PJET, Vol. 2, No. 1, February 2024: 22-43

Poly Journal of Engineering and Technology

journal homepage: https://journals.bdu.edu.et/index.php/PJET/index

ORIGINAL ARTICLE

GE'EZ-AMHARIC MACHINE TRANSLATION USING DEEP LEARNING

Genet Worku Gebeyehu [1], Tsegaye Abebe Altaye [2], Esubalew Alemneh Jalew[2,*]

[1] Faculty of Computing, Bahir Dar Institute of Technology, Bahir Dar University, P. O. Box 26, Bahir Dar, Ethiopia

[2] ICT4D Research Center, Bahir Dar Institute of Technology, Bahir Dar University, P. O. Box 26, Bahir Dar, Ethiopia

ABSTRACT

Neural machine translation (NMT) which has come to be the breakthrough in the field of machine translation is now greatly being used by many translation services such as google translate. This generic deep learning approach of machine translation (MT) with the help of attention mechanism is used as the core method of our Ge'ez-Amharic translation. Other researchers have applied Statistical Machine Translation (SMT), morpheme-based translation methods with the involvement of linguistic professionals for Ge'ez-Amharic translation. But they have recorded low accuracy, low speed of translