarily convey the meaning of the original text. That means sometimes the semantics may differ from the original text. In paraphrase it relates to “dynamic equivalence”, i.e., the translated text would contain the gist of the original text but may not necessarily contain the word-to-word translation. Translation companies and departments typically evaluate MT quality by the effort it takes for a human to post-edit the output. It is often measured in pages per hour, or in the number of key strokes per segment. Specialists training MT engines rely on automated tests and metrics. They are better suited for A/B testing and experimentation and show the impact of the tiniest changes, where humans might not notice the difference. MT quality metrics include BLUE-score, WER, PER, TER, METEOR, ROUGE, HyTER, and NIST[35][38][39][40].

2.2 The need for Machine Translation

Advances in information technology have combined with modern communication requirements to foster translation automation. The need of the relationship between technology and translation goes back to the beginnings of the Cold War, as in the 1950s competition between the United States and the Soviet Union was so intensive at every level that thousands of documents were translated from Russian to English and vice versa. However, such high demand revealed the inefficiency of the translation process, above all in specialized areas of knowledge, increasing interest in the idea of a MT. Information technology has produced a screen culture that tends to replace the print culture, with printed documents being dispensed with and information being accessed and relayed directly through computers (e-mail, databases and other stored information). These computer documents are instantly available and can be opened and processed with far greater flexibility than printed matter, with the result that the status of information itself has changed, becoming either temporary or permanent according to need [32][38][37][39][41].

According to the basic need of MT English language translation is essential because from the world population only twenty percent of the world population speak English and from the available Internet resource 50% is in English language and in order to process large amount of text and speech data written in English[4]. On the other way, according to [32], 72.1 % of the consumers

spend most or all of their time on sites in their own language, 72.4 % say they would be more likely to buy a product with information in their own language and 56.2 % say that the ability to obtain information in their own language is more important than price. These are just a few of the many reasons that translation has become essential in the modern and ever more globalized world that we live in. Within the cutting edge world, there's an expanded require for dialect interpretations because language is an effective medium of communication. The demand for reading has become more in recent years due to the increment within the trade of data between different locales utilizing distinctive territorial dialects. Availability to web report in other dialects has been a concern for information professionals and other individuals or organizations who want to satisfy their information need [37][39][41].

2.3 Process of MT

As illustrated; in figure 2,2 MT passes many steps. The method of the MT system is source language input is analyzed first and finally creates an internal representation. This representation is controlled and exchanged to a frame that's appropriate for the target dialect. At that point, the final output was produced within the target language.

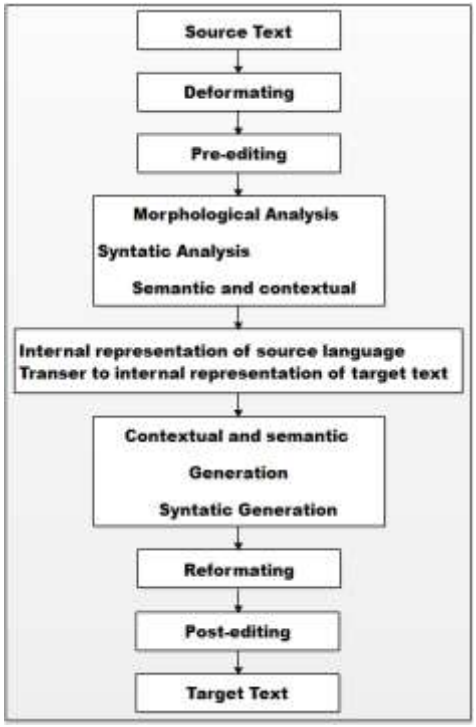

Figure 2.2: Typical machine translation process[34]

2.4 Approaches to Machine Translation and their Properties

2.4.1 Rule-based Machine Translation

A rule-based MT system is a usual of rules for interpreting the rules called grammar rules, lexicon, and computer programs. It can be extended and maintained. The rule-based approach is the first MT technique ever developed. Linguists with linguistic knowledge, write rules. In different stages of translation, rules play a major role: syntactic method [9][42][43]. However, it was limited by lexical selection in sentences of transfer and analysis failures [44].

2.4.2 Example based Machine Translation

One of the central concepts behind example-based MT (EBMT) is reprocessed the examples of accessible translations as the center for the new interpretation. EBMT has issues with language comprehension specifications, such as morphology, syntactic features, and word order [44][45]. The procedure of EBMT classified into three steps:

Matching Stage: This stage in EBMT finds examples that, based on their similarity with the input, will contribute to the translation. The way to implement the matching stage based on how the samples are stored. Samples stored in old systems as annotated tree structures and explicit ties linked to the constituents in the two languages[44].

Alignment Stage: The equivalent input parsed Alignment; whatever parts of the relevant translation reused. Using bilingual dictionaries or comparing with other examples, alignment done. It is essential to automate the alignment process in example-based MT[45].

Re-combination Stage: This stage is the final stage of the path to EBMT. Recombination ensures that the recycled parts are placed together correctly in the instance found during alignment. It takes sentences of the source language and a set of translation patterns as inputs and produces sentences of the targeted language as outputs. Recombination strategy development relies on previous matching and alignment phases [44][45].

2.4.3 Interlingua Machine Translation

Interlingua MT is a technique that uses Interlingua for any AI used to represent the meaning of natural languages, as for MT purposes. Ideally, the text representation of Interlingua should be adequate to produce sentences in any style [4]. Idioms spoken in different parts. More words have a single identical word in another language in some cases in one language. Such functional variations between styles discussed by the Interlingua method. The downside is that Interlingua's model is too complicated [36]. The model is because no simple technique built so far to create a perfect representation of Interlingua. Information storage on the nature and behavior of each word in the language and the Interlingua lexicon is required. The report includes acts and incidents. There are an analyzer and synthesizer for each style in a standard Interlingua MT system. The analyzer represents the context of the given text through Interlingua [37].

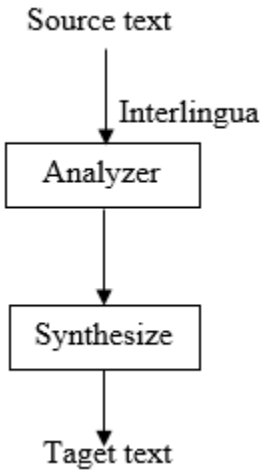

Figure 2.3 Interlingua Machine Translation[37]

2.4.4 Statistical Machine Translation

SMT uses mathematical models of interpretation created from monolingual and bilingual learning data analysis. This technique [46] primarily uses computing power to create sophisticated data models to convert one source language into another. Algorithms used, the translation is chosen from the training data to select common words or phrases [44]. Creating SMT models is a reasonably fast and easy process that involves uploading files to train the engine for a particular pair of languages. Training and SMT for a specific area requires at least two million words, but at a much lower level, it is conceivable to get a suitable quality threshold SMT technology uses

bilingual companies to train them to learn language patterns and use the dataset to improve their fluency [47]**.** When prepared using domain-specific training data; health, financial, or technological fields, SMT engines [47] can prove to be of higher output value. SMT technology is central processing unit (CPU),CPU-intensive and requires extensive hardware to excute models of translation at acceptable levels of performance. Cloud-base systems are therefore favored, allowing them to scale up to meet their users ' demands without the client having to invest significantly in the cost of hardware and software. SMT has trouble handling criteria Linguistic skills, such as grammar, syntactic functions, and word order, insufficient [46].

2.4.5 Neural Machine Translation

NMT [8] is a newly emerging NMT approach, recently applied to tasks related to AI translation. In contrast to the traditional phrase-based translation [11] that consists of many small, separately tuned sub-components, neural machine interpretation endeavors to construct and prepare a single, tremendous neural organize that peruses a sentence and produces a rectify interpretation [46]. NMT framework classified based on input, arrange, and two new sorts of attention-based models: a common approach in which all source words have gone to and a nearby way in which as it were a subset of source words consider at one time.

Most of the NMT systems proposed to belong to an encoder family with an encoder and a decoder for each language or include in each sentence a language-specific encoder. An encoder NN reads and encodes an inputted sentence in a fixed-length matrix. A decoder then returns an encoded vector translation. The entire encoder-decoder process, consisting; encoder and the decoder combine, is mutually prepared to maximize the likelihood of a rectify interpretation given a source sentence [8][11][20]. NMT use BiRNN with integration of either LSTMs estimate the condition probabilities of $p(Y_{1,}\ldots\ldots, YT'|X1,\ldots..,xT)$ where $Y_{1,}\ldots\ldots, YT'$ input sequence and $X1,\ldots..,xT$ output sequence whose length T' may differ from LSTM or gated reccurnt unit (GRU) handling of rare words [46][26].

2.4.6 Hybrid Machine Translation

The expansion of methodologies in the past decade and the introduction of new applications for automated translation processes have highlighted the limitations of adopting one single approach to the problems of translation. In the past, many MT projects were begun by researchers who saw

MT as a testbed for a particular theory or particular method, with results that were either inconclusive or of limited application. It is now widely recognized that there can be no single method for achieving good-quality automatic translation, and that future models will be 'hybrids', combining the best of methods[37][39][41].

One approach is the idea of running parallel MT systems and combining the outputs: the so-called 'multi-engine' system – the group at Carnegie-Mellon University has investigated combinations of multiple systems. More commonly, hybrids are currently envisaged as systems combining the statistical methods of SMT or EBMT, SMT or rule based and SMT or NMT with some linguistics-based methods, particularly for morphological and syntactic analysis. An example is the research at Microsoft. However, there are other ways to combine corpus-based and rule-based methods, as the research at the National Tsing-Hua University illustrates: lexical and syntactic rules can be statistically derived from corpus data and optimized by feedback from output[35][37][39][41].

Hybrid MT system is an MT approach; the reading accomplished by using multiple engines as a single translation system[8][24][48]. The motivation for developing HMT systems stems from the failure of any unique technique to achieve a satisfactory level of accuracy. HMT incorporates the assets of both the translation method and the empiricist method. The essential of HMT is to combine the core of MT engines using serial coupling or parallel coupling [44][46]. According to current research findings building a model using hybrid approach is very effective than that of single one. Some recent work has focused on hybrid approaches that combine the transfer approach with one of the corpus–based approaches. This was designed to work with fewer amounts of resources and depend on the learning and training of transfer rules. The main idea in this approach is to automatically learn syntactic transfer rules from limited amounts of word aligned data. This data contains all the needed information for parsing, transfer, and generation of the sentences [19] [20].

2.5 Artificial Neural Network

ANN [49], are computational models that are artificially inspired, allowing a computer to learn from observational data. Also, ANN's [50] are a group of algorithms designed to recognize patterns, modeled by taking the human brain. They interpret sensory data using a type of raw input clustering, labeling or machine perception. The ANN is the main tools in the machine learning

industry [40]. The patterns they identify are mathematical, stored in vectors that need to convert into all real-world data, whether images, sound, text, or time series. The neuron's basis is to respond to previously learned models. We call it an ANN when we build the same kind of replication in terms of engineering and computer science. Similar to the biological neuron, data flows in, is interpreted through an ANN and findings are distributed. As shown in (Figure 2.4), NN consist of input, output, weights and biases activation function.

Generally speaking, ANN's are a chain of nodes associated with each other via the link from which they begin to interact [40]. Neurons are carrying out operations and the result. Let's look at an example. Suppose a new film released in a movie theater. Now, in a specific movie theater, there are nearby opportunities to watch this movie. Our brain makes a split-second decision to watch the film. The split-second decision is quite clear to the mind to activate from the neurons we have, but it's difficult for the same replication forms in a machine [33]. A perceptron is very significant in the implementation of NN. If there is a NN of one layer, we can say it a perceptron. It mostly used to define the data in supervised learning. A perceptron consists of four different things: values of input or one input layer, weights, and bias, a function of summation, and activation [33]. In general, NN consists of a collection of artificial neurons, commonly known as nodes or groups, and a set of directed edges between them, intuitively representing the synapses in a biological NN [40][49] [50][51]. This study uses the ANN as the base for implementing hybrid NMT model.

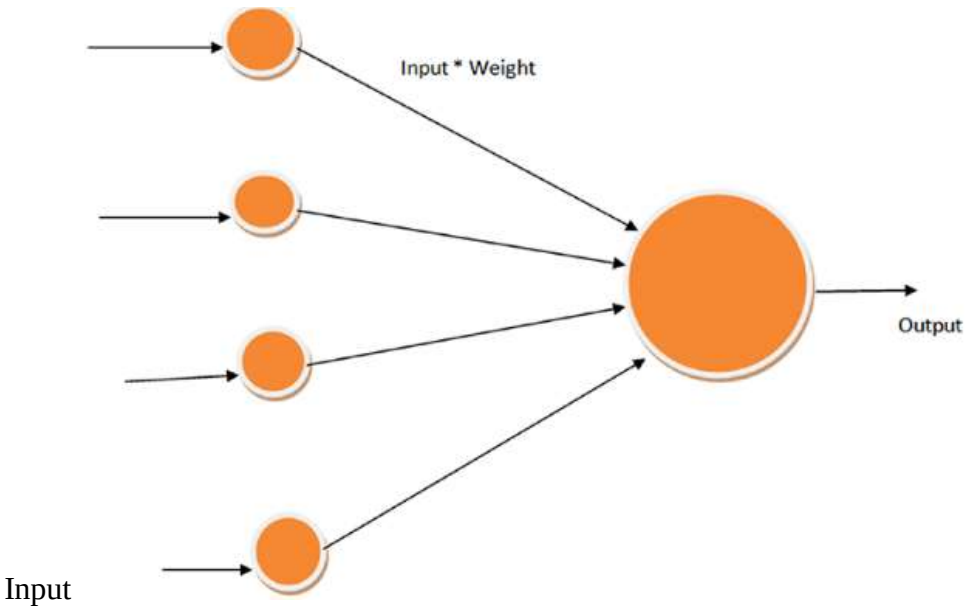

Figure 2.4 Artificial Neural Network[51]

2.5.1 Feedforward Neural Network – Artificial Neuron

This feedforward NN is one of ANN's purest form, with data or information moving in one direction [49]. The data passes and exits on the output nodes through the input nodes. The feedforward NN [40] is an actual category of new ANN known for its design simplicity. There are an input layer, hidden layers, and a returning layer in the feedforward NN [52]. Data only goes one direction–from the input layer to the output layer–and never goes backward. It may or may not have the hidden layers of this NN. In simple words, it usually uses a category activation method to have a front propagated wave and no backpropagation [53]. Below in Figure 2.5 shows the entirety of the items of inputs and weights are calculated and encouraged to the yield.

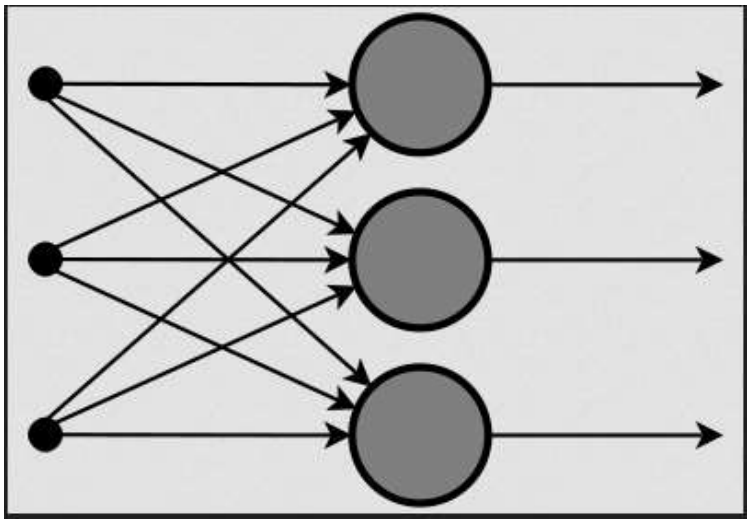

Figure 2.5 Feedforward Neural Network[53]

2.5.2 Radial-basis-Function

Radial-basis fun (RFB) NN [53] thinks through by measuring the gap of a section concerning the center. RFB NN has two layers. Firstly, to begin with where the highlights combine with the RFB NN work within the inward(inner) layer. After that, the yield of these highlights is taken into thought, whereas weighted similar output and returns within a short time-step, that is essentially a memory, illustrated in Figure 2.6[54]. RFB NN has been applied in different areas or systems. Both factors increase the risk of significant power outages [55].

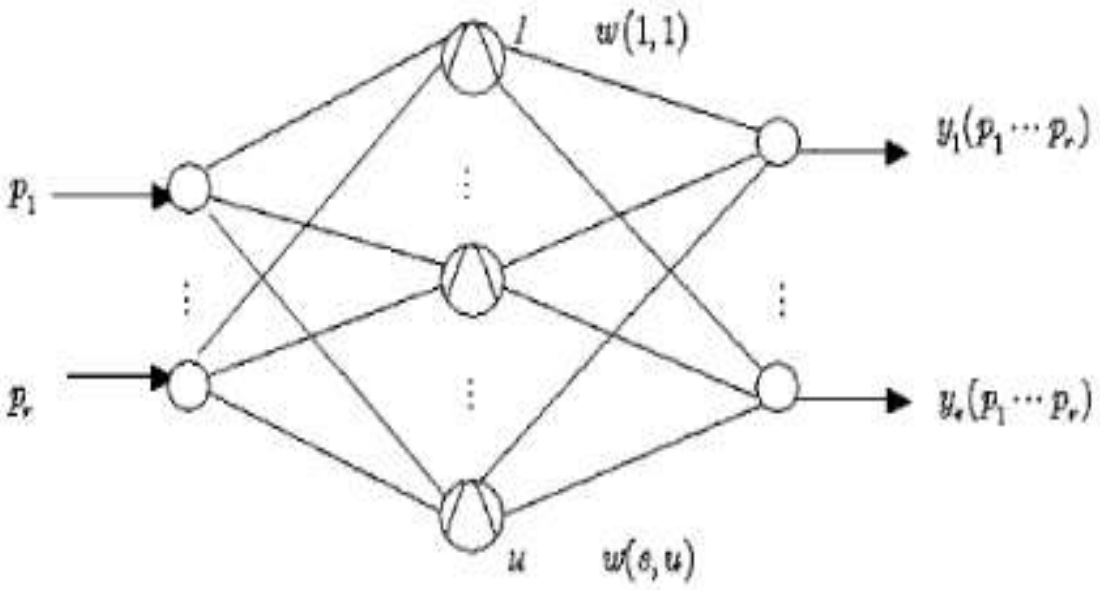

Figure 2.6 RFB Neural network [54]

2.5.3 Kohonen Self-Organizing –Neural Network

A Kohonen NN(KohNN) [53] map aims to input vectors of self-assertive measurement to discrete outline comprised of neurons. The map ought to prepare its possess organization of the preparing information[53]. When outlining, the area of the neuron maintains consistently but the point(weights) differs calculating on esteem. This self-organization prepare has distinctive parts; within the to begin with every neuron accepter prepares with minimum weight and the input vector. Kohonen's systems fundamental sorts of self-organizing neural networks. The capacity to self-organize gives good conceivable outcomes - adjustment to once-obscure input information [53]. It appears to be the foremost common way of learning, which utilize in our brains, where no design characterized. Those patterns take shape during the learning process, which combines with regular work. The precise scheme of rivalry and later modifications of synaptic wages may have various forms. KohNN [56] utilize to recognize designs within the information. Its application found in the therapeutic investigation to cluster information into diverse categories. Kohonen's outline properly categorize(classify) patient's history either glomerular or tubular, with high precision with high accuracy. There are many sub-types based on rivalry, which differ themselves by the precise self-organizing algorithm [53][56].

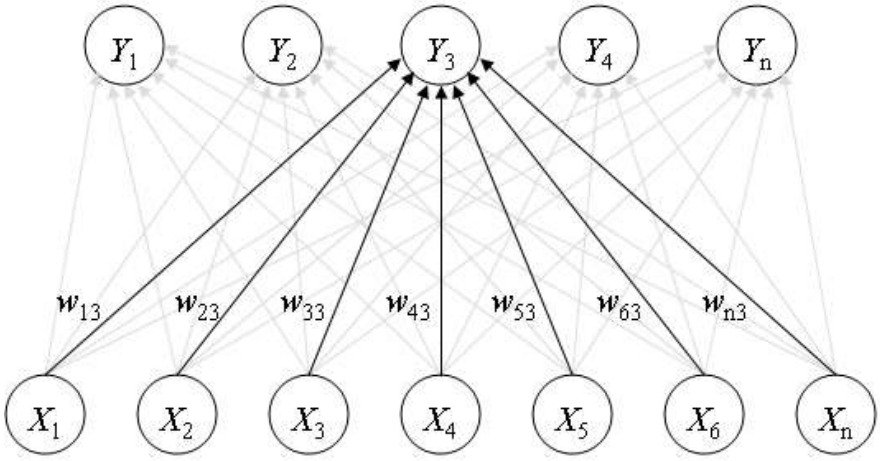

Figure 2.7 Example of a self-organizing network[56]

2.5.4 Re-current Neural Network

The architectures of NMT models [3] differ in their exact structure. A common well-chosen for sequential data RNN, applicable in NMT. Usually, an RNN uses for the two components, namely called an encoder part and decoder part. However, RNN models differ [13][28] in terms of such as

a) *directionality* – unidirectional(bidirectional)
b) *depth* – multi-layer(single) and
c) *type* – LSTM, or GRU.

The basic building block in an RNN is a Recurrent Unit (RU). There are many different variants of RU, such as the rather clunky LSTM and the somewhat simpler GRU. Experiments in the literature suggest that the LSTM and GRU have roughly similar performance [53][57]. The RNN deal on the standard of sparing the yield of a layer and bolstering this back to the input to assist in foreseeing the result of the layers [53]. The idea behind the RNN is to make the use of sequential information why we call them recurrent because it accomplishes a similar task for each of the members of a sequence [53]. The first layer form identical to the feed-forward NN product of the sum of the weights and the features. The RNN prepare begins once this is and compute, this implies that from one time step to the following, each neuron will keep in mind a few data it had within the past time-step[58]. This stage enables each neuron to act like a memory cell in performing computations. In this prepare, we ought to let the neural arrange to work on the front proliferation

and keep in mind what data it needs for afterward use. RNN are popular models [53] that scored a great success in NLP tasks. The application of RNN; in text to speech conversion models, sound and audio, video and image process. Below Figure 2.8 shows RNN's are very powerful because they integrate two components. In RNN, there are different and specific kinds of networks i.e., recursive RNN, LSTM, GRU and sequence to sequence[59]. The process sequence of RNN mapped using one to one, one to many and many to many sequence of words [53][58]. RNN is implemented in chapter 4 in detail with more mathematical models. RNN includes two different types of network LSTM and GRU [59]. This study uses RNN used to construct (build) the sequence-to-sequence model for the hybrid MT system.

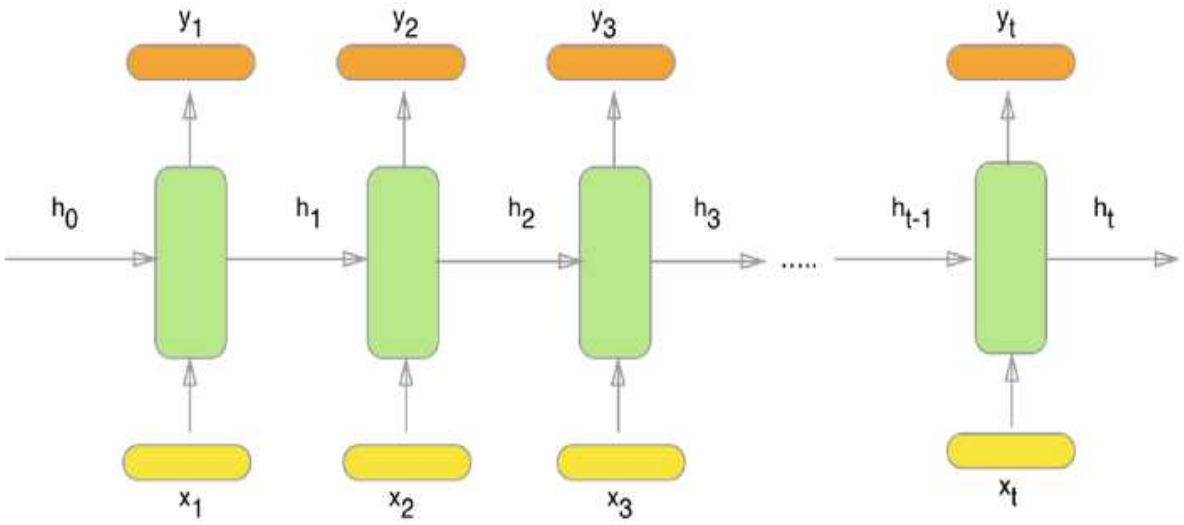

Figure 2.8 Recurrent Neural Network [59]

2.5.5 Convolutional-Neural Network

Convolutional neural networks (CNN) [3] are comparative to feed-forward neural systems. Where the neurons are capable of learning better point(weights) and predispositions. CNN is acceptable for computer vision applications for flag and picture handling which returns over OpenCV [13]. The implementation of CNN is systems like : image classification, image captioning and signal processing. Computer vision practices are take over by CNN [28] the reason that of their performance(accuracy) in [40] image classification. In CNN, convolutional layers consist of single or multiple its layer, pooling or fully connected, and uses a multilayer perceptron's [40]. In convolutional layers, input and putting the output to the subsequent layer is the primary operations. CNN shows wonderful outcomes in image and speech areas [3][60].

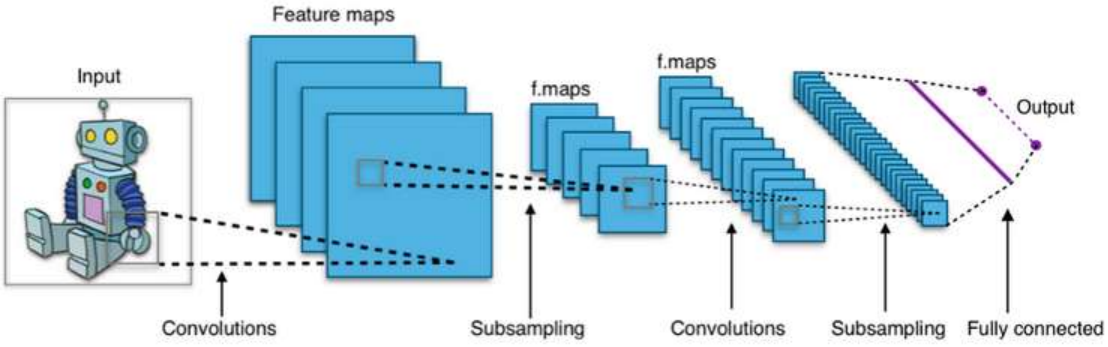

Figure 2.9 Convolutional Neural Network[60][61]

2.5.6 Recursive Neural Network

A recursive neural network (ReNN) is a form of deep NN which constructed by recurring application of the same weight set via a structure to make a structured summary; or scalar prediction of variable input structures, via topological ordering of a given structure[20]. Renn is a co-ordinating machine learning framework for processing structured area information (i.e., protein topologies, web pages for heypertext markup language(HTML), deoxyribonucleic acid regulatory networks, NLP parse trees, image analysis, etc.) [40][60]. One of its goals was to fill the variance between symbolic and sub-symbolic processing models the difference observed. In particular, computer systems that can combine numerical and symbolic data in the identical model developed. Their main advantage, in addition to feature-based methods, is their potential to work information patterns of varies sizes and, topologies. (i.e., trees or graphs) where the data related to the problem encrypted with fixed-size vector systems [62][63].

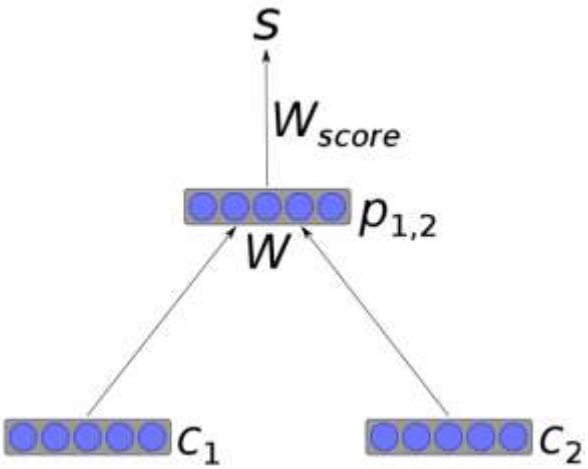

Figure 2.10 Recursive Neural Network[60]

2.5.7 Modular Neural Network

Modular neural network(MNN) have a collection of diverse systems working freely and contributing towards the yield. Each neural arrange features a set of inputs that are one of a kind compared to other systems developing and performing sub-tasks. These systems don't associate or flag every part, in finishing the errands. The advantage of a secluded neural organize is that it breakdowns a large computational handle into littler components diminishing the complexity [64][65][66].

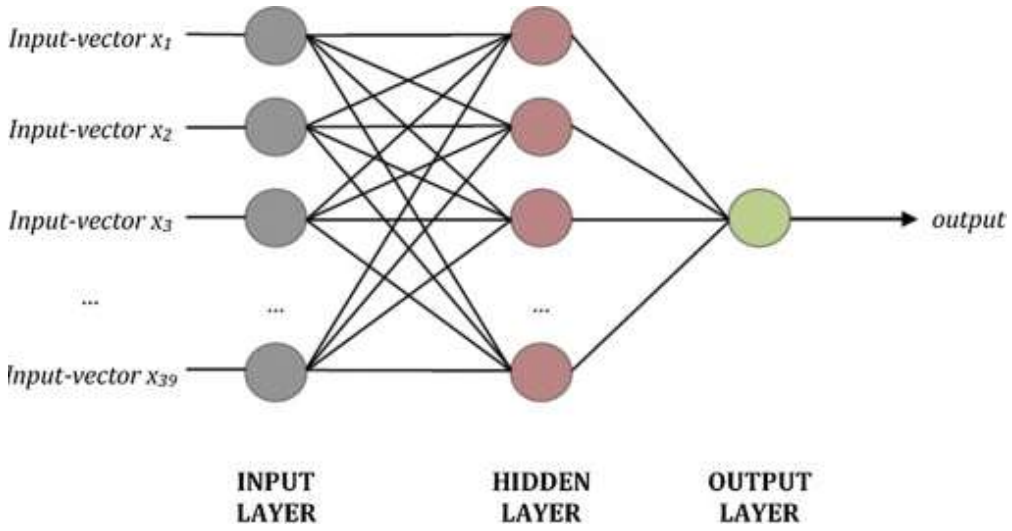

Figure 2.11 Modular Neural Network[66]

2.6 Application of Deep Learning, Machine Learning, and Artificial Intelligence in Machine Translation

The past years intense media excite in AI. In numerous articles, often outside technological publications, mechanical learning, deep learning, and AI are published [67][68].

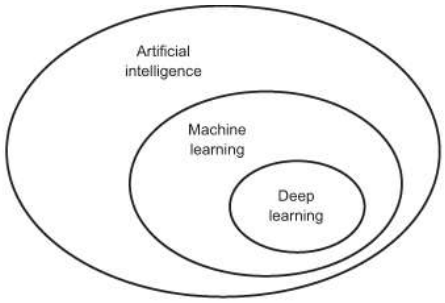

Figure 2.12 Deep learning, AI and machine learning [67]

2.6.1 Deep learning in Machine Translation

The name deep learning, referred to as stacked NN. In deep learning or knowledge networks composed of several layers, based on artificial intelligence and machine learning concept. The layer of the systems is with a massive amount of layers and hyperparameters in one(four) basic network models [2]:

- Unsupervised-pertained networks,
- CNN,
- RNN,
- RecNN

Deep learning, is a family of profound systems that can learn high-level features through several NN rather than one. The subfield of learning by machines, a modern approach to learning representations from information that emphasizes learning from successive layers of more deep observations[35][40][51]. The deep word in profound learning a deeper understanding; conventional, in-depth learning often requires ten or hundreds of successive representations automatic access to training information. In deep learning representation of layers are learned via neural network extra significant advancement has been the appearance of deep learning NN, in which unlike layers of a multilayer-network separate unlike parameters until it can understand what it is observing for [59][67][69].

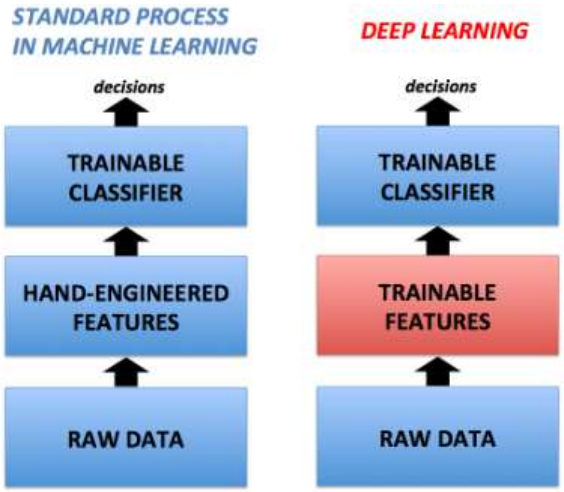

Figure 2.13 Difference between standard process in machine learning and Deep[69]

2.6.2 Artificial Intelligence in Machine Translation

MT it's AI that learns words and patterns of words so that it can respond to human searches and questions. MT is branch of AI tells the machine to understand, interpret the interaction of human and computers using natural language. The aims of AI is using data to offer solutions to existing problems[69]. The AI process loop is shown as follows in can be achieved intelligent agent the observe identify patterns using the data, plan find all possible solutions, optimize find optimal solution from the list of possible solutions, action execute the optimal solution and learn and Adapt the result giving expected result, if no adapt in NLP, to be able to communicate successfully in a chosen language [35][67][70].

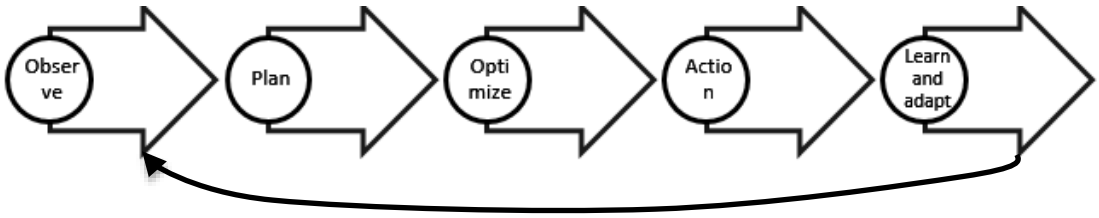

Figure 2.14 AI process loop[70]

Stages of Artificial Intelligence

Stage 1 – Machine Learning

Stage 2 – Machine Intelligence Eg – Deep Neural Networks.

Stage 3 – Machine Consciousness

2.6.3 Machine Learning in Machine Translation

Machine learning, a subpart of AI it gives the machine the capability to learn and adapt their task with the help of data that can be provided concerns a set of statistical methods to predict the target language. NLP and word analytics includes statistical approaches for labeling sentiment, entities, parts of speech and other aspects of text using machine learning [41]. Once the machine learns somewhat, the AI to progress, and operate without specifically being programmed [35][58][70]. So, we need three things to accomplish machine learning:

- **Input data points:** For MT these data points could be written text.

- **Examples of the predictable outcome:** In MT task, these could be human being generated transcripts text files.
- **Measuring the algorithm is doing a good job:** to determine the distance(length) between the algorithm's current output and its expected output.

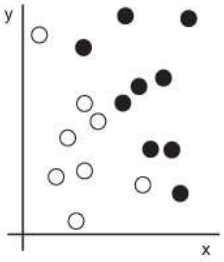

Figure 2.15 Some sample data[71]

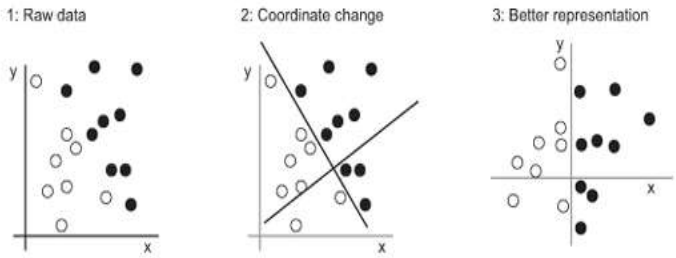

Figure 2.16 Coordinate change[71]

Machine learning approaches [71] are:

- **Supervised machine learning**:
- **Unsupervised machine learning**
- **Semi-supervised machine learning**

2.7 Afaan Oromo Language

Afaan Oromo is the language Cushitic family named as Afro-Asiatic Phylum [72]. Afaan Oromoo has widely spoken language in Ethiopia. Afaan Oromoo language Similar to other Ethiopian languages has a very rich morphology and registered with language code om and orm [73]. Alphabet, known as Qubee adopted and became an official script of the language. Presently, the language(Afaan Oromo) is an official language, of Oromia Region, and implemented as an didactic(instructional) media and educational language, of the region.

2.7.1 Afaan Oromo Scripting

The language (Afaan Oromo) uses a Latin grounded alphabet known as Qubee that includes twenty-eight necessary letters. From twenty-eight basic letters, five of them known as vowels, the other five letters are known as double consonants (Qubee dachaa), and the rest are known as consonants. Double consonant letters derived from a combination sequence of two consonant letters. Qubee typified by capital and small scripts (letters) that are familiars in the English [74]

writing system (alphabet). Vowels are sound creators similarly to the English language sounds by themselves. Vowels in Afaan-Oromoo formulated as long and short vowels. The first letters “p,“ “v“ and “z“ in Afaan-Oromoo there is no root words formulated from those letters. But, in writing the language, they are scripptrd to refer to foreign words such as “Paappaayyaa” (“Papaya”).

Table 2.1: Afaan Oromoo alphabet

Number	Capital	Small	Type	Long	Short
1	A	a	Vowel	Aa	A
2	B	b	Consonant	-	-
3	C	c	Consonant	-	-
4	D	d	Consonant	-	-
5	E	e	Vowel	Ee	E
6	F	f	Consonant	-	-
7	G	g	Consonant	-	-
8	H	h	Consonant	-	-
9	I	i	Vowel	Ii	I
10	J	j	Consonant	-	-
11	K	k	Consonant	-	-
12	L	l	Consonant	-	-
13	M	m	Consonant	-	-
14	N	n	Consonant	-	-
15	O	o	Vowel	Oo	O
16	P	p	Consonant	-	-
17	Q	q	Consonant	-	-
18	R	r	Consonant	-	-
19	S	s	Consonant	-	-
20	T	t	Consonant	-	-
21	U	u	Consonant	Uu	U
22	V	v	Consonant	-	-
23	W	w	Consonant	-	-
24	X	x	Consonant	-	-
25	Y	y	Vowel	-	-
26	Z	z	Consonant	-	-
27	CH	Ch	Double Consonant	-	-
28	DH	Dh	Double Consonant	-	-
29	NY	Ny	Double Consonant	-	-
30	PH	Ph	Double Consonant	-	-
31	SH	Sh	Double Consonant	-	-

2.7.2 Afaan Oromo sentence structure

English and Afaan Oromo take variance in their syntactic form. In Afaan Oromo, the form structure of the sentence form is subject-object-verb. Subject-object-verb is the form of sentence

formation where the subject comes in the first, the object, and the verb next to the object. For example, let us see Afaan Oromo sentence “Ababbaan dhuggatti dhuge”, “Ababbaan” is subject, “dhuggatti” is object and “dhuge” is the verb of the sentence [74].

In English, the form of sentence structure is subject-verb-object. For example, let us see the translation of the above Afaan Oromo sentence into English it will be “Abebe drunk alcohol ” in this case “Abebe” is subject, “drunk” is verb and “alcohol” is the object.

2.7.3 Language Articles

The English language structured from two types of articles known as definite article and indefinite article. In circumstance of Afaan Oromo, no articles that will be written beforehand nouns differently that of English. Instead in Afaan Oromoo the end vowel of the noun is throw down and suffixes (utti,-attii,-ittii,-icha) instead of using definite articles put in to visible definiteness.

2.7.4 Language topology

Afaan Oromo language structure uses subject object verb format [73][74];

- gender (feminine/masculine)
- post-positions
- case-marking: , number, verb(affixes mark person), gender of subject;
- passives (passive,middle active),
- aspect and tense
- twenty five consonant and ten vowel
- phonemes: pitch accent

Examples:

“I threw the sphere to you” → “I ball to you threw” → “**Ani kubbaa siif nan ha'e**”

“She came from the USA” → “She USA from came” → “**Isheen Ameerikaa irraa dhufte**”

2.7.5 Punctuation mark

Punctuation marks and apostrophe used in both languages applied for the same purpose. in English apostrophe mark (‘) indicate possession, instead of Afaan Oromo used in writing to represent

tongue known as hudhaa. It shows a significant function in Afaan Oromo reading and writing. For example, “kaa'uu” [74].

2.8 Related Work

The following point’s shows that related work review strategies for selecting the best six papers:

- Journal problem identification
- Focus area
- Comparing the problem
- How they have come to their methodologies
- Proposed models
- Performance
- Gap
- Finally converting the difference into the research question

Different researches was proposed for text translation and speech recognition for technological supported languages like English to other non-Ethiopian languages. On the other hand, attempts in this field for under resourced languages like Afaan Oromoo, in particular is not yet started so far popularly with translation quality. For Afaan Oromoo, most of the research is conducted on SMT due to the comparative availability of resources as compared to other non- Ethiopic languages.

In 2009, Sisay Adugna, with the objectives of to apply existing SMT systems on English – Afaan Oromo language pair by using available parallel corpus and to identify the challenges that need a solution regarding the language pair. Lack of utilization or accessibility of online collection for information need of Afaan Oromo speakers is considered as the main problem by the author. The researcher found digitally available Afaan Oromo versions and prepared parallel corpus for model training and testing (90%:10% ratio). The author used SRILM toolkit and GIZA++ for language modeling and word alignment respectively, the Moses decoder was used for decoding purpose.

The documents were preprocessed by using different scripts which are customized to handle some special behaviors of Afaan Oromo such as apostrophe. Sentence aligning, tokenization, lowercasing and truncating long sentences that take the alignment to be out of optimality were also done by those scripts. The author performs different experiments by varying number of N-grams, the highest BLEU score of 43.96% is observed for 1- gram scoring [76].

In 2013, Jabesa Daba proposed a method bidirectional English-Afaan Oromo MT system. The research work is implemented by combining of rule based and statistical approaches. In order to achieve the objective, a corpus is collected from different domain and prepared in a format suitable for use in the development process and classified as training set and test set. Since the system is bidirectional, two language models are developed; one for English and the other for Afaan Oromo. Translation models which assign a probability that a given source language text generates a target language text are built and a decoder which searches for the shortest path is used. Two major experiments are conducted by using two different approaches. The first experiment was implemented using statistical approach, and the second experiment was implemented using statistical and rule based approach. The evaluation result of BLUE score recorded in the second approach experimental result shows 37.1% [77].

In 2018, Million et.al. proposed bidirectional English to Afaan Oromoo SMT approach. The study investigates the effects of phrase level, word level and sentence level language translation. The experimental result shows that phrase length and alignment verities that happened because of syntactic structure and differences in word correspondence needs to be controlled to improve the performance of the translation system. The author used 19300 and 12200 sentences as a monolingual corpora for creating English and Afaan Oromo language models, respectively. the data set or corpus was collected from Ethiopian criminal code, Ethiopian constitution, Oromia Regional State Duties and Responsibilities and Holy Bible The author used GIZA++,Anymalignment and hunalign for word level, phrase level and sentence level alignment, respectively. Experimental result shows that 27% BLUE score is recorded at phrase level alignment with maximum phrase length of 16 [78].

In 2016, Marcis proposed a hybrid MT system to cover OOV word translation [12]. Keeping the NMT as a primary translator, he proposed to use SMT as post-translator for English-Latvian

language pair. The proposed system suggests that in case of OOV word, in the NMT output, it should extract alignment information of the sentence that contains the OOV word from the attention mechanism of the NMT system. Then it prepares a Moses Extensible Markup Language (XML) for translation with an SMT system. Finally, the sentence is translated with Moses. The alignment information is extracted using a heuristic algorithm that firstly identifies reciprocal maximum alignment, then it searches leftward and rightward to identify additional source tokens for which the target token is assigned with maximum alignment weight. Next, it identifies maximum alignments among unpaired token while performing the leftward and rightward search to expand alignment again. Finally, the remaining source tokens and target tokens with the highest weights are aligned. For statistical training, they used a platform based on Moses. The hybrid system showed increased BLEU-score of three (3 approximately) for translating OOV words. Still, this hybrid system is claimed to perform well in case of terminology translation, fixing incorrectly translated non-translatable token (like dates, times, number etc), aiding in formatting tag replacement in document translation.

In 2016 Arthur et al., researched an approach that enhances NMT systems, with discrete probability-based translation lexicons [10]. Translation of low-frequency words is encoded efficiently by this lexicon which is derived from either automatic learning as SMT hand-made lexicon or as a combination of both. NMT models calculate the next target word's probability for English-Japanese language pair. The NMT system was built using the Chainer Toolkit. They implemented two SMT systems for comparison: a phrase based MT(PBMT) system using Moses and a hierarchical PBMT system using Travatar. Their experiments showed improvement of 2.02 BLEU and 0.13-0.44 national institute of standards and technology score and faster convergence time. For the bias method, their system showed less effectiveness for the hand-made lexicon because it did not cover sufficient target-domain words OOV problem for fluent sentence translation. The linear interpolation method performed less than the bias method because of the constant interpolation coefficient that was fixed for every context. Guet al., in 2016, showed that by using different interpolation coefficient for every decoding step, the performance can get better [29].

Utsuro et al., in 2017 implemented a method that pick phrases that contain OOV words using the statistical approach of branching entropy [79]. During training, the selected phrases are replaced

with tokens. Then they are post translated through the phrase translation file (table) of SMT. The phrase selections are done automatically by detecting phrase boundaries applying left/right branching entropy. They evaluated the system with Japanese-Chinese and Japanese-English dataset. For acquiring the word alignment and phrase translation table, they used Moses. The training procedures and hyperparameter choices for NMT models were similar to. This hybrid system achieved 5.6 more BLEU score than baseline SMT. But the system was not compared with NMT models based on sub-word units. They also did not explore the integration of NMT models with the re-ranking framework for minimizing un-translated content. The studies discussed above were conducted on different datasets, using different machine translation tools with different methodologies.

Junczys-Dowmunt et al, in 2016 proposed an integration of attention-based NMT with phrase-based SMT decoder. This work was submitted to the world MT 2016 shared operation, on news translation. They re-implemented the inference step of the NMT model with two phrase-based decoding algorithm- stack decoding and cube pruning. They created a baseline with Moses along with some additional feature functions. They launched a C++ compute unified device architecture (CUDA) format(version) of the inference steps for the neural models, which implemented the framework of Moses log-linear model as several illustration of the same element(feature). They described their proposed algorithm for integrating graphical processing unit (GPU) based soft-attention neural translation models into Moses. They ran the system for Russian-English dataset and it outperformed the best pure neural system submitted in the conference for MT (WMT), 2016 by 1.1 BLEU points. In terms of translation speed handling of OOV problem, their system was slower even than with a basic phrase-based SMT, operating with 24 central processing unit(CPU)threads.

Dahlmann et al., in 2017, proposed a hybrid identifier(search) for attention basis NMT system is [80]. In their system, a target phrase is learned with SMT models that enhance(extends) a suggestion in the NMT beam finder(search) while the source word that is translated by this phrase gets the attention of the NMT model. This hybrid system generated translation partially by phrases and partially by words. NMT beam search generated translation one word at a time in a left-to-right order. The proposed system generates phrases in addition to words. When the source position is focused, only then phrases that do not overlay with at present translated source words are

suggested as a hypothesis. The experiment was conducted on English-Russian ecommerce task and the WMT 2016 German-English task. For the phrase-based baseline, they used a Moses like in-house phrase decoder. NMT system was implemented using Tensorflow deep learning library. With the hybrid system, they achieved 0.3%-1.4% of improved BLEU score for different contextual data. Yet, their approach is less efficient if used without SMT language model because the language model is important for selecting appropriate phrases. Also, they did not use long word phrases which degraded the translation quality.

A cascade hybrid framework combined of NMT and PB-SMT is proposed by Du et al. [8] to improve MT quality. The training data is pre-translated by NMT system. This data is used to build a target-target SMT system. Then this SMT system is tuned with pre-translated development data. Finally, the SMT system produces output translation by re-decoding the pre-translated test data. During the process, they replace the Unknown tokens generated for OOV words with the corresponding source word. They conducted the experiment on Japanese-English and Chinese-English dataset. Their hybrid pipeline system achieved an improved BLEU score of 4.04. Re-ordering and correction of phrases were not considered in their experiment, which could improve the result a bit more. Lack of decoding mechanism for during the process is the major drawbacks of the model in the study. The problem is raised from in appropriate integration and alignment of translation model to the system.

Table 2.2: Summary of related works with their gap

#	Research Title	Year, Authors	Method	Focus	Dataset	Tool	Accuracy	Gap
1	Towards Hybrid Neural Machine Translation for English-Latvian	2016, Marcis Pinnis	Extracts alignment information of unknown token from the NMT systems attention mechanism and translates them with SMT	OOV word translation	Public corpora	DL4MT	28.11	Un able to perform well in case of terminology translation, fixing incorrectly translated non-translatable token
2	Incorporating Discrete Translation Lexicons into NMT	2016, Arthur, Neubig, Nakamura	A hybrid lexicon tailored from handmade and PBSMT is incorporated in NMT	Reducing error for low - frequency content word translation	KFTT and BTEC	Chainer	50.34	Less effectiveness for the hand-made lexicon because it did not cover sufficient target-domain words
3	Patent NMT integrated with Large Vocabulary Phrase Translation by SMT	2017, Long, Kimura, Utsuro, Mitsuhashi, Yamamoto	Using branching entropy, phrases with OOV words are selected and post translated with SMT	OOV word translation	NIST	Moses	43.9	They cannot handle a larger vocabulary lead to unknown word problem

4	The AMU-UEDIN, Submission to the WMT16 News Translation Task	2016, Dowmunt, Dwojak, Sennrich	Functions in Phrase Attention-based NMT, system(models) Feature-based SM	Integrates attention-based neural translation models with phrase-based	WMT 16 newstest	C++/CUDA	25.3	The training and decoding complexity increases proportionally with the number of target tokens
5	Neural Machine Translation Leveraging Phrasebased systems(Models)in a Hybrid Search	2017, Dahlmann, Matusov, Petrushkov, Khadivi	The NMT Beam Search was modified to insert phrasal translations to make a hybrid search technique	Improve translation quality	WMT and E-Commerce	Moses	35.3	They did not use long word phrases which degraded the translation quality
6	Neural PreTranslation for Hybrid Machine Translation	2017, Du, et.al	Training data is pre-translated with NMT, then SMT re-decodes them	Improve translation quality	ASPEC-JE and LDC ZH-EN	GIZA++	39.53	lack of decoding mechanism for During the process, they replace the Unknown tokens generated for OOV words

2.9 Summary

The researchers recognized that the most hybrid MT approach uses a NMT based architecture in a different structure that understands word aligned functionality. Since the NMT process is based on RNN architecture-based, it uses encoder elements and decoder element includes source and target languages to translate the given input sentence to target output. Moses and NMT framework are the most tools used in the model-training of a HNMT system. However, mini-batch stochastic gradient behavior analysis is the simplest way of model-verification ways and BLEU-score metric for model performance testing mechanism.

Furthermore, as the researchers' reviewed results shows that using a hybrid approach of NMT and SMT to exactly(directly) learn the conditional distribution from a bilingual, parallel corpus. Since SMT and NMT have some advantages over one another, the idea of combining them to make a hybrid MT system has come out and emerged as research topic since 2016. Therefore, there should be a strategy that specified either in machine learning strategy or deep learning. Both have an equal contribution to this reviewing result. Moses and NMT frameworks has the highest priority for model construction of the system.

Finally, in the structure of MT, the authors understood the Hybrid MT approach is the more recommended technique in primary studies which is the most widely used mechanism. In testing, is used mostly parallel corpus for checking the model after integrating on the basis of model-testing processes.

CHAPTER THREE

3. Research Methodology

We used experimental research methodology to address the research hypothesis and to accomplish the research objectives. This chapter described the model of HNMT by stating the research direction, systematic review of the research problem, the process model, ways of dataset collection, preparation, and the valuation method. Also, this chapter described the flow of the research methodology by looking into the present NMT evaluation method to get an in-depth understanding of our applicability of the evaluation method, selecting appropriate tool; defining the system prototyping tools of the proposed approach; describing proposed method training, validation, and testing practice, and compare and contrast available development tools to select appropriate tools for the proposed method.

3.1 General Research Methodology

The general research methodology used in this study is experimental approach. The series breaks down the research methodology into six steps[37][38]:

3.1.1 Problem definition

Problem definition, shares best practices about defining the problem. How the right set up is often more important than the choice of algorithm. A few hours spent at this stage in the process can save many weeks' work further downstream, preventing as from solving the wrong problem.

3.1.2 Dataset(corpus)

Preparing the training and testing data is a core part of a machine learning. It's an active not passive part of machine learning research and is one of the most powerful variables to create high-quality machine learning systems.

3.1.3 Evaluation

Before jumping into developing more features and iterating on model architectures, it's important to have a clear plan for how to evaluate the performance of your model. This lesson covers how to evaluate your approach.

3.1.4 Features

This section explain examples of categorical, continuous and derived features and how to choose the right feature for the right model. We also cover areas to look out for such as changing features, feature breakage, leakage and coverage.

3.1.5 Model creation

In this section the next job is to choose the right model for dataset and find the algorithm to implement and train that model. It offers tips about picking, tuning and comparing model

3.1.6 Experimentation(Accuracy testing)

This section clearly shows the model experimentation and the model accuracy testing.

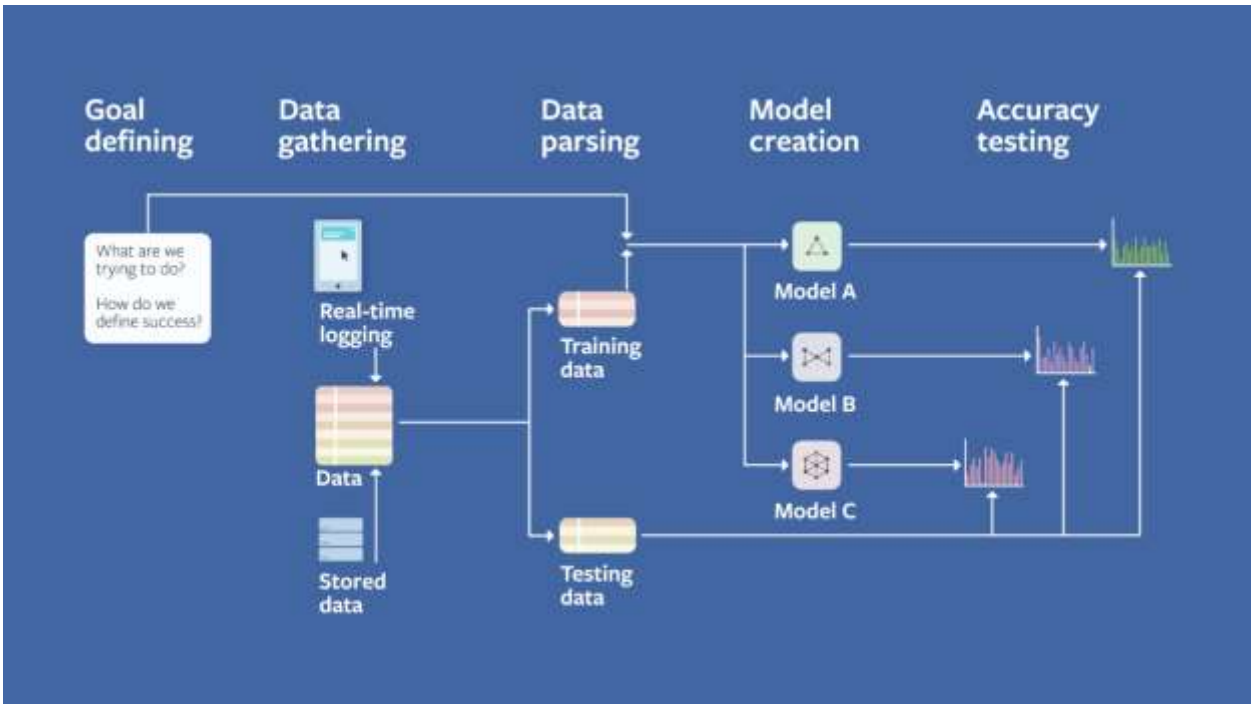

Figure 3.1: General research approach[37]

3.2 Dataset Collection and Preparation

A well collected, sized, and defined text data used to provide or show practical morphological comportment of a language. The collection of text data is, therefore, an essential component in developing an HNMT. Thus, the corpus used in this research collected from religious

web documents, FDRE constitution, FDRE criminal code, Council of Oromia regional state, and English-Afaan Oromo language education materials.

Data Selection Strategies

- Broad enough base of general domain data
  - 300k+ sentence pairs (350k+ for "difficult" language pairs)
  - Full sentences
  - Models(basic vocabulary, grammar, and fluency)
- Supplement with in-domain data or domain-specific

Afaan Oromoo corpus and dataset of limited vocabulary were developed by applying different rules and strategies, and we prepared more than 13000 corpus sentences as listed in Table 3.1

Table 3.1: Source of dataset and size

Name	***Description***	***Domain***	***Aligned data***	***Languages***
Religious documents	Holly bible	Legal	3100 sentence	English and Afaan Oromo
FDRE constitution	Ethiopian constitution	Legal/General domain	1770 sentence	English and Afaan Oromo
FDRE criminal code	Ethiopian criminal code	Legal/General domain	3500 sentence	English and Afaan Oromo
Proclamation	Council of Oromia regional state	Legal/General domain	3780 sentence	English and Afaan Oromo
Education materials	English-Afaan Oromo language education materials	Legal/General domain	1150 sentence	English and Afaan Oromo

Dataset file format, conversion, and preparation

- File format:- text format
- Eight-bit Unicode Transformation Format(UTF-8):- Unicode character encoded
- No formatting markup or Unicode formatting characters
- Parallel corpus sentence aligned:-two files with an identical number of lines

Data preprocessing steps

The goal of data preprocessing is to achieve the highest quality MT output with the available data. Data preprocessing steps are listed as follows.

- Lowercasing
- Punctuation removal
- Stop words removal
- Text standardization
- Spelling correction
- Tokenization
- Bringing the data into parallel corpus format
- Language-specific preparation of the identical dataset for HNMT system training
- Cleaning the corpus
- Unify data from a different source

3.3 Training and Testing the Model

It is challenging to train, validate, and test the model using HNMT for those under corpus resourced language like Afaan Oromo. To overcome massive amounts of corpus preparation problems, the researcher used zero resources[25] machine learning strategies to train and test the model [26], machine learning approach to train and tested on the model. We use hold out cross validation machine learning approach, from the prepared dataset (70% for training and 30% for model testing). Once the English-Afaan Oromo corpus and datasets are prepared, then it passes through the training phase, tuning phase and final testing step.

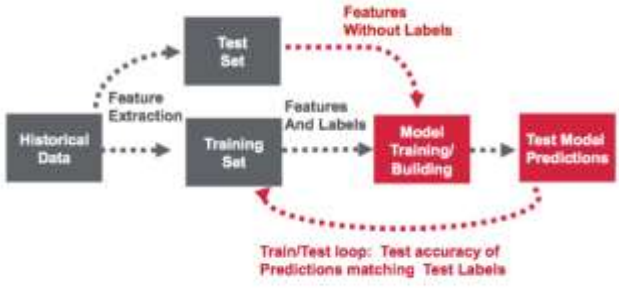

Figure 3.2: Model training and testing work flow[37]

3.4 Tools and Programming Language

Beforehand choosing a tool, the researcher needs to understand various open-source NMT toolkits that are available during the time of writing. The system model uses OpenNMS-tr: capable of tensor flow deep learning framework [27][82]. It is a large-scale, general-purpose, open-source machine learning toolkit nowadays, for the implementation of these HNMT. Open source NM-tr is a preferable toolkit for morphologically abundant languages like Afaan Oromo and in the state-of-art NMT approaches. Compared with other NMT tools, OpenNMS-tr has the best features for hybrid engine integration.

The best feature of OpenNMS-tr tool [82]:

- Randomly complex encoder structure: Implement the encoder by mixing RNNs, LSTM, attention-mechanism, etc. in parallel or in sequence. This study uses this feature to implement LSTM enabled RNN with attention mechanism.
- Hybrid enc-dec architecture: This feature the architecture by integrating self-attention encoder and RNN decoder or vice versa. This study uses this feature used to construct the hybrid model.
- Neural source-target alignment: Used to train with guided alignment to constrain attention vectors and output alignments as part of the translation application programming interface. OpenNMT-tf also implements most of the techniques commonly used to train and evaluate RNN sequence models, such as: automatic evaluation during the training, multiple decoding strategy, N-best rescoring, gradient accumulation, scheduled sampling and checkpoint averaging. This study uses this feature for NN source to target sentence alignments.
- Multi-source training: source text and Moses translation as inputs for machine translation. This study uses this feature to train SMT components with NMT model.
- Multiple input format: Used to handle multiple source text input format. This feature support text of mixed word/character embeddings or real vectors serialized in TFRecord files. This study uses this feature word-to-vector embedding.

- On-the-fly tokenization: This feature apply advanced tokenization dynamically during the training and detokenize the predictions during inference or evaluation. This study uses this feature word tokenization during the training and detokenization during evaluation.
- Model adaptation: This feature enable the model to adapt with a few training epochs. This study uses this feature to specialize a model to a new domain in a few training steps by updating the word vocabularies in checkpoints.
- Automatic evaluation: support for saving evaluation predictions and running external evaluators (e.g. BLEU). This study uses this feature for automatic evaluation during model training and modeling using BLEU-score metric.
- Hybrid precision training: take advantage of the latest minibatch stochastic gradient decent, NVIDIA, ADAM, Adadelta optimizations and activation functions to train the models with half-precision floating points. This study uses this feature minibatch stochastic gradient decent, ADAM optimizer and softmax activation function for minimum error loss in both training and testing phases.

3.5 Discussion with NLP Expert

To more insight into the research, a thorough conversation made with NLP experts of the SMT, Afaan Oromo language, and NMT experts. The discussion helped to gain the essential concept of how the language translated of the word in Afaan Oromo text generated in machine translation. The debate used as a primary input in addition to secondary information reviewed in the course of the study.

3.6 System Prototyping Tool

The model prototype, to develop the proposed system and its experimentation, python language was used. We used an anaconda environment to implement the proposed solution model prototype development and implementation/experimentation. Why because anaconda is a python grounded platform that includes most data-science packages; scikit-learn, pandas, NumPy, SciPy, google machine learning launching pad(platform), Keras and TensorFlow deep learning framework. It supports packaged with pip install tool (conda), Anaconda browser(navigator) for spyder and jupyter. Anaconda Python is an open-source sharing of the comparatively programming languages. Also, used in data science, machine learning, deep learning, applications aiming at simplifying

package management and deployment. The system uses a python anaconda environment for text translation and the model uses python libraries and packages used primarily for scientific computing and technical computing. Also, through python, the system can quickly call open-source machine translation API function and to access translation components layers. The most common definition introduces a tensor as a multilinear function from a product of vector spaces to the real numbers [28]. Software/framework used in this tool:

- **Python**: is a programming language used to implement model experiment and prototype development. It's compatible with deep learning frameworks and support pip like install features. MT libraries and frameworks are easily added to the model using this tool. This study uses the tool to implement the proposed model construction.
- **TensorFlow**: is deep learning framework used to design, build, train deep learning models and to create the model architecture. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in machine learning and developers easily build and deploy ML-powered applications. This study uses the framework to construct NN with hidden layers.
- **Pandas**: Panda's data frames are the most widely used in-memory representation of complex data collections within Python that allows data preprocessing. High-performance, easy-to-use data structures, and data analysis tools. This study uses it for data reading, manipulation, writing and handling the data frame.
- **Scikit-Learn**: A set of python modules for machine learning and data mining. This study uses it for feature extraction and training and testing model.
- **Numpy:** Array processing for number, strings, and objects. This study uses it for handling converting the text to numeric data for features and training and testing the model.
- **Natural language toolkit (NLTK):** NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries and package. This study uses the framework for text processing libraries.

3.7 Evaluation Metrics

To judge the quality of an MT, one measures of its closeness to one or more reference human translations according to a numerical metric. We use BLEU-score MT evaluation metrics to evaluate the proposed English-Afaan Oromo HNMT target language with reference language. The cornerstone of the BLEU-score parameter is a precision-oriented measure that counts the n-gram matches between the MT output and the gold standard references, quick, language-independent, a numerical translation metric closeness [31]. There are different MT metrics frequently used for optimization and evaluation: translation edit rate(TER) and Meteo MT evaluation method they are recall oriented which are used for specific morphology structure not preferable to evaluate the proposed model. There are several kinds of strategy within this category: introducing a corpus to build the HNMT system, introducing corpus-based tools to weight the HNMT output, and carrying out a statistical post-editing of an HNMT output[48].

BLEU scorebased on the concept that good translations should contain words and phrases from references, the BLEU metric scores suggestions regarding to form, N-gram precision. For each n-gram dimension up to N, (N= 4, BLEU 4 variant), a separate precision score P_n is computed as the percentage of n-grams, in the assumption also located in the reference. The penalty (B) is based on the duration of the translation hypothesis E′ and reference E [24][30][36][39][44].

$$B(E', E) = \begin{Bmatrix} 1 & if|E| > |E'| \\ e^{1-\frac{|E|}{|E'|}} & if|E'| \leq |E| \end{Bmatrix} \; BLEU_N(E', E) = B \; x \; exp\left(\sum_{n=1}^{N} \frac{1}{N} \log P_n\right)$$

Equation 3.1 BLEU score metrics

CHAPTER FOUR

4. Proposed Solution

4.1 The Proposed Solution System Architecture

The architecture of HNMT is the familiar of NLP that handle OOV, long sentence, and absence of decoding problems based on RNN architecture. The proposed model uses the HNMT technique on the bases of the NMT architecture. The proposed solution experimented on English-Afaan Oromoo prepared parallel datasets that translate English written text into Afaan Oromo text using deep learning approach. The proposed HNMT system takes the three components from SMT, and it depends more on the NMT model. The proposed model incorporates SMT components to avoid the problems, as shown in section 1.3. The integration components are word reward feature, n-, translation model, gram language model phrase-based NMT model. The RNN encoder-decoder is trained on a parallel corpora dataset and performs an end-to-end translation. On the other hand, under the existing architecture, it is not an easy task to get better translation quality by incorporating other translation components.

HNMT system is an MT model that uses NMT techniques to automatically translate text written from the English language to Afaan-Oromoo language. The new system incorporated with the assimilation of SMT finest components. Specifically, as for many other NLP tasks, the system uses attention-based RNN. The necessary elements of better components that they operate on sequences: take an input sequence, construct an output sequence. The proposed model incorporates SMT and NMT with different features as a one system architecture under the log linear framework. Why, because to handle the problems described in section 1.3, the issues shown in the NMT system then alleviated by incorporating with SMT components as a baseline of NN architecture as we call HNMT. The proposed model classifies the system architecture components into three layers, as pictured in Figure 4.1. The problems stated in section 1.3 handled by incorporating the layers as one architecture as follows:

- Encoder layer
- HNMT layer
- Decoder layer

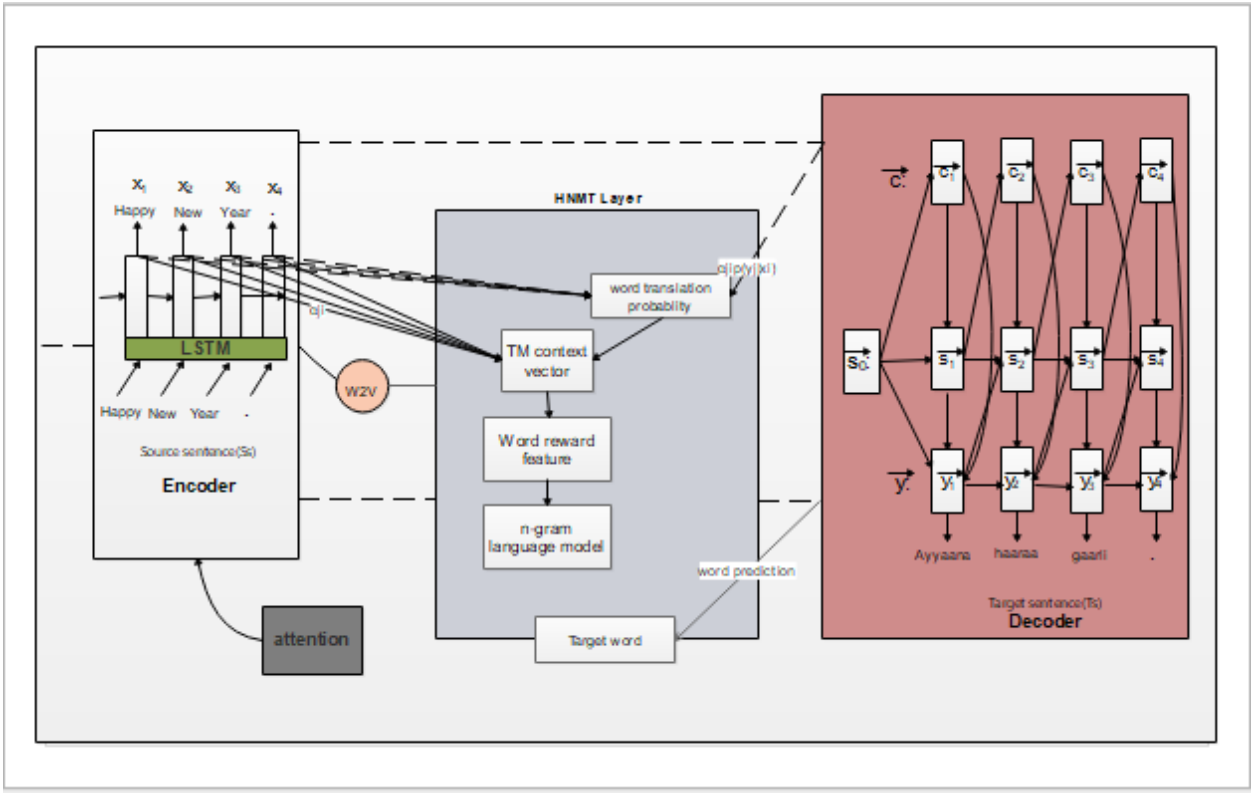

Figure 4.1. HNMT system architecture

Algorithm 1: Incorporation of the two model

Input: source language Ss
Output: target language Ts with best prediction translation quality

1: *Begin*
2: *Load dataset //Open text file for reading*
3: *Input source sentence Ss*
4: *Encode one line at each time Si*
5: *Encode word by word xn to Ss*
6: *map xn word to vector xn→w2v //W2V*
7: *for Si to Sn*
8: *apply atten Sn*
9: *feed Si to HNMT layer*
10: *while(Sn tooknized SOS)*
11: *Predict P(e_i\\t_i) // predict word translation probabilities*
12: *Score P word translation*
13: *Store the sate//LSTM*
14: *Decode P→Ts*
15: *match Ts to Ss //The output match to the target*
16: *check if Ts match to Ss*
17: *Return Ts EOS*
18: *else*

19: *go to feed Si to HNMT layer*
20: *End*

The given source sentence *Ss* is is encoded first to provide the input for the encoder layer. The encoder, part encodes one line at a time source sentence *Si* for all loaded datasets. Then the encoder, encodes each sequence of the source sentence word-by-word. The first layer encodes x_n until *Ss* loaded datasets end of the sentence-of-sentence (SOS) tokenized for each *Si* to *Sn* by mapping $x_n \rightarrow w2v$ the word and applying attention to the token. The encoder layer then feeds *Si* the second layer loop until *Sn* tokenized to SOS. After feeding *Si* predict $P(e_i \backslash t_i)$ for each word translation probability of selecting the best translation word translation probabilities and score p-word translation. Score the state and decode the best word translation probability scored by *P*, $P \rightarrow Ts$. Finally, *check if Ts match to Ss* to the output of decoded target sentence match to the target until Ts return to end-of-sentence (EOS) token. In the decoding step, if *Ts match to Ss* does not give the same vector, go back to *Si* to the second layer.

4.2 Components of Hybrid Neural Machine Translation Layers

4.2.1. Translation model

The translation model developed a word-mapped parallel corpus with standard catchphrase based SMT technique and employed to score word. The composed word and phrase translations tables help us make fluent source sentence translation.

4.2.2 Word reward feature

The necessary of this feature is to regulate the length of translation(insufficient translation difficulties). Word reward feature also improve insufficient translation problem for the extended decoder reading.

4.2.3 Language model

The aims of the language model, as shown in algorithm 2, works to improve the local fluency, which trained on the target parallel sentence, which works as follows. How likely words follow in a particular sequence.

Algorithm 2: Define language model for English and Afaan Oromoo

Input: to train language model for both English-Afaan Oromoo language
Output: set $p_w(f_w|e_w)$ to normalized $score(f_w|e_w)$

1: train language model for English p_{LM}
2: initialize word score p_w uniformly
3: iterate
4: set $score(f|e)$ to 0 for all dictionary entries (f,e)
5: for all Afaan_oromoo sentences fs = (f_1, …, f_n)
6: for all possible English sentence translations e_s
7: compute sentence transl. probability $p_s(e_s|f_s)$
8: by $p_w(f_1|e_1)p_w(f_2|e_2)…p_w(f_n|e_n)$.
9: .$p_{LM}(e_1)p_{Lm}(e_2|e_1)…p_{LM}(e_n|e_{n-1})$
10: endfor
11: normalize $p_s(e_s|f_s)$ so their sum is 1
12: for all sentence translations e_s
13: for all words e_w in e_s
14: add $p_s(e_s|f_s)$ to $score(f_w|e_w)$
15: enfor
16: enfor
17: endfor
18: for all translation pairs (f_w, e_w)
19: set $p_w(f_w|e_w)$ to normalized $score(f_w|e_w)$
20: endfor
21: enditerate

4.2.4 Word to Vector

Word to vector(W2V) embeddings(mapping) that allow the model to agree with word representation, words with the same context to have the same illustration. W2V embedding uses to models the proposed solution: CBOW and skip-gram model[4].

Algorithm 3: W2V

Input: source word
Output: mapping word

1: Begin
2: CBOW model:
3: Input: w(t-2)
4: w(t-1)
5: w(t-n)
6: Projection: sum(w(t-n)
7: Output: w(t)
8: skip-gram model:
9: Input: w(t)
10: Projection: sum(w(t-n)
11: End

4.2.5 LSTM

To handle the transforms, RRN the present information entirely by applying the LSTM to the model and handle long term dependency for variation of source sentence input[57].

4.2.6 Word Translation Probability

We define a method in the, target language using a parallel corpus to estimate the likelihood of word translation. It takes the $p_w(f|e)$ form, the overall probability of translating the English word e as f, regardless of circumstance.

Algorithm 4: Word translation probability

Input: to score word translation probability for both English-Afaan Oromoo language
Output: set $p_w(f_w|e_w)$ to normalized $score(f_w|e_w)$

1: train translation probability for English p_{LM}
2: initialize word translation probabilities p_w uniformly
3: iterate
4: set $score(f|e)$ to 0 for all dictionary entries (f,e)
5: for all Afaan_oromoo sentences fs = (f_1, …, f_n)
6: for all possible English sentence translations e_s
7: compute sentence transl. probability $p_s(e_s|f_s)$
8: by $p_w(f_1|e_1)p_w(f_2|e_2)\ldots p_w(f_n|e_n)$.
9: .$p_{LM}(e_1)p_{Lm}(e_2|e_1)\ldots p_{LM}(e_n|e_{n-1})$
10: endfor
11: normalize $p_s(e_s|f_s)$ so their sum is 1
12: for all sentence translations e_s
13: for all words e_w in e_s
14: add $p_s(e_s|f_s)$ to $score(f_w|e_w)$
15: enfor
16: enfor
17: endfor
18: for all translation pairs (f_w, e_w)
19: set $p_w(f_w|e_w)$ to normalized $score(f_w|e_w)$
20: endfor
21: enditerate

4.2.7 Attention Mechanism

The attention mechanism used in the proposed model that inputs two sentences, shifts them into a pattern the words of one side sentence formulate the columns. The terms of the other side sentence express the rows, and finally, it constructs pairs, classifying relevant context. The primary understanding at the back of attention mechanism is that at any situation, the decoder is producing a specific output word. The decoder looks at the concentration of information around the whole words, in the source sentence. Then determines under which terms in the source sentence are appropriate to produce the proper output word [8].

$$\alpha_{ts} = \frac{\exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'=1}^{S} \exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)}$$

Equation 4.1: Attention weights

$$\boldsymbol{c}_t = \sum_s \alpha_{ts} \bar{\boldsymbol{h}}_s$$

Equation 4.2: Context vector

$$\boldsymbol{a}_t = f(\boldsymbol{c}_t, \boldsymbol{h}_t) = \tanh(\boldsymbol{W}_c[\boldsymbol{c}_t; \boldsymbol{h}_t])$$

Equation 4.3: Attention vector

4.3 Encoder-Decoder Layer

This portion briefly illustrates the architecture of RNN encoder-decoder layers. NMT is a NN that directly models the conditional probability *P(y|x)* of the translating a given sentence, *x1*……, *xn*, to the target statements, *y1*……., *ym*. The NMT architecture uses the two LSTM memories to reset LSTM and update LSTM. Reset LSTM avoids the information of network some previous hidden states, which may be yelling for the existing layer state. Also, the second update, LSTM, manages the degree of information transferred to the current state from the previous state.

4.3.1 Recurrent Neural Network

The RNN, as we have explained in section 1.3, structured using two networks, components. An encoder part and decoder part with four layers of architecture.

4.3.1.1 RNN Encoder

The RNN encoder encodes the given source sentence s, $\tilde{s}$= $s1, s2, \ldots, sn$ into a sequence of the vector. The encoder is a BiRNN with two hidden activated layers. At the encoding stage, the encoder primarily designs the given source sentence $\tilde{s}$ into word-vectors $\vec{v}$ = (v1, v2, ..., vn), vi$\in$ $R^K{}_s$, where K_s is the dictionary volume of the source language. Then the network bring up to updates the hidden state x_{enc}^{i} at each step by

$$\boldsymbol{x}_{\boldsymbol{enc}}^{\boldsymbol{i}} = \boldsymbol{f}_{\boldsymbol{enc}}(\boldsymbol{s}_{\boldsymbol{i}}, \boldsymbol{x}_{\boldsymbol{enc}}^{\boldsymbol{i-1}})$$

Equation 4.4: Steps to update the layer

In Equation 4.4 f_{enc}, is the activation function. The activation function; for f_{enc} is *tanh* function. The update hidden state, of the layer is the combination of forwarding($\vec{v}$) and backward ($\tilde{v}$) hidden states computed according, to the given(source) sentence, as described in Equation 4.5.

$$x_{enc}^{i} = \left[\overrightarrow{x_{enc}^{l} P}\right], \left[\overleftarrow{x_{enc}^{l} P}\right]^{P}$$

Equation 4.5: Backward and forward hidden state

4.3.1.2 RNN Decoder

The RNN decoder decodes and generates the given source sentence s, $\tilde{s}= s1, s2, \dots, sn$ into the target translation $\dot{y} = y1, y2, \dots, ym$ depending on the sequence of vector and target word previously generated by an encoder. In Decoding phase, the possibility of the returned succession handled as follows

$$p(\vec{y}) = \prod_{j=1}^{j} p\left(y_j | \{y_{j-1}, y_{j-2}, \dots, y_1\}, \vec{s}\right)$$

Equation 4.6: The probability of the output sequence

$$= \prod_{j=1}^{j} f_{dec}\left(z_{dec}^{j}, y_{i-1}, h_i\right)$$

Equation 4.7: The probability of the decoding output hidden state

The hidden state, activation function, in the decoding layer indicated in Equation 4.8 where, f_{dec}^{i} hidden state at step j, which is computed by,

$$z_{dec}^{j} = f'_{dec}\left(z_{dec}^{j-1}, y_{i-1}, h_i\right)$$

Equation 4.8: Hidden state of decoding layer for the activation function

In Equation 4.8 f_{dec} and f'_{dec} The two features are nonlinear activation functions for the segment. The context vector h_j is the encoder used the estimate, as a calculating weighted sum, hidden states of the layer:

$$h_j = \sum_{i=1}^{P_x} \alpha_{ji} x_{enc}^{i}$$

Equation 4.9: Decoder weighted sum for hidden state

In Equation 4.9 where, α_{ji} is the weight observed as a cordination measure of how appropriately a target word y_j is translated from a source word x_i.

4.4 Hybrid Neural Machine Translation Layer

The researcher consider that incorporating the SMT component could assist get better translation quality for the systems(NMT). Figure 4.1. to exemplify proposed idea. The general model architecture of HNMT explained under algorithm 5 for every three layers. At each phase to estimate a focus(target) word in toting up to the likelihoods (probabilities) projected by BiRNN, we combine the three model. The translation file(table), calculate approximately (estimating)from the word-aligned bilingual dataset. The table can enhance word(lexical) change and translate the small- rate words, which taken or tokenized as unidentified lexicons. The language model enables full use of the target parallel dataset to advance narrow fluency. Log-identity framework is used to incorporate these practically active components.

Algorithm 5: Hybrid architecture of the system

Input: source language Ss
Output: target language Ts with best prediction translation quality

Begin
Load dataset //Open text file for reading
Input source sentence Ss
Encode one line at each time Si
  Encode word by word xn to Ss
    map xn word to vector xn→w2v //W2V
     for Si to Sn
      apply atten Sn
      feed Si to HNMT layer
          while(Sn tooknized SOS)
          Predict P(e_i\t_i) // predict word translation probabilities
           Score P word translation
            Store the sate//LSTM
              Decode P→Ts
                match Ts to Ss //The output match to the target
              check if Ts match to Ss

17:	*Return Ts EOS*
18:	*else*
19:	*go to feed Si to HNMT layer*
20:	*End*

The given source sentence *Ss* is is encoded first to provide the input for the encoder layer. The encoder part, encodes one line at a time source sentence *Si* for all loaded datasets. Then the encoder, encodes each sequence of the source sentence word-by-word. The first layer encodes x_n until *Ss* loaded datasets end of the sentence-of-sentence (SOS) tokenized for each *Si* to *Sn* by mapping $x_n \rightarrow w2v$ the word and applying attention to the token. The encoder layer then feeds *Si* the second layer loop until *Sn* tokenized to SOS. After feeding *Si* predict $P(e_i \backslash t_i)$ for each word translation probability of selecting the best translation word translation probabilities and score p-word translation. Score the state and decode the best word translation probability scored by *P*, $P \rightarrow Ts$. Finally, *check if Ts match to Ss* to the output of decoded target sentence match to the target until Ts return to end-of-sentence (EOS) token. In the decoding step, if *Ts match to Ss* does not give the same vector, go back to *Si* to the second layer.

4.4.1 Features of Hybrid Neural Machine Translation

The proposed model includes the RNN encoder-decoder, bi-directional word translation probabilities, language model, and word reward feature, as described below.

4.4.1.1 RNN Encoder-Decoder

The BiRNN encoder-decoder layer is a powerful layer [8]. This element is the conditional probability projected by the NMT architecture. The BiRNN that predicts the focused target word depending upon the source sentence, and the word previously generated target words for the model.

$$H_{rnn} = \sum_{j=1}^{J} \log(g(y_{j-1}, s_j, c_j)$$

Equation 4.10: BiRNN that predicts the destination word

4.4.1.2 Bi-directional Word Translation Probabilities

The bi-directional term(word) translation possibilities predict vocabulary translation probabilities of the language. At every phase of decoding, we predict the lexical translation possibilities among target contestants of the source language and the parallel source words.

$$H_{tp1} = \sum_{j=1}^{J} \sum_{i=1}^{I} \alpha_{ji} \log(p(y_i|x_i)$$

Equation 4.11: Prediction lexical translation probabilities

$$H_{tp1} = \sum_{j=1}^{J} \sum_{i=1}^{I} \alpha_{ji} \log(p(x_i|y_i)$$

Equation 4.12: Soft alignment among target word and predicted by BiRNN

The soft alignment of the target word and predicted word in Equation 4.12 shows that estimated by BiRNN. Where, α_{ji} are the weighted of soft alignments among the target word y_i and correlated source words, predicted by the BiRNN encode-decoder. $p(y_i|x_i)$ and $p(x_i|y_i)$ is the word translation likelihoods predicted from the word-aligned bilingual dataset. The word alignment trained with the SMT system with the prepared dataset. The word translation probabilities processed as formulated in Equation 4.13 as follows:

$$p(x_i|y_i) = \frac{E(x,y)}{\sum_{x'} E(x',y)}$$

Equation 4.13: Word translation possibilities

$$p(y_i|x_i) = \frac{E(y,x)}{\sum_{y'} E(y',x)}$$

Equation 4.14: Occurrence rate of the parallel words

The occurrences rate of the identical words, within the source, and target words computed as in Equation 4.14 where $\mathrm{E(x, y)}$ and $\mathrm{E(y, x)}$ are the rate existing of the matching words x, y, and y, x.

4.4.1.3 Language model

The language model trained on target monolingual corpus. Thus language model enables us to become used of a vast scale monolingual dataset of the target goal language.

$$L_{lm} = \sum_{j=1}^{J} \log(p(y_j|y_{j-1}, \dots, y_{j-n+1}))$$

Equation 4.15: Language model

4.4.1.4 Word Reward Feature

The feature is the number amount of words in the target sentence, which possibly will monitor an overfitting length of word translation.

$$L_{wp} = \sum_{j=1}^{J} 1$$

Equation 4.16: Word reward feature

The proposed HNMT model contrasted with the previous RNN model, we enhance extra components for every state created by the BiRNN decoder in case of decoding. The translation produced after the last state with the top total score generated.

4.4.2 Handling the out of Vocabulary Problem

As stated in section 1.3, the NMT encoder-decoder mostly tackles an OOV issue. The preprocessing approach did not help from the contextualized information of the source, and target word in case of decoding. Instead, we used and add a word probability translation table, that automatically obtained from a word- associated bilingual dataset, to translate the OOV words in decoding phase. To construct the accurate translation for the OOV word, firstly, we dig out its parallel source word. The alignment is done based on the alignment possibilities predicted by the BiRNN model, and the unknown token “UNK” representation suggests to the source word which could not listen to the vocabulary. Then we gain translation outputs from the word translation vocabulary table. The last generated of word regulated by the proposed log-identity model, by taking into account the rich contextual information on both sides of the source, and target in case of decoding.

4.4.3 Decoding Mechanism

The BiRNN decoder operates on a beam search algorithm[83] to obtain a translation that enlarges the conditional possibilities of interpretation, handled as

$$\hat{y} = \underset{\tilde{y}}{argmax}\, p(\tilde{x}|\tilde{y})$$

Equation 4.17 BiRNN decoder translation possibilities

The source sentence inputted to the decoder, the decoder, estimate, and return target sentences word through the word. The decoding step starts with the first state. At every time step, the decoder picks out the beam size on K-best states and enlarges them up to the EOS token generated. The last translation produced by mapping outback from the previous final state created with the top score. We modified the decoder of TensorFlow to compute translation under the log-identity framework. In every decoding state, the TensorFlow merely uses the record generated by BiRNN to select the K-best score from the target dictionary. In the proposed decoder part, we additionally compute the word translation possibilities, the language model record, and the actual sentence length at every state. For every word, for the target sentence of vocabulary, we estimate a total full score. Then we use the record to produce the best score list from the time when new features included than the previous NMT model. The weights of the log-identity models are tuned using the benchmark BLUE score algorithm. To better the decoder, the proposed model uses the priority queue algorithm to select the most excellent score to be added rather than adding all scores at every state.

4.5 Neural Machine Translation Log-linear model

We implement the log-linear model to incorporate the three models into the NMT system. The most popular incorporation model is a log-linear model in SMT was introduced by [84].

$$p(\tilde{e}|\tilde{f}) = \frac{\exp(\sum_{i=1}^{m} \lambda_i H_i(\tilde{f}|\tilde{e}))}{\sum_{\bar{e}'} \exp(\sum_{i=1}^{m} \lambda_i H_i(\tilde{f}|\bar{e}'))}$$

Equation 4.18: Log-identity model

where, $H_i(\tilde{f}|\tilde{e})$ is a feature function and λ_i is the weight add. The benefit of the log-linear model, is that features can be easily inserted on it [84]. A standard phrase-based SMT [6] typically

contains 8 elements: the word penalty, and the phrase penalty, the reordering model, the bi-directional lexical weights $p(f|e)$ and $p(e|f)$, the bidirectional translation probabilities $p(f|e)$ and $p(e|f)$the language model. These features have been proven effective to improve translation quality. We use Figure 4.1 to clarify our idea. At each step to predict a target word, in addition to the probabilities predicted by RNN, we add an n-gram language model and a word translation table model. The translation table, estimating from a word-aligned bilingual corpus, can improve lexical translation and translate the low-frequency words which are taken as unknown words. The language model can make full use of target monolingual corpus to improve local fluency.

4.6 Mini Batch Gradient Descent

Mini batch gradient descent algorithm used for calculating the weights of each layer. It allows the model by making an estimation on training data and employing the error on the estimate to update the model to minimize the error. To calculate model parameters of coefficients that enable reduce the error rate of the model, on the trained dataset. The algorithm operates by making a difference to the model that shifts it a mass of a gradient. Moreover, the mini-batch gradient descent works by formulating a different slope of errors low to the nearest a minimum error value [85] [86].

The pseudocode for mini batch gradient descent work as follows[85]*:*

```
model = initialization(...)
n_epochs = ...
train_data = ...
for i in n_epochs:
train_data = shuffle(train_data)
X, y = split(train_data)
predictions = predict(X, model)
error = calculate_error(y, predictions)
model = update_model(model, error)
```

Algorithm 6: Mini-batch Stochastic, Gradient Descent Algorithm

Data: Training data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]^\top$, $\mathbf{x}_i \subset R^{m+1}$, $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_n]$, $\mathbf{y} \subset R^K$, $0 < \gamma \leq 1$, threshold $b < 1$, , converged = True

Result: fitted $\mathbf{W}$

Initialize $\mathbf{W}$;

$t = 0$;

while *not converged* **do**

  Create K' mini-batches as $\mathbf{B} = [B_1, ..., B_{K'}] = \text{Shuffle}(\mathbf{X})$;

  for $k \in \{1, ..., K\}$ **do**

    $\mathbf{g} = \frac{1}{|B_k|} \sum_{i \in B_k} \left[\frac{\partial \mathcal{L}(f(\mathbf{x}_i; \mathbf{W}), \mathbf{y}_i)}{\partial \mathbf{W}} + \frac{|B_k|}{n} \lambda \frac{\partial R(\mathbf{W})}{\mathbf{W}} \right]$;

    $\Delta \mathbf{W} \leftarrow \mathbf{g} + \alpha \Delta \mathbf{W}$;

    $\mathbf{W} \leftarrow \mathbf{W} + \Delta \mathbf{W}$

  end

  t = t + 1

end

The mini-batches B_k include a set of data indexes to compute the error. Then, update model parameters based on the gradient average to reduce the variance of the gradient. The mini-batch stochastic gradient descent (SGD) works similar to standard SGD ; the process starts by initializing the model parameters, entering a parameter updating loop and is ended by evaluating the validation set. Compared to standard SGD, the iterative process will continue over the mini-batches and the model parameters will be updated after processing each mini-batch. In each iteration, a set of mini-batches are processed that represent the whole training data inone epoch

4.6.1 Adam Optimizer

The proposed solution uses Adam optimization algorithm. Why because Adam prdict the first (mean) and second (variance) to regulate the coefficients corrections which is suitable optimization algorithm for the sequential types of data format (e.g.NMT) and suitable optimizer used in Tensorflow and Keras framework. Adam optimizer is very tolerant of initialing the learning rate which is known as alpha and more training parameters as [85] .

```
__init__(
  learning_rate=0.001,
  beta1=0.9,
```

beta2=0.999,

epsilon=1e-08,

use_locking=False,

name='Adam')

Algorithm 6: Adam deep learning optimizer

Input: initialize the training rate

Output: to return stochastic factual operation through parameters θ_t

1: Expect: α: Stepsize

2: Expect: $\beta_1, \beta_2 \in [0,1)$: decay rates for the instant predict

3: Expect: $f(\theta)$: Stochastic factual operation through parameters θ

4: Expect: θ_0: Primary parameter vector

5: $m_0 \leftarrow 0$(Initialize first instant vector)

6: $v_0 \leftarrow 0$(Initialize second instant vector)

7: $t \leftarrow 0$(Initialize timestep)

8: while θ_t not converged do

9: $t \leftarrow t+1$

10: $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$

11: $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$

12: $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g^2{}_t$

13: $\hat{m}_t \leftarrow m_t/(1-\beta^t{}_1)$

14: $\hat{v}_t \leftarrow v_t/(1-\beta^t{}_2)$

15: $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \in)$

16: end while

17: return θ_t

4.7 Activation Function

We use activation functions, to formulate the result of a neuron forward. The result gained by the neurons of the subsequent layer to the neuron is associated with the fit for the final layer built-in[87]. The activation function used in the proposed model is the softmax activation function[88]. The main importance of softmax activation is when each separate result of a neuron in output layer calculated based on

$$y_i = \frac{e^{z_i}}{\sum_{i=0}^{m} e^{z_i}}$$

Equation 4.19 Softmax activation function

4.8 Loss Error Function

This function required to calculate how close an actual NN is to the ideal coefficient for the period of the training process. To compute the loss of the trained data, we used the categorical cross-entropy. Categorical cross-entropy loss is the combination of softmax function and cross-entropy computed as. Categorical cross-entropy evaluates the distribution of the estimation of the activations in the final output layer, one for every feature through true class sharing. In case of the likelihood of the true feature is fixed to 0 and 1for the other classes. To put the class infrequently, the true class stands for as a one- strong encoded vector, and the nearer the model's final output results are to that vector, the minor the loss[89][90].

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}\mathbf{1}_{y_i \in C_c}\log p_{model}[y_i \in C_c]$$

Equation 4.20: Categorical cross-entropy loss

CHAPTER FIVE

5. Implementation of the Proposed Solution

5.1 Overviews

As described earlier in chapter 3, data gathered from the website, which is from legal and government domains. The data required during model training, validation, and testing in HNMT modeling help to evaluate the efficiency of the system. The system accuracy calculation for the BLUE score during the translation period. These data specified at design time according to the interest of owners and developers. At this time, the development of the HNMT system which can learn from previously learned data, which uses deep machine learning approaches to improve unpredictable circumstances. Such a model also included in systems like in expert systems, NLP, speech recognition, computer vision, robotics, and in the software such as grammar checking and writing platform framework and tool.

The implementation phase gives a demonstration on how to design and implement a simple web-based MT system that translates the given source sentence to the target sentence. The system can translate the given source language to the target language by learning from previously learned dataset words. The architectural elements of the MT system are by learning its working environments. Additionally, the implementation part shows how to code (how to make practical/ feasible) each of the exists components in the HNMT architecture. It also shows that how to interrelate(interact) before and after source and target word translated discussed with the corresponding implementation.

5.2 Implementation Environment

5.2.1 Software (framework, package)

- Anaconda Environment: it uses as scripting editors for python programming languages (2019.1.3 community edition). The environment simplifies the method and process of incorporating two different models. So, it's a good approach instead of writing python language on a separate tool rather than writing independently on different tools. The followings are runtime carrying out execution exists in python environments. We used an anaconda environment for the proposed solution model prototype development and implementation. The

reason that because of, as stated in section 3.5, anaconda, is a Python-based platform that curates major data science applications (scientific computing). The packages easily imported in Python, such as pandas, scikit-learn, SciPy, NumPy, and google's machine learning platform, TensorFlow, and Keras deep learning framework. The package comes with conda (a pip like an install tool), anaconda navigator for a GUI experience, Jupiter, and spyder for an ID.

- Tensorflow: a comfortable framework for deep learning, to construct the network layers, preprocessing the data, build the model, and train and estimate the model.
- nlkt: NLP package compatible with python environment
- Python interpreter: python 3.6 python.exe
- Jupyter: a web application that allows us to create and run IPython in the browser.

5.2.2 Hardware

- Laptop
  - Processor: Intel® Core™ i3-2350M CPU @ 2.30GHz 2.30 GHz
  - Installed memory (RAM): 4.00GB
  - System type: 64 –bit operating system, x64-based processor
- Desktop computer
  - Processor: Intel® Core™ i5-6500 CPU @ 3.20GHz 3.19 GHz
  - Installed memory (RAM): 4.00GB
  - System type: 64 –bit operating system, x64-based processor

5.2.3 Programming Language

- Python – for implementation of the machine learning algorithm and MT approaches because it has better scientific and numerical libraries that offer a working environment of analysis of data in terms of number. In the next part, the prototype of the MT system would describe briefly according to the proposed solution. The system is a simple web-based application that translates text for users either in a single sentence or paragraph. The system consists of two essential features that have internal elements and external elements as a combined component structure. These are server elements, dataset learning elements, and main user interface

elements. The high-level architecture of the prototype shown in Figure 5.2 High-level architecture of the prototype.

5.3 Prototype Modeling

5.3.1 Hybrid Neural Machine Translation (System) Modeling

The proposed solution uses bi-directional four NN layers for prototype implementation. Two neural networks for input and two NN for output.

- encoder_inputs,= [batch_size, max_encoder_time]
- decoder_inputs,= [batch_size, max_decoder_time,]
- decoder_outputs,= [max_decoder_time, batch_size]
- Embedding layer: vocabulary size V chosen. All remaining words changed to "UNK" token, and all the rest come to be the same embedding.
- Encoder [forward RNN and a backward RNN], and each has 1000 hidden units.
- Decoder [1000 hidden units].
- Batch size[].
- Epoch[30]
- LSTM enabled
- Word embeddings [620-dimensional].
- Mini-batch stochastic gradient descent to train the networks. Each contains 20 sentence pairs.
- Adam optimizer rate of parameters (learning_rate=0.001, beta1=0.9, beta2=0.999, epsilon=1e-08).
- The HNMT model trained through about 850,000 updates for the RNN encoder.
- For model training and testing, central processing unit(CPU) configured a single machine.
- BLEU-4(4-gram BLUE score) for evaluation

Table 5.1: List of module setup in HNMT model

Parameter	Setup	Parameter	Setup
"attention":	"scaled_luong"	"num_train_steps":	30,
"attention_architecture":	"standard",	"decay_scheme":	"luong234",
"batch_size":	10,	"num_units":	512,

"colocate_gradients_with_ops ":	true,	"optimizer":	"Adam",
"dropout":	0.2,	"residual":	false,
"encoder_type":	"bi",	"share_vocab":	false,
"eos":	"</s>",	"word_option":	"",
"forget_bias":	1.0,	"sos":	"<s>",
"infer_batch_size":	10,	"src_max_len":	50,
"init_weight":	0.1,	"src_max_len_infer":	null,
"learning_rate":	3, 1.0, and 0.0001	"steps_per_external_eval":	null,
"max_gradient_norm":	5.0,	"steps_per_stats":	100,
"metrics":	["bleu"],	"tgt_max_len":	50,
"num_buckets":	5,	"tgt_max_len_infer":	null,
"num_encoder_layers":	2,	"time_major":	true,
"num_decoder_layers":	2,	"unit_type":	"lstm",
"beam_width":	10	"eopch":	30
"layer type":	lstm		

5.3.2 Common Gateway Interface

The common gateway interface(CGI) shows how the web client of the system interacts with the webserver to translate the source text to the target text. The interface contains four components as pictured in Figure 5.1, functioning as a one-component via hypertext transfer protocol(HTTP) protocol.

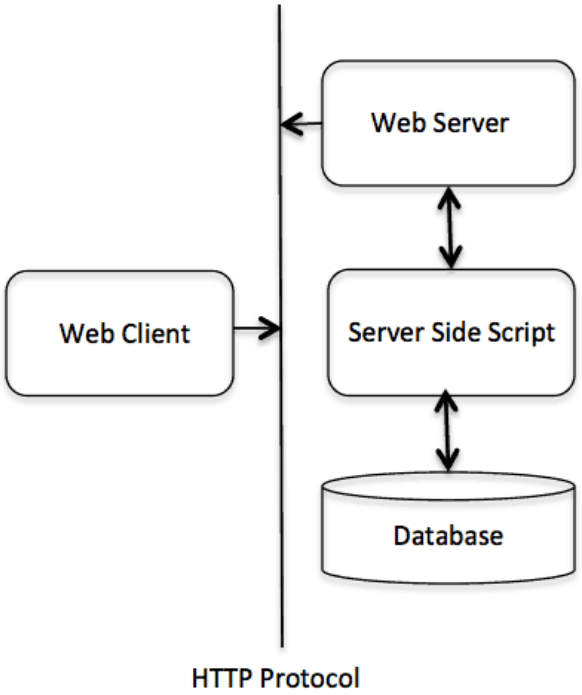

Figure 5.1 Common Gateway Interface

5.3.3 High-level Model Architecture

Our high-level model architecture is based on an N-tier language translation design (Figure 5.2). The architecture is classified into four tiers; client, presentation and interaction, analysis and retrival, and resource.

1. Clients:- offering access to the system. Normally, existing client, to connect with our architecture over a plug-in web interface. It enable the clients to use up the server and offering functionality for translation services.

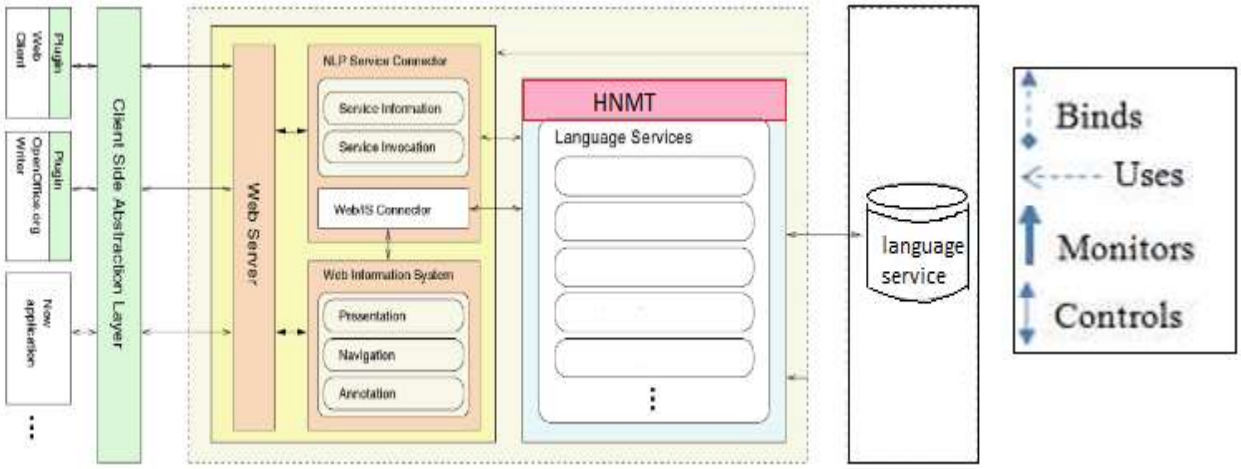

Figure 5.2 High-level architecture of the prototype

2. Presentation and Interaction:- This section includes a standard web server and a module categorized, NLP service connector in Figure 5.2. The module interact, and it handles the communiqué movement among the NLP structure(framework) and the web server. Moreover, it formulates language service results, through gathering result as of the NLP services and converting them interested in a layout appropriate for translation to the user.

3. Analysis and Retrieval:- This section directly retrieved(accessed) through NLP service connector. It includes the core HNMT models to give translation service to the client.

4. Resources:- include the data assets of the language pairs.

5.3.4 Graphical User Interface

We are going to show the prototyping interface in three steps; progress one, progress two and progress three.

Figure 5.3: Progress one

Figure 5.4: Progress two

Figure 5.5: Progress three

5.4 Coding implementation

Sample code 5.1: Importing packages and Tensorflow

```
import tensorflow as tf
import string
import re
from pickle import dump
from unicodedata import normalize
from numpy import array
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.spatial.distance import cdist
from tensorflow.python.keras.models import Sequential
from tensorflow.python.keras.layers import Dense, GRU, LSTMCell, Embedding
from tensorflow.python.keras.optimizers import Adam
from tensorflow.python.keras.preprocessing.text import Tokenizer
from tensorflow.python.keras.preprocessing.sequence import pad_sequences
%matplotlib inline
```

Sample code 5.2: Attention mechanism

```
class EncoderLSTM(nn.Module):
  def __init__(self, input_size, hidden_size, n_layers=1, drop_prob=0):
    super(EncoderLSTM, self).__init__()
    self.hidden_size = hidden_size
    self.n_layers = n_layers

    self.embedding = nn.Embedding(input_size, hidden_size)
    self.lstm = nn.LSTM(hidden_size, hidden_size, n_layers, dropout=drop_prob, batch_first=True
def forward(self, inputs, hidden):
    # Embed input words
    embedded = self.embedding(inputs)
    # Pass the embedded word vectors into LSTM and return all outputs
    output, hidden = self.lstm(embedded, hidden)
    return output, hidden

  def init_hidden(self, batch_size=1):
    return (torch.zeros(self.n_layers, batch_size, self.hidden_size, device=device),
            torch.zeros(self.n_layers, batch_size, self.hidden_size, device=device))
```

Sample code 5.3: Sparse Categorical Crossentropy loss function

```
from __future__ import absolute_import, division, print_function, unicode_literals
import tensorflow as tf
tf.keras.backend.clear_session()
from tensorflow import keras
from tensorflow.keras import layers

inputs = keras.Input(shape=(784,), name='digits')
x = layers.Dense(64, activation=' softmax ', name='dense_1')(inputs)
x = layers.Dense(64, activation=' softmax', name='dense_2')(x)
outputs = layers.Dense(10, activation=' softmax ', name='predictions')(x)

model = keras.Model(inputs=inputs, outputs=outputs)

model.compile(optimizer=keras.optimizers.Adam(),

              loss=keras.losses.SparseCategoricalCrossentropy(),

              metrics=[keras.metrics.SparseCategoricalAccuracy()])
```

Sample code 5.4: Language model

```
class LM_Model(Model):
    def __init__(self,
                 cost_layer = None,
                 sample_fn = None,
                 valid_fn = None,
                 noise_fn = None,
                 clean_before_noise_fn = False,
                 clean_noise_validation=True,
                 weight_noise_amount = 0,
                 indx_word="/data/lisa/data/PennTreebankCorpus/dictionaries.npz",
                 need_inputs_for_generating_noise=False,
                 indx_word_src=None,
                 character_level = False,
                 exclude_params_for_norm=None,
                 rng = None):
    if args.each:
            if args.pickle:
                safe_pickle(binarized_corpus, base_filename + '.pkl')
            if args.ngram and args.split:
                if args.split >= 1:
                    rows = int(args.split)
                else:
                    rows = int(ngram_count * args.split)
                logger.info("Saving training set (%d samples) and validation "
                            "set (%d samples)."
                            % (ngram_count - rows, rows))
                rows = numpy.random.choice(ngram_count, rows, replace=False)
                safe_hdf(ngrams[total_ngram_count + rows],
                         base_filename + '_valid')
                safe_hdf(
                    ngrams[total_ngram_count + numpy.setdiff1d(
                        numpy.arange(ngram_count),
                        rows, True
                    )], base_filename + '_train'
                )
            elif args.ngram:
                logger.info("Saving n-grams to %s." % (base_filename + '.hdf'))
                safe_hdf(ngrams, base_filename)
        binarized_corpora += binarized_corpus
        total_ngram_count += ngram_count
        input_file.seek(0)
```

Sample code 5.5: CPU configuration with LSTM and attention mechanism

```
cells = []
for i in range(num_layers):
  cells.append(tf.contrib.rnn.DeviceWrapper(
      tf.contrib.rnn.LSTMCell(num_units),
      "/cpu:%d" % (num_layers % num_cpus)))
attention_cell = cells.pop(0)
attention_cell = tf.contrib.seq2seq.AttentionWrapper(
    attention_cell,
    attention_mechanism,
    attention_layer_size=None,
    output_attention=False,)
cell = GNMTAttentionMultiCell(attention_cell, cells)
```

Sample code 5.6: word reward feature

```
class WRF_Model(Model):
def word_rward_feature():
    if args.ngram:
        assert numpy.iinfo(numpy.uint16).max > len(vocab)
        ngrams = numpy.empty((sum(combined_counter.values()) +
                              sum(sentence_counts), args.ngram),
                             dtype='uint16')
    binarized_corpora = []
    total_ngram_count = 0
    for input_file, base_filename, sentence_count in \
            zip(args.input, base_filenames, sentence_counts):
        input_filename = os.path.basename(input_file.name)
        logger.info("Binarizing %s." % (input_filename))
        binarized_corpus = []
        ngram_count = 0
        for sentence_count, sentence in enumerate(input_file):
            if args.lowercase:
                sentence = sentence.lower()
            if args.char:
                words = list(sentence.strip().decode('utf-8'))
            else:
                words = sentence.strip().split(' ')
            binarized_sentence = [vocab.get(word, 1) for word in words]
            binarized_corpus.append(binarized_sentence)
            if args.ngram:
                padded_sentence = numpy.asarray(
                    [0] * (args.ngram - 1) + binarized_sentence + [0]
                )
                ngrams[total_ngram_count + ngram_count:
                       total_ngram_count + ngram_count + len(words) + 1] =  \
                    as_strided(
                        padded_sentence,
                        shape=(len(words) + 1, args.ngram),
                        strides=(padded_sentence.itemsize,
                                 padded_sentence.itemsize)
                    )
            ngram_count += len(words) + 1
            return WRF_Model(Model)
```

Sample code 5.7: Word translation probabilities

```
class WTP_Model(Model):
def create_word_translation_probablity():
    counters = []
    sentence_counts = []
    global_counter = Counter()

    for input_file, base_filename in zip(args.input, base_filenames):
        count_filename = base_filename + '.count.pkl'
        input_filename = os.path.basename(input_file.name)
        if os.path.isfile(count_filename) and not args.overwrite:
            logger.info("Loading word counts for %s from %s"
                        % (input_filename, count_filename))
            with open(count_filename, 'rb') as f:
                counter = cPickle.load(f)
            sentence_count = sum([1 for line in input_file])
        else:
            logger.info("Counting words in %s" % input_filename)
            counter = Counter()
            sentence_count = 0
            for line in input_file:
                if args.lowercase:
                    line = line.lower()
                words = None
                if args.char:
                    words = list(line.strip().decode('utf-8'))
                else:
                    words = line.strip().split(' ')
                counter.update(words)
                global_counter.update(words)
                sentence_count += 1
        counters.append(counter)
        sentence_counts.append(sentence_count)
        logger.info("%d unique words in %d sentences with a total of %d words."
                    % (len(counter), sentence_count, sum(counter.values())))
        if args.each and args.count:
            safe_pickle(counter, count_filename)
        input_file.seek(0)
        load WTP_Model(Model)
```

5.5 Experimenting

The experiments carried out on English-to-Afaan Oromo small-sized parallel dataset CPU configured a single machine. The small-sized parallel dataset collected from the web, containing about more than 13000 Afaan Oromo, and English sentences. As pictured in Figure 5.6 in case of model training, the experiment carried out on 30 epochs and 10 batch size for eight weeks CPU configured TensorFlow and on the top of keras layers.

The number of training example(epochs) for the proposed hybrid neural machine translation is implemented in 30 epochs. We used hold out cross validation approach for data splitting (Figure 5.6) from the prepared small sized parallel dataset for model traning and testing.

```
None
Train on 9000 samples, validate on 1000 samples
Epoch 1/30

Epoch 00001: val_loss improved from inf to 3.46747, saving model to model.h5
9000/9000 - 235s - loss: 4.2046 - val_loss: 3.4675
Epoch 2/30

Epoch 00002: val_loss improved from 3.46747 to 3.32327, saving model to model.h5
9000/9000 - 71s - loss: 3.3008 - val_loss: 3.3233
Epoch 3/30

Epoch 00003: val_loss improved from 3.32327 to 3.24655, saving model to model.h5
9000/9000 - 77s - loss: 3.1650 - val_loss: 3.2466
Epoch 4/30

Epoch 00004: val_loss improved from 3.24655 to 3.13213, saving model to model.h5
9000/9000 - 75s - loss: 3.0460 - val_loss: 3.1321
Epoch 5/30
Epoch 00023: val_loss improved from 1.92686 to 1.89138, saving model to model.h5
9000/9000 - 75s - loss: 0.7623 - val_loss: 1.8914
Epoch 24/30

Epoch 00024: val_loss improved from 1.89138 to 1.89051, saving model to model.h5
9000/9000 - 74s - loss: 0.6930 - val_loss: 1.8905
Epoch 25/30

Epoch 00025: val_loss improved from 1.89051 to 1.87639, saving model to model.h5
9000/9000 - 73s - loss: 0.6315 - val_loss: 1.8764
Epoch 26/30

Epoch 00026: val_loss improved from 1.87639 to 1.86436, saving model to model.h5
9000/9000 - 73s - loss: 0.5746 - val_loss: 1.8644
Epoch 27/30

Epoch 00027: val_loss improved from 1.86436 to 1.85138, saving model to model.h5
9000/9000 - 80s - loss: 0.5210 - val_loss: 1.8514
Epoch 28/30

Epoch 00028: val_loss did not improve from 1.85138
9000/9000 - 86s - loss: 0.4748 - val_loss: 1.8525
Epoch 29/30

Epoch 00029: val_loss improved from 1.85138 to 1.84120, saving model to model.h5
9000/9000 - 71s - loss: 0.4295 - val_loss: 1.8412
Epoch 30/30

Epoch 00030: val_loss improved from 1.84120 to 1.83889, saving model to model.h5
9000/9000 - 77s - loss: 0.3962 - val_loss: 1.8389
```

Figure 5.6: Traning Epochs

5.5.1 System Setup

The parallel training data for English-Afaan Oromoo contains a dataset of restricted vocabulary size get ready using more than 13000 sentences for Afaan Oromo (435k) and English (371k) words individually. We use hold out machine learning approach(chapter 3) for datasets classification and model training and evaluation. We describe experiment findings with BLEU-4(4-gram BLUE score) for evaluation. To measure up the proposed model performance with the previous work, we used a similar training dataset, containing more than 13000 sentence pairs with

371k English words and 435k Afaan Oromo words. The proposed model compared and contrasted with translation systems:

- RNNSearch and
- PBSMT

5.5.2 Model training and Testing

In this section, we describe how model training and model testing can be held and how to train the proposed HNMT model that automatically translate the given source text to the target text. Training our proposed model, by learning the values of *wij*(weights) and *bj*(biases) parameters is the important part of deep learning. We can observed that, this learning procedure in a NN as an iterative process of "return and going" by the layers of neurons. The return is a back propagation and going is forward propagation of the history(information). We bound the vocabulary size to 20K most common words for the source, and target languages. All the remaining terms tokenized by a unique symbol unknown"UNK". We trained the language model through KN-discount on the target part of the parallel corpus. The language model and word translation table then implemented as features incorporated with the OpenNMS-tr. Firstly, by tunning up different parameter, learning rate, batch size and network size into the model. Secondly, by appling testing datafeeding with length of the source

CHAPTER SIX

6. Evaluation, Result, and Discussions

In the evaluation, result and discussion part, the methodology and the internal structures, algorithms, and prototype is justified and done according to the MT research strategy method. So, the proposed solution and prototype issued OOV problem leads to unknown word output, lack of translation problem for long sentences, and lack of decoding mechanism problem and the target resources for MT, and the reason for HNMT is in this study. HNMT deep learning algorithms, model representation, results, and findings which answers the questions, how can be handle the OOV translation problems lead to unknown word output using a hybrid translation approach? , how can be models incorporated with the parallel dataset to enhance the lack of translation for long sentences to improve translation quality for fluent translation? and how can be alleviated lack of decoding translation problems occurred in NMT using a hybrid approach? Finally, assurance way, handling of OOV, decoding mechanism and translating long sentence problems are also justified in the solutions. For assurance and validation, testing; and model checking are used to evaluate and discuss the results of the developed prototype. For results purposes, as described in Table 5.1, different state modules are collected during the configuring of the system at execution time.

6.1 BLEU-score Evaluation result

We describe the evaluation results of the HNMT model, which shows the proposed model evaluated with the existing systems called SMT, RNN based on the BLEU score(4-gram) evaluation metric, with in the same sized dataset, batch size, beam size, and training epochs. The evaluation result of proposed model shows the performance(potential) of our method and the rest two model by splitting the dataset into two sets(training and testing) parts. The BLEU score evaluation metric evaluates the proposed model and classifies the dataset for model training and model testing phase. In the model training phase, the parameter scored using a 4-gram BLEU score (Figure 6.1) and similarly in the model testing phase(Figure 6.2). The models trained in the hold-out cross validation approach for the subsequent optimized phase estimation, for the parallelism

of translation quality. We used holdout cross-validation training to adequately regulate sentence BLEU scores after the typical highest - probability training.

```
HNMT model training
src=[Alaabaan itiyoophiyaa gara irraa magariisa gidduun keelloo jalaan diimaa ta'ee ni qabaata],
 target=[The Ethiopian flag shall consist of green at the top, yellow in the middle and red at the bottom],
 predicted=[The Ethiopian flag will comprise of green at the top, yellow in the central and red at the bottom.]
src=[Mana Murtii Dhimmoota Magaalaa jechuun qaama dhimmoota magaalaa ilaallatanirratti aangoo abbaa seerummaa qabaatee dhaa
bbate jechuu dha.],
 target=[City Court means a court established to decide on city-related cases.],
 predicted=[City Court means a court which established to decide on belongings city- interrelated case.]
src=[Gaa'elli kan raawwatamu fedhii bilisaa fi guutuu namoota walfuuchuuf fedhanii qofaani.],
 target=[Marriage shall be entered into only with the free and full consent of the intending spouses.],
 predicted=[Marriage shall be entered into only with the free and full consent of the intending spouses.]
src=[Namni kamiyyuu mirgi lammummaa isaa garmalee hin mulqamu akkasumas mirga lamummaa isaa geeddaruus hin dhorkamu.],
 target=[No one shall be arbitrarily deprived of his nationality nor denied the right to change his nationality.],
 predicted=[No one shall be arbitrarily deprived of his nationality nor denied the right to change his nationality.]
src=[Namni kamiyyuu miidhaa duraa baqachuu fi biyyoota biroo keessatti irkataa tahee jiraachuuf gaafachuuf mirga ni qaba.],
 target=[Everyone has the right to seek and to enjoy in other countries asylum from persecution.],
 predicted=[Everybody has the right to seek and to enjoy in other countries asylum from discrimination.]
src=[Abbaa keenya Waaqa biraa ayyaannii fi nageenni isinii haa ta'uu.],
 target=[Grace be unto you, and peace, from God our Father and the Lord Jesus Christ.],
 predicted=[Grace be unto you, and peace, from God our Father and the Lord Jesus Christ.]
src=[Namni kamiyyuu naannoo biyya isaa keessa bilisaan sossohuu fi jiraachuuf mirga ni qaba.],
 target=[Everyone has the right to freedom of movement and residence within the borders of each State.],
 predicted=[Everyone has the right to freedom of movement and residence within the borders of each State.]
src=[Ani seera kiyya yaada isaanii keessa nan kaa'a.],
 target=[I will put my laws into their mind.],
predicted=[I will keep my laws into their opinion.]
src=[Ani balleessaa isaanii nan dhiisaaf.],
target=[I will be merciful to their unrighteousness.],
 predicted=[I will be merciful to their unrighteousness.]
src=[Itti waamamni af-yaa'ii Mana Marii Magaalaa mana Marii magaalichaaf ta'a.],
target=[The speaker of the City Council shall be accountable to the City council.],
predicted=[The speaker of the City Council shall be accountable to the City council.]

BLEU-1: 0.885471
BLEU-2: 0.857942
BLEU-3: 0.839190
BLEU-4: 0.837384
```

Figure 6.1: HNMT model training evaluation result

```
test
src=[nutti ni goona], target=[well make it], predicted=[well do it]
src=[kitaaba kee dubbisi], target=[read your book], predicted=[use the book]
src=[dhuga miti], target=[it isnt true], predicted=[its not gold]
src=[dhabatanii jiru], target=[they stopped], predicted=[they kissed]
src=[ani baay'ee hargane], target=[i burp a lot], predicted=[i am a lot]
src=[kun dhugaa dha], target=[its all right], predicted=[its all]
src=[ani adama jira], target=[im in adama], predicted=[im am]
src=[caalaan taphataa jira], target=[chala is playing], predicted=[chala is wise]
src=[maal kenna], target=[what gives], predicted=[anything new]
src=[maallaqa dhoksi], target=[hide the money], predicted=[hide the money]
BLEU-1: 0.571282
BLEU-2: 0.569920
BLEU-3: 0.539285
BLEU-4: 0.531282
```

Figure 6.2: HNMT model testing evaluation result

The evaluation of a model on the finest En-Orm models observed in Table 6.1, which indicates that fine-tuning the models with different hyperparameters improve BLEU scores. On En-Orm, model fine-tuning gets better BLEU score close point.

Table 6.1: BLEU score evaluation result in different n-gram training and testing

n-gram	***Training***	***Testing***
BLEU-1	0.885471	0.571282
BLEU-2	0.857942	0.569920
BLEU-3	0.839190	0.539285
BLEU-4	0.837384	0.531282

6.2 Result

In this section, we will table(Table 6.1, Table 6.2, Table 6.3, and

Table **6.4**), figure and explain some experimental results on the bases of the training set and test sets through adjusting training data(%), batch size, activation function, noise and training epochs. We observed that the experimental result of the proposed method in a sufficiently great modifies the translation quality of the typical NMT model. Additionally, our HNMT system outperforms the NMT and PBSMT system on a similar small training dataset. A further novel finding is that setting and tunning the wright learning rate, training data size, testing data size, and noise to reduce the training and testing loss with the specified epochs and batch size is the challenging parts in

ANN sequential data. In case of model training with the initial learning rate setup, the learning rate set up to 3, and test loss is typically higher than training loss(Table 6.2,

Table **6.4**). The expermental result confirmed that, when the epochs are between 10-30 the training loss and testing loss is the same. We showed that by degrading the learning rate parameter from 3 to 0.01, it leads to good results through parameter variation(Table 6.3). The learning rate variation confirmed that the findings on test loss decline to a result much nearer to training loss.

Table 6.2: Traning and testing loss learning rate to 3

Neural network	Epochs	Training loss	Testing loss	Learning rate
Four neural network layer	0-2	0.213	0.246	3
	3-5	0.253	0.251	3
	6-7	0.252	0.251	3
	8	0.251	0.251	3
	9	0.250	0.251	3
	10-30	0.249	0.251	3

Our experimental results demonstrated that the result shown in tables(Table 6.2, Table 6.3 and Table 6.4) tuning the model with in the same training and testing dataset model(70,30), noise(80%) and batch size(1-10) we obtain good results with the learning of 0.01(Table 6.3).

Table 6.3: Traning and testing loss learning rate to 0.1

Neural network	Epochs	Training loss	Testing loss	Learning rate
Four neural network layer	0	1.527	1.466	0.1
	2	0.219	0.222	0.1
	3	0.215	0.219	0.1
	4	0.219	0.221	0.1
	5	0.215	0.219	0.1
	6	0.215	0.219	0.1

On the English-Afaan Oromoo dataset, in case of model training and testing, the resulting harm the model performance. Also, we perceive about 1.312(Table 6.3) BLEU points decrement on the training set and 1.247 points decrement on the test set. Another promising outcome was that in most execution, increasing the batch size with integrated components does not significantly affect both in the training loss or test loss. Also, in a slight percentage of execution, rising batch size to greater or equal to twenty make happen test loss to drop to some extent below training loss

Table 6.4: Traning and testing loss learning rate to 0.001

Neural network	Epoch	Training loss	Testing loss	Learning rate
Four neural network layer	0	0.168	0.194	0.001
	1	0.161	0.189	0.001
	2	0.161	0.189	0.001
	3	0.161	0.189	0.001
	4-14	0.161	0.189	0.001
	15	0.186	0.251	0.001

We observed that our proposed translation model enabled HNMT model significantly improved the OOV problem compared with the rest of the two(or more) models (Figure 6.3).

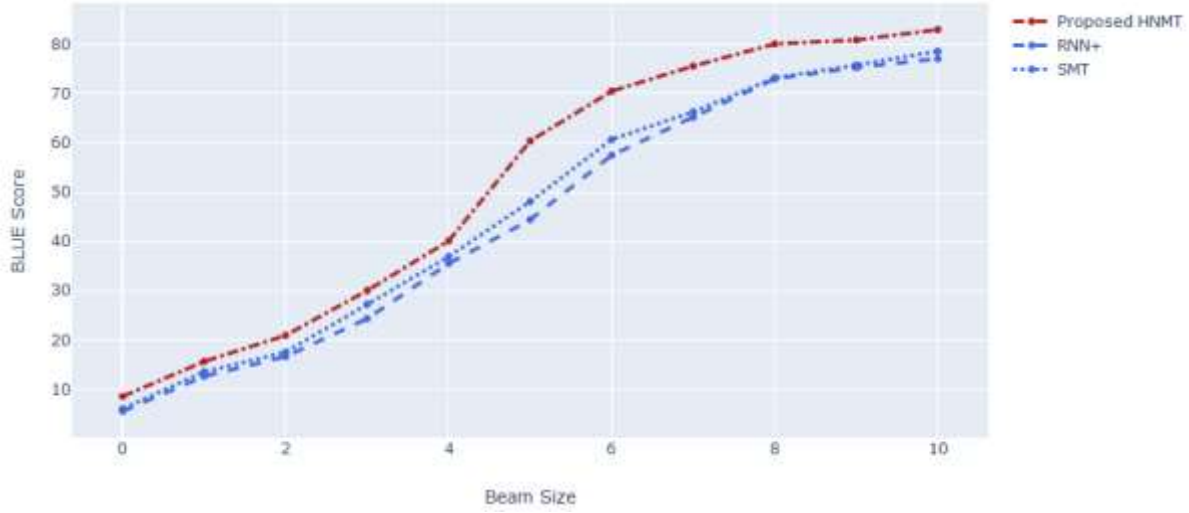

Figure 6.3: HNMT model that reduces OOV problem

The statistical percentage of the OOV word for the SMT, RNNSearch, and HNMT(Table 6.5) based on a variation of batch size indicate that our proposed model OOV percentage is less than from the rest of two models. The incorporating translation model, properly reduces the OOV word percentage in the training and test sets.

Table 6.5: Statistics of the percentages of the OOV words

Model	Training	Testing
RNNSearch	3.2 percent	3.4 percent
SMT	1.2 percent	1.4 percent
HNMT	0.6 percent	0.7 percent

Our proposed HNMT model reduces the OOV problem that leads to UNK word output with beam size 10 (Figure 6.4). The experimental result is showing that, the translation model scored a good result with the best word and phrase aliened. The result shows that our model increases the translation quality of the sentence compared with SMT, RNNSearch, and HNMT based on the variation of beam size indicate that our proposed model scored a good result from the rest of the two(or more) models. The incorporating translation model, properly achieved the reduction of OOV word that leads to unknown word output in both sets.

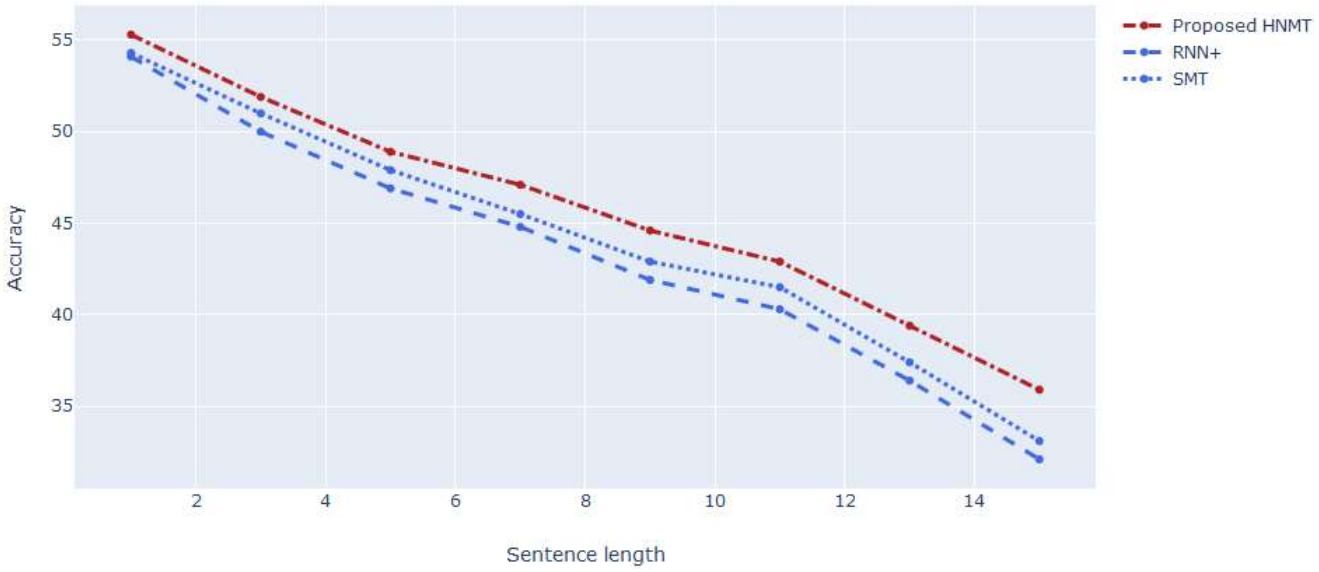

Figure 6.4: Enhancing lack of translation problem for long sentence by adding

To enhancing the translation problem for a long sentence is reduced by incorporating our proposed HNMT model with the word reward feature (Figure 6.4.). The included feature appropriately increased the translation quality by regulating insufficient translation problems for extended decoder reading. Also, the word reward feature model regulates the length of translation by including bi-directional word translation probability by making the decoder favor for a long sentence. The result showing that the word reward feature scored a good result by predicting lexical translation probabilities among target contents. The finding of the model increases the translation quality of the source(sentence)compared with SMT, RNNSearch, and HNMT based on the variation of beam size indicate that our proposed model scored a good result from the rest of the two models (

Table 6.6, Table 6.7). The incorporating translation model, properly increased the reduction of OOV word that leads to unknown word output in both sets.

Table 6.6: BLEU scores

System	***BLEU-4 score***
SMT	0.693453
RNN	0.687024
HNMT	0.837384

Table 6.7: BLEU scores the testing dataset

System	***BLEU-4 score***
SMT	0.29821
RNN	0.29325
HNMT	0.531282

On the contrary, upgrading the training data percentage from 70% to 80% fundamentally degrading the quantity of data segment points in the trained dataset. With so little data, high batch size and high learning rate make happen the training model to jump repeatedly over the minimum point. The training and testing accuracy of the model directly affected by the amount of data splitting, batch size, beam size, epochs and learning rates (Figure 6.5). Further analyses and discussions will be explained in section 6.3.

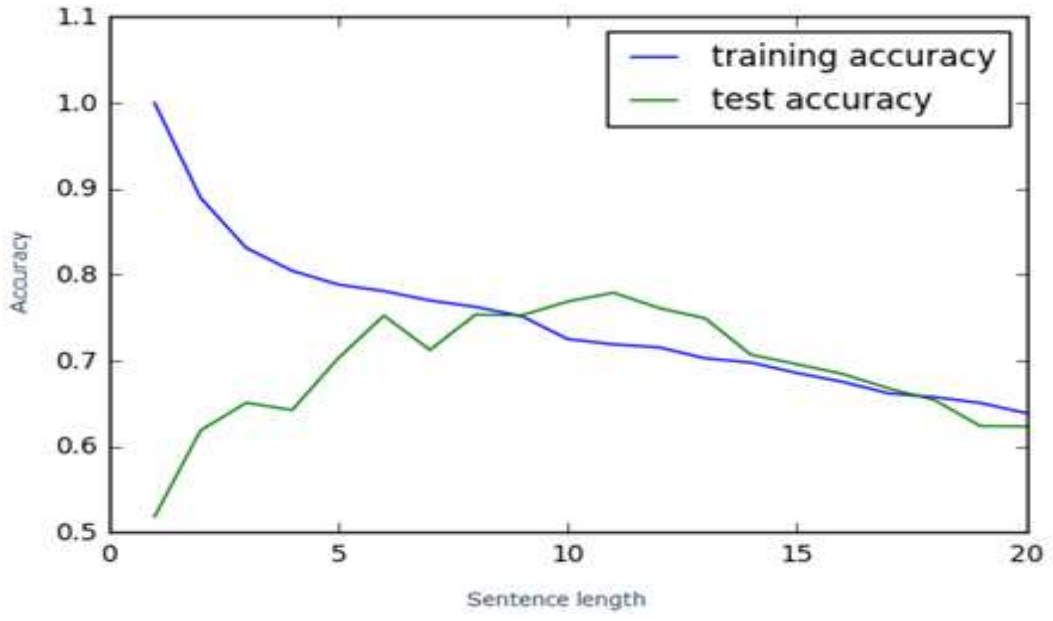

Figure 6.5: HNMT training and testing accuracy

6.3 Discussion

The result from this study suggests that the HNMT model alleviates the most common translation problem shown in the NMTSearch system. In order to deeply analyze the functioning of the proposed solution, we contrast the result of the systems.

Late as taking the first sentence in Our HNMT correctly translates Afaan Oromoo words in bold as an example, the rest system omits the translation of transmission. In fact, the target words are in the vocabulary but not selected by the RNN model. By integrating a translation table, our method produces the correct translation for the source words omitted by the RNNsearch. This can be characterized to the fact that the translation table consists of word pairs with translation probabilities estimated from the word-aligned training dataset, providing another way to measure the relationship between source and target words. In this example, the translation table contains the word pairs with high probabilities. To further improve the quality of lexical translation, we employ a conventional n-gram (n=4) language model to improve the local fluency. For example, there is another entry “caalaan, transfer” for the Afaan oromoo word “caalaan” in the translation table. The n-gram language model could help the decoder predict the correct translation because plm(transfer|series of high speed) smaller than plm(transmission|series of high speed).

Table 6.8: Translation examples

In this table(Table 6.8) Afaan Oromoo words in bold are correctly translated by our HNMT compared with the previous models used for comparison on this research.

Input:	Alaabaan itiyoophiyaa gara irraa magariisa gidduun keelloo jalaan diimaa ta’ee ni qabaata
Target:	The Ethiopian flag shall consist of green at the top, yellow in the middle, and red at the bottom.
Prediction:	The Ethiopian flag **will comprise** of green at the top, yellow in the **central** and red at the bottom.
Input:	Mana Murtii Dhimmoota Magaalaa jechuun qaama aangoo dhimmoota magaalaa ilaallatanirratti abbaa seerummaa qabaatee dhaabbate jechuu dha.
Target:	City Court means a to decide on city-related cases court established.

Prediction:	City Court means a court **which** established to decide on **belongings** city-interrelated case.
Input:	Gaa'elli kan raawwatamu fedhii bilisaa fi guutuu namoota walfuuchuuf fedhanii qofaani.
Target:	Marriage shall be entered into full consent of the intending spouses and only with the free.
Prediction:	Marriage shall be entered into full consent of the intending spouses and only with the free.
Input:	Namni kamiyyuu mirgi lammummaa isaa garmalee hin mulqamu akkasumas mirga lamummaa isaa geeddaruus hin dhorkamu.
Target:	No one shall arbitrarily deprived of his nationality in the case denied the right to change his nationality.
Prediction:	No one shall arbitrarily deprived of his nationality in the case denied the right to change his nationality.
Input:	Namni kamiyyuu miidhaa duraa baqachuu fi biyyoota biroo keessatti irkataa tahee jiraachuuf gaafachuuf mirga ni qaba.
Target:	Everyone has the right to seek, to enjoy in other countries asylum from persecution.
Prediction:	**Everybody** has the right to seek and to enjoy in other countries asylum from **discrimination**.
Input:	Abbaa keenya Waaqa biraa ayyaannii fi nageenni isinii haa ta'uu.
Target:	Grace be unto you, and peace from God our Father and the Lord Jesus Christ.
Prediction:	Grace be unto you, and peace, from God our Father and the Lord Jesus Christ.
Input:	Namni kamiyyuu naannoo biyya isaa keessa bilisaan sossohuu fi jiraachuuf mirga ni qaba.
Target:	Everyone has the right to freedom of movement and residence within the borders of each State.

Prediction:	Everyone has the right to freedom of movement and residence within the borders of each State.
Input:	Ani seera kiyya yaada isaanii keessa nan kaa'a.
Target:	I put my laws into their minds.
Prediction:	I will **keep** my laws in their **opinion**.
Input:	Ani balleessaa isaanii nan dhiisaaf.
Target:	I will be merciful to their unrighteousness.
Prediction:	I will be merciful to their unrighteousness.
Input:	Itti waamamni Mana Marii Magaalaa af-yaa'ii mana Marii magaalichaaf ta'a.
Target:	The speaker of the City Council shall be responsible to the City council.
Prediction:	The speaker of the City Council shall be responsible to the City council.

Translating the OOV words(Table 6.5) shows the statistics of the OOV words for PBSMT, RNNSearch and our systems on test sets. It is observed that all the systems confront the OOV problem because the source words do not occur in the training corpus or the word pairs are not learned due to the word alignment error. However, the difficulty is more severe for the RNNSearch system since it limits the vocabulary size to reduce the model complexity. The OOV words harm the translation quality. As shown in the first example, the source word(shall consist) is not translated. After integrating the translation table within the log-linear model, this word was correctly translated. As demonstrated above, our method reduces the OOV words percent for the NMT system(Table 6.5). Moreover, the number of OOV words in our system is half of that in the PBSMT system. As we know, PBSMT system extracts word/phrase translations from word-aligned bilingual corpus. However, constrained by the inaccurate word alignment, not all words in the bilingual corpus are covered in the phrase table, causing OOV problems in the PBSMT system.

Ideally, the RNN encoder-decoder is capable of translating all the words as long as they are encoded in the vocabulary. As the vocabulary used in the RNN encoder-decoder is limited for practical reasons, adding word translation table into RNN encoder-decoder combines the strength of both RNN and PBSMT, leading to a further reduction of OOV ratio. Table 6.8 shows the effect of translating OOV words with the translation table. Our Method shares the same settings with

section Experiment. In these settings, the translation table is not only used to score word pairs, but also used to translate OOV words. The OOV means that the translation table is only used to score word pairs in the vocabulary of the RNN encoder-decoder, but not generate new translation candidates for the OOV words. It is observed that by translating OOV words, we obtained an absolute improvement of BLEU scores.

The experimental finding from the result(Figure 6.5, Figure 6.3, Figure 6.4, and

Table 6.7) specifically, the following result conclusions formulated. From these results, it is clear that by combining the word reward features and word translation table, the proposed method achieved practical improvements over the two methods (SMT, RNNsearch). The results demonstrate three main reasons for the resulting improvement. Firstly, the translation probabilities benefit the NMT system, to perform effective lexical translation. Secondly, the translation table alleviated OOV that cause unknown word results by recovering translations of unknown words result. Thirdly, the word reward component makes the decoder favors long translation. The average lengths of the outputs on the test set of our system and other 0.53, 0.50, and 0.51, respectively. This indicates that our method alleviates the inadequate translation problem.

Our method allows the NMT system, to incorporate additional language models. We includes language model(4-gram) trained, to the focus(target) sideways of parallel dataset to the OSNMT-tr system. It is observed that our method obtained further improvements on the test set, as the n-gram, language model captures local target contextual information and improve the fluency. Compared with the SMT, NMT system, our system (HNMT) achieves an absolute improvement of 2.4(0.24) points in BLEU-4 score, which is statistically significant at $p = 0.01$ learning rate.

CHAPTER SEVEN

7. Conclusion, Recommendation and Future work

7.1 Conclusion

In this paper, we addressed our results on HNMT using the process, only the phrase translation pair characteristics are determined. The observations from the various experiments prompted us to combine the SMT approaches with the NMT approaches. The score of the translations obtained with this combination was increased and this also results in improved translation quality according to our experiments. In this experimental study, we proposed a new model to avoid the OOV handling problem, inadequate translation problem and long sentence decoding problem.

The experiment was carried out with a small size of the dataset. We improved the issues by incorporating SMT components under the log-linear model, which makes the NMT approach be easily extended. The translation table was trained on word-aligned parallel dataset via the standard phrase-based SMT method, and the language model was trained on parallel dataset target sentences. The proposed method alleviates the major limitations of the current NMT system and we achieved our research objectives. The translation table recovers the omitted translations of source words and the OOV words, and the language model increases local fluency by making full use of parallel corpus. Experiments on English-to-Afaan Oromoo translation tasks show that our system achieves significant improvements over the baseline on a small amount of the training corpus collected from the web. The proposed method achieved better performance compared with the previous (e.g., RNN-search, and SMT) models. The research significantly showed that by combining the two model the translation result can be improved for the state-of-art.

This subsection summarizes the findings and contributions made on this study. Firstly, we propose a new approach for Afaan Oromoo-English language translation with its challenges by incorporating all NMT features. Secondly, the study introduce the Afaan Oromoo language to ANN deep machine learning techniques. Thirdly, we contribute dataset set for users, researchers and developers of NLP tasks. Finally, this research scored a new result using hybrid MT approach and alleviated the translation problems occurred NMT system. This subsection summarizes the findings and contributions made on this study.

- Firstly, we propose a new approach for Afaan Oromoo language translation with its challenges by incorporating all NMT features.
- Secondly, the study introduce the Afaan Oromoo language to ANN deep machine learning techniques.
- Thirdly, we contribute dataset set for users, researchers and developers of NLP tasks.
- Finally, this research scored a new result using hybrid MT approach and alleviated the translation problems occurred NMT system.

7.2 Recommendation

HNMT systems have the capability of translating long sentences itself based on real-time learning to the changing environments because they have better performance and mitigate runtime problems. There are several rooms this work can strengthen. First, a model size more significant than the amount of a combined word-based SMT, and NMT models will result from the proposed method. To minimize the scale of the model, we must explore system reduction strategies. Second, the proposed model lacks the SMT word recommendations function values that are very useful in NMT. Therefore, to improve the scoring accuracy, we want to attach information to the SMT word recommendations. Also, we want to introduce more complicated models to use SMT phrase recommendations, where NMT can benefit more from multi-word phrase recommendations than word-level ones as phrase-based SMT works better.

Finally, we would like to recommend those who want use and reference this research study, its better that if you consider the following points.

- Model training and testing using more than one dataset.
- Model training and testing using largescale dataset or more corpus size dataset.
- Train, validate and test the model using GPU enabled server.
- Train the model by using different parameters.
- researchers to develop systems using a GPU

7.3 Future work

As a new approach, NMT still has an option for improvement. The current RNN encoder-decoder is a word-based translation system. To the future, we formulate to improve the NMT model, with word-piece. Why because which are good at capturing local word reordering, idiom translation, etc. Also, we try to incorporate more than three models by structuring more than four NN neurons. This blend of strategies allows us to continue working in the future. The experimentation is carried out by using more than two standard language assets(standard dataset). We try to build the network using more four layers and model training and testing using more than one dataset,model testing using largescale dataset,train, validate and test the model using GPU enabled server, train the model by using different parameters

REFERENCES

[1] N. B. and N. Lacascio, *Fundamentals of Deep Leaning,* First Edit. Gravenstein Highway North, Sebastopol: O'Reilly Media, 2017.

[2] J. Patterson and A. Gibson, *Deep Learning A practitioner's approach*, vol. 14, no. 14. Gravenstein Highway North, Sebastopol: O'Reilly Media, Inc., 2008.

[3] V. Armando, *Introduction to Deep Learning Business Applications for Developers*, 2nd editio. Bernardete Ribeiro Coimbra, Portugal: Apress, 2018.

[4] Kantanmt, "MACHINE TRANSLATION," *kantanmt.com*, 2019. [Online]. Available: https://kantanmt.com/documents/Machine_Translation.pdf.

[5] K. Cho *et al.*, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," *arXiv cs.Cl*, vol. 3, 2014.

[6] D. Koehn, P.;Och, F.J.;andMarcu, "Statisticalphrasebased translation," *Proc. HLT-NAACL*, vol. 4, pp. 127–133, 2003.

[7] Z. Yang, Z. Hu, Y. Deng, C. Dyer, and A. Smola, "Neural Machine Translation with Recurrent Attention Modeling," vol. 2, no. 2014, pp. 383–387, 2017.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," pp. 1–15, 2014.

[9] S. S, "Statistical Vs Rule Based Machine Translation; A Case Study on Indian Language Perspective," *ACM*, vol. 4, p. 10, 2017.

[10] M. Carpuat and D. Wu, "Improving Statistical Machine Translation using Word Sense Disambiguation," *Proc. EMNLP-CoNLL*, no. June, pp. 61–72, 2007.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," pp. 1–9, 2014.

[12] M. Pinnis, "Towards hybrid neural machine translation for English-Latvian," *Front. Artif. Intell. Appl.*, vol. 289, pp. 84–91, 2016.

[13] G. N. and S. N. P. Arthur, "Incorporating Discrete Translation Lexicons into Neural Machine Translation," *Proc. EMNLP, Novemb. 1-5, Texas, USA*, vol. 6, pp. 1557–1567, 2016.

[14] T. D. and R. S. M. Junczys-Dowmunt, "The AMU-UEDIN Submission to the WMT16 News Translation Task:Attention-based NMT Models as Feature Functions in Phrase-

based SMT," *Proc. First Conf. Mach. Transl. WMT 2016, Coloca. with ACL 2016*, vol. 4, pp. 319–325, 2016.

[15] L. Dahlmann, E. Matusov, P. Petrushkov, and S. Khadivi, "Neural Machine Translation Leveraging Phrase-based Models in a Hybrid Search," no. Lm, pp. 1411–1420, 2017.

[16] J. Du and A. Way, "Neural Pre-Translation for Hybrid Machine Translation," vol. 1, pp. 18–22, 2017.

[17] M. Junczys-Dowmunt, T. Dwojak, and R. Sennrich, "The AMU-UEDIN Submission to the WMT16 News Translation Task: Attention-based NMT Models as Feature Functions in Phrase-based SMT," 2016.

[18] C. D. V. Hoang, R. Haffari, and T. Cohn, "Improving Neural Translation Models with Linguistic Factors," *Proc. Australas. Lang. Technol. Assoc. Work. 2016*, pp. 7–14, 2016.

[19] G. F. S. and C. D. Fennig, "Ethnologue. Languages of Africa and Europe ed," vol. 72, 2018.

[20] N. Kalchbrenner and P. Blunsom, "Recurrent Continuous Translation Models," *Proc. ACL Conf. Empir. Methods Nat. Lang. Process.*, no. October, pp. 1700–1709, 2013.

[21] Y. W. et Al, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," *arXiv Cs Cl*, vol. 2, p. 23.

[22] P. Arthur, G. Neubig, and S. Nakamura, "Incorporating Discrete Translation Lexicons into Neural Machine Translation," *arXiv Cs Cl*, vol. 2, 2016.

[23] Z. Long, R. Kimura, T. Utsuro, T. Mitsuhashi, and M. Yamamoto, "Patent NMT integrated with Large Vocabulary Phrase Translation by SMT at WAT 2017," *Wat 2017*, no. 2016, pp. 110–118, 2017.

[24] H. Karlbom, "Hybrid Machine Translation: Choosing the best translation with Support Vector Machines," vol. 3, no. September, p. 12, 2016.

[25] F. T. Y. Vural, "Zero-Resource Translation with Multi-Lingual Neural Machine Translation," *arXiv Cs Cl*, vol. 1, 2016.

[26] S. Shen *et al.*, "Minimum Risk Training for Neural Machine Translation," *arxiv, cornell Univ.*, vol. 2, 2015.

[27] T. Xu *et al.*, "This paper is included in the Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16). Open access to the Proceedings of the 12th USENIX Symposium on Operating Systems Design and

Implementation is sponsored by US," *TensorFlow A Syst. Large-Scale Mach. Learn.*, p. 619, 2016.

[28] R. Zadeh and B. Ramsundar, *Tensorflow for Deep Learning*, First Edit. Gravenstein Highway North, Sebastopol: O'Reilly Media, 2018.

[29] J. Unpingco, *Python for Probability , Statistics , and Machine Learning*. San Diego, CA USA Additional: Springer, 2016.

[30] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "B LEU : a Method for Automatic Evaluation of Machine Translation," no. July, pp. 311–318, 2002.

[31] B. Dorr, M. Snover, and N. Madnami, "Part 5: Machine Translation Evaluation," *Handb. Nat. Lang. Process. Mach. Transl.*, no. 1, pp. 801–894, 2010.

[32] Common Sense Advisory, "The Value of Reliable Data and Insights," *CSA Res.*, vol. 4, 2019.

[33] N. Brad, "LLC dba Diplomatic Language Services," *Online*, 2019. [Online]. Available: http://dlsdc.com/blog/machine-translation-advantages-and-disadvantages/.

[34] Robin, "Natural Language Processing," *anguage.worldofcomputing.ne*, 2010. [Online]. Available: http://language.worldofcomputing.net/machine-translation/rule-based-machine-translation.

[35] S. P. Singh, A. Kumar, H. Darbari, L. Singh, A. Rastogi, and S. Jain, "Machine translation using deep learning: An overview," *2017 Int. Conf. Comput. Commun. Electron. COMPTELIX 2017*, no. April 2019, pp. 162–167, 2017.

[36] M. Denkowski, "Machine Translation for Human Translators," *Carnegie Mellon Univ.*, vol. 1, p. 97, 2015.

[37] J. KB., *Machine Translation Theory And History*, Fourth Edi. California: Association for the Advancement of Artificial Intelligence, 2019.

[38] S. Saini and V. Sahula, "A survey of machine translation techniques and systems for Indian languages," *Proc. - 2015 IEEE Int. Conf. Comput. Intell. Commun. Technol. CICT 2015*, no. Census 2001, pp. 676–681, 2015.

[39] L. Rossi and D. Wiggins, "Applicability and application of machine translation quality metrics in the patent field," *World Pat. Inf.*, vol. 35, no. 2, pp. 115–125, 2013.

[40] J. Zhang and C. Zong, "Deep Neural Networks in Machine Translation: An Overview," *IEEE Intell. Syst.*, vol. 30, no. 5, pp. 16–25, 2015.

[41] N. Hardeniya, *NLTK Essentials*, Six Editio. BIRMINGHAM - MUMBAI: Packt Publishing Ltd, 2018.

[42] A. Hurskainen and J. Tiedemann, “Rule-based Machine translation from English to Finnish,” vol. 2, pp. 323–329, 2018.

[43] V. Macketanz, E. Avramidis, A. Burchardt, J. Helcl, and A. Srivastava, “Machine Translation: Phrase-Based, Rule-Based and Neural Approaches with Linguistic Evaluation,” *Cybern. Inf. Technol.*, vol. 17, no. 2, pp. 28–43, 2017.

[44] H. W. Xuan, W. Li, and G. Y. Tang, “An advanced review of hybrid machine translation (HMT),” *Procedia Eng.*, vol. 29, pp. 3017–3022, 2012.

[45] K. Chunyu, P. Haihua, and J. J. Webster, “Example-based machine translation: A new paradigm,” *Transl. Inf. Technol.*, p. 57, 2002.

[46] M. Akter, M. Shahidur Rahman, M. Zafar Iqbal, and M. Reza Selim, “A Review of Statistical and Neural Network Based Hybrid Machine Translators,” *2018 Int. Conf. Bangla Speech Lang. Process. ICBSLP 2018*, pp. 1–6, 2018.

[47] D. Vilar, J. Xu, L. F. D’Haro, and H. Ney, “Error Analysis of Statistical Machine Translation Output,” *Proc. fifth Int. Conf. Lang. Resour. Eval. (LREC 2006)*, pp. 22–28, 2006.

[48] J. A. R. Fonollosa and M. R. Costa-jussà, “Latest trends in hybrid machine translation and its applications,” *Computer speech and language*, vol. 32, no. 1. pp. 3–10, 2015.

[49] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” pp. 1–38, 2015.

[50] F. M. Soares and A. M. F. Souza, *Neural Network Programming with Java*, Fourth Edi. BIRMINGHAM - MUMBAI: Packt Publishing Ltd, 2016.

[51] A. I. Wiki, “A Beginner’s Guide to Neural Networks and Deep Learning,” *skymind*, 2019. [Online]. Available: https://skymind.ai/wiki/neural-network.

[52] Z. Jian and W. X. Wu, “The application of feed-forward neural network for the X-ray image fusion,” *J. Phys. Conf. Ser.*, vol. 312, no. SECTION 6, 2011.

[53] K. Maladkar, “Types of Artificial Neural Networks Currently Being Used in Machine Learning,” *analyticsindia*. [Online]. Available: https://www.analyticsindiamag.com/.

[54] I. Sadeghkhani, A. Ketabi, and R. Feuillet, “Radial Basis Function Neural Network Application to Power System Restoration Studies,” *Comput. Intell. Neurosci.*, vol. 2012,

no. Figure 1, pp. 1–10, 2012.

[55] M. J. Er, S. Wu, J. Lu, and L. T. Hock, “Face recognition with radial basis function (RBF),” *IEEE Trans. Neural Networks*, vol. 13, no. 3, pp. 697–710, 2002.

[56] J. McCullock, “Kohonen Self-Organizing Maps,” *mnemstudio*. [Online]. Available: http://mnemstudio.org/neural-networks-kohonen-self-organizing-maps.htm.

[57] Oinkina Hakyll, “Understanding LSTM networks,” *colah*, 20115. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs.

[58] N. Ketkar, *Deep Learning with Python*, Fifth Edit. Bangalore, Karnataka: Apress media, 2017.

[59] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, “On the Properties of Neural Machine Translation: Encoder-Decoder Approaches,” *cs.CL*, vol. 2, 2014.

[60] O. Davydova, “7 types of Artificial Neural Networks for Natural Language Processing,” *medium.com*. [Online]. Available: https://medium.com/@datamonsters/artificial-neural-networks-for-natural-language-processing-part-1-64ca9ebfa3b2.

[61] A. Mehta, “A Complete Guide to Types of Neural Networks,” *Digitalvidya*. [Online]. Available: https://www.digitalvidya.com/blog/types-of-neural-networks/.

[62] A. Chinea, “Understanding the principles of recursive neural networks: A generative approach to tackle model complexity,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5768 LNCS, no. PART 1, pp. 952–963, 2009.

[63] O. Irsoy and C. Cardie, “Deep recursive neural networks for compositionality in language,” *Proc. Neural Inf. Process. Syst.*, pp. 2096–2104, 2014.

[64] L. ~M. Happel B. and M. ~J. Murre J., “The Design and Evolution of Modular Neural Network Architectures,” *Neural Networks*, vol. 7, pp. 985–1004, 1994.

[65] R. A. Jacobs and M. I. Jordan, “Learning Piecewise Control Strategies in a Modular Neural Network Architecture,” *IEEE Trans. Syst. Man Cybern.*, vol. 23, no. 2, pp. 337–345, 1993.

[66] M. N. Dailey and G. W. Cottrell, “Organization of face and object recognition in modular neural network models,” *Neural Networks*, vol. 12, no. 7–8, pp. 1053–1074, 1999.

[67] N. Ketkar, *Deep Learning with Python*, Fifth Edit. Bangalore, Karnataka: Apress media, 2017.

[68] E. project Focus, “Artificial Neural Networks (ANN) and Different Types,” *elprocus*. [Online]. Available: https://www.elprocus.com/artificial-neural-networks-ann-and-their-types/.

[69] K. Duh, “Deep Learning for Natural Language Processing and Machine Translation,” Tokiyo Japan, 2018.

[70] M. Swamynathan, *Mastering Machine Learning with Python in Six Steps*, Second Edi. Bangalore, Karnataka: Apress media, 2017.

[71] E. H. Almansor and A. Al-ani, *Machine Learning and Data Mining in Pattern Recognition*, vol. 3587. Springer International Publishing, 2005.

[72] E. S. leslie and U. G. Thomas, “G. B. Gene Students in Ancient oriental civilayzation,” *G. B. Gene Students Anc. Orient. civilayzation*, vol. No.60, no. S. leslie and U. G. Thomas, Eds, 1982.

[73] W. C. Oromo, “ethnologue language of the world,” *Oromo, West Central*, 2019. [Online]. Available: https://www.ethnologue.com/language/GAZ.

[74] T. A. A. Kebebew, “Afaan Oromo front matter,” *wikibooks.org*, 2018. [Online]. Available: https://en.wikibooks.org/wiki/Afaan_Oromo.

[75] S. Fattah, M. Rehn, E. Reierth, and T. Wisborg, “Systematic literature review of templates for reporting result,” pp. 1–8, 2013.

[76] S. Adugna, “English-Oromo Machine Translation: An Experiment Using a Statistical Approach,” Msc thesis Addis Ababa University, 2009.

[77] J. Daba, “‘Bidireactional English – Afaan oromo Machine translation using hybrid approach’, Msc thesis Addis Ababa University,” Addis Ababa University, 2013.

[78] M. Meshesha, “English-Afaan Oromo Statistical Machine Translation,” *Int. J. Comput. Linguist.*, vol. 9, no. 1, p. 6, 2018.

[79] T. M. and M. Y. R. Kimura, Z. Long, T. Utsuro, “Effect on reducing untranslated content by neural machine translation with a large vocabulary of technical terms,” *Proc. 7th PSLT*, vol. 3, pp. 9–20, 2017.

[80] E. M. L. Dahlmann, “Neural Machine Translation Leveraging Phrase-based Models in a Hybrid Search,” *Proc. EMNLP*, vol. 2, pp. 144–156, 2017.

[81] J. D. and A.Way, “Neural Pre-Translation for Hybrid Machine Translation,” *Proc. MT Summit XVI*, vol. 1, no. Research Track Nagoya, Sep18-22, pp. 17–29, 2017.

[82] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” *Methods Enzymol.*, vol. 101, no. C, pp. 582–598, 1983.

[83] H. Hu, X.; Li, W.; Lan, X.; Wu, H.; and Wang, “Optimized beam search with constrained softmax for nmt,” *MT Summit*, vol. XV, p. 8, 2015.

[84] H. Och, F. J., and Ney, “The alignment template approach to statistical machine translation,” *Proc. 41st Annu. Meet. Assoc. Comput. Linguist.*, vol. 30, pp. 417–449, 2004.

[85] D. P. Kingma and J. L. B. Ba, “ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION,” *Conf. Pap. ICLR*, vol. 9, pp. 1–15, 2017.

[86] F. Farhadi, “Learning Activation Functions in Deep Neural Networks,” *Introd. to Deep Learn. Bus. Appl. Dev.*, vol. 1, p. 171, 2017.

[87] C. E. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation Functions Comparison of Trends in Practice and Research for Deep Learning,” *IEEE Intell. Syst.*, vol. 1, pp. 1–20, 2018.

[88] C. A. and G. I. Bengio Y., “Deep learning,” *MIT Press*, 2019. [Online]. Available: http://www.deeplearningbook.org.

[89] Peltarion L., “Categorical crossentropy,” *AI model*, 2019. [Online]. Available: https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy.

[90] A. Rusiecki, “Trimmed categorical cross-entropy for deep learning with label noise,” *IEEE Trans. Neural Networks*, vol. 55, p. 12, 2019.

APPENDIXES

Appendix: A data source

1. https://siifsiin.com/2019/11/22/mirgoota-namoomaa-keewwata-13fi-14/
2. https://www.coursehero.com/file/p77p03q/The-speaker-of-the-City-Council-shall-be-accountable-to-the-City-council-Itti/
3. http://www.ethcriminalawnetwork.com/system/files/FDRE%20Criminal%20Code%20-%20Afan%20Oromo.pdf
4. https://www.unodc.org/cld/document/eth/2005/the_criminal_code_of_the_federal_democrati c_republic_of_ethiopia_2004.html Criminal code English version 2.
5. http://www.abyssinialaw.com/codes-commentaries-and-explanatorynotes?download=1208:fdre-criminal-code-afaan-oromo-version 3. https://www.google.com/search?client=opera&q=heera+motummaa+naannoo+oromiyaa &so urceid=opera&ie=UTF-8&oe=UTF-8#q=heera+mootummaa+feederaalawaa+itoophiyaa+pdf Heera mootumaa Ethiopia
6. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=17&cad=rja&uact=8 &ved=0ahUKEwjo Ethiopian constitution
7. https://www.lds.org/scriptures/nt/matt/1?lang=eng Holly bible English version
8. https://app.box.com/s/08vsopn8cwb63rv32yth Holly bible Afaan Oromo version.
9. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0ahUKEwj9h5nt5fvTAhVM82MKHQ_MAdUQFggxMAI&url=http%3A%2F%2Fextwprlegs1.fao.org%2Fdocs%2Fpdf%2Feth153469.pdf&usg=AFQjCNGbexcFJpSJBb9jyfcIl7 mXO4zhUA Megleta Oromia both in English and Afaan Oromo.
10. https://www.coursehero.com/file/p77p03q/The-speaker-of-the-City-Council-shall-be-accountable-to-the-City-council-Itti/

Appendix: B. Python source code

B.1 Word to vector sample code

```
import tensorflow as tf
import numpy as np

corpus_raw = 'Alaabaan itiyoophiyaa gara irraa magariisa gidduun keelloo jalaan diimaa ta'ee ni qabaa
The Ethiopian flag shall consist of green at the top, yellow in the middle and red at the bottom.'

# convert to lower case
corpus_raw = corpus_raw.lower()

words = []
for word in corpus_raw.split():
    if word != '.': # because we don't want to treat . as a word
        words.append(word)

words = set(words) # so that all duplicate words are removed
word2int = {}
int2word = {}
vocab_size = len(words) # gives the total number of unique words

for i,word in enumerate(words):
    word2int[word] = i
    int2word[i] = word

# raw sentences is a list of sentences.
raw_sentences = corpus_raw.split('.')
sentences = []
for sentence in raw_sentences:
    sentences.append(sentence.split())

WINDOW_SIZE = 2
```

```
data = []
for sentence in sentences:
    for word_index, word in enumerate(sentence):
        for nb_word in sentence[max(word_index - WINDOW_SIZE, 0) : min(word_index + WINDOW_SIZE, len(sentence)) + 1] :
            if nb_word != word:
                data.append([word, nb_word])

# function to convert numbers to one hot vectors
def to_one_hot(data_point_index, vocab_size):
    temp = np.zeros(vocab_size)
    temp[data_point_index] = 1
    return temp

x_train = [] # input word
y_train = [] # output word

for data_word in data:
    x_train.append(to_one_hot(word2int[ data_word[0] ], vocab_size))
    y_train.append(to_one_hot(word2int[ data_word[1] ], vocab_size))

# convert them to numpy arrays
x_train = np.asarray(x_train)
y_train = np.asarray(y_train)

# making placeholders for x_train and y_train
x = tf.placeholder(tf.float32, shape=(None, vocab_size))
y_label = tf.placeholder(tf.float32, shape=(None, vocab_size))
```

```
EMBEDDING_DIM = 5 # you can choose your own number
W1 = tf.Variable(tf.random_normal([vocab_size, EMBEDDING_DIM]))
b1 = tf.Variable(tf.random_normal([EMBEDDING_DIM])) #bias
hidden_representation = tf.add(tf.matmul(x,W1), b1)

W2 = tf.Variable(tf.random_normal([EMBEDDING_DIM, vocab_size]))
b2 = tf.Variable(tf.random_normal([vocab_size]))
prediction = tf.nn.softmax(tf.add( tf.matmul(hidden_representation, W2), b2))


sess = tf.Session()
init = tf.global_variables_initializer()
sess.run(init) #make sure you do this!

# define the loss function:
cross_entropy_loss = tf.reduce_mean(-tf.reduce_sum(y_label * tf.log(prediction), reduction_indices=[1]))

# define the training step:
train_step = tf.train.GradientDescentOptimizer(0.1).minimize(cross_entropy_loss)

n_iters = 10000
# train for n_iter iterations

for _ in range(n_iters):
    sess.run(train_step, feed_dict={x: x_train, y_label: y_train})
    print('loss is : ', sess.run(cross_entropy_loss, feed_dict={x: x_train, y_label: y_train}))

vectors = sess.run(W1 + b1)
```

```
def euclidean_dist(vec1, vec2):
    return np.sqrt(np.sum((vec1-vec2)**2))

def find_closest(word_index, vectors):
    min_dist = 10000 # to act like positive infinity
    min_index = -1
    query_vector = vectors[word_index]
    for index, vector in enumerate(vectors):
        if euclidean_dist(vector, query_vector) < min_dist and not np.array_equal(vector, query_vector):
            min_dist = euclidean_dist(vector, query_vector)
            min_index = index
    return min_index


from sklearn.manifold import TSNE

model = TSNE(n_components=2, random_state=0)
np.set_printoptions(suppress=True)
vectors = model.fit_transform(vectors)

from sklearn import preprocessing

normalizer = preprocessing.Normalizer()
vectors =  normalizer.fit_transform(vectors, 'l2')

print(vectors)
```

```
import matplotlib.pyplot as plt


fig, ax = plt.subplots()
print(words)
for word in words:
    print(word, vectors[word2int[word]][1])
    ax.annotate(word, (vectors[word2int[word]][0],vectors[word2int[word]][1] ))
plt.show()
```

B.2. To construct BiRNN cell encoder forward and backward cells

```
# Build RNN cell
encoder_cell = tf.nn.rnn_cell.BasicLSTMCell(num_units)

encoder_outpus: [max_time, batch_size, num_units]
encoder_state: [batch_size, num_units]
encoder_outputs, encoder_state = tf.nn.dynamic_rnn(
    encoder_cell, encoder_emb_inp,
    sequence_length=source_sequence_length, time_major=True)
```

```
# Construct forward and backward cells
forward_cell = tf.nn.rnn_cell.BasicLSTMCell(num_units)
backward_cell = tf.nn.rnn_cell.BasicLSTMCell(num_units)

bi_outputs, encoder_state = tf.nn.bidirectional_dynamic_rnn(
    forward_cell, backward_cell, encoder_emb_inp,
    sequence_length=source_sequence_length, time_major=True)
encoder_outputs = tf.concat(bi_outputs, -1)
```

B.3. Loss logists

```
crossent = tf.nn.sparse_softmax_cross_entropy_with_logits(
    labels=decoder_outputs, logits=logits)
train_loss = (tf.reduce_sum(crossent * target_weights) /
    batch_size)
```

B.4. To construct BiRNN decoder cell

```
# Build RNN decoder cell
decoder_cell = tf.nn.rnn_cell.BasicLSTMCell(num_units)

# Helper
helper = tf.contrib.seq2seq.TrainingHelper(
    decoder_emb_inp, decoder_lengths, time_major=True)
# Decoder
decoder = tf.contrib.seq2seq.BasicDecoder(
    decoder_cell, helper, encoder_state,
    output_layer=projection_layer)
# Dynamic decoding
outputs, _ = tf.contrib.seq2seq.dynamic_decode(decoder, ...)
logits = outputs.rnn_output
projection_layer = layers_core.Dense(
    tgt_vocab_size, use_bias=False)
```

B.5. Calculating minibatch gradient descent Adam Optimizer

```
# Calculate minibatch gradients descent Optimizer
params = tf.trainable_variables()
gradients = tf.gradients(train_loss, params)
clipped_gradients, _ = tf.clip_by_global_norm(
    gradients, max_gradient_norm)
optimizer = tf.train.AdamOptimizer(learning_rate)
update_step = optimizer.apply_gradients(
    zip(clipped_gradients, params))
```

B.6. To define beam search

```
# Define a beam-search
decoder_initial_state = tf.contrib.seq2seq.tile_batch(
    encoder_state, multiplier=hparams.beam_width)

decoder = tf.contrib.seq2seq.BeamSearchDecoder(
        cell=decoder_cell,
        embedding=embedding_decoder,
        start_tokens=start_tokens,
        end_token=end_token,
        initial_state=decoder_initial_state,
        beam_width=beam_width,
        output_layer=projection_layer,
        length_penalty_weight=0.0)

outputs, _ = tf.contrib.seq2seq.dynamic_decode(decoder, ...)
```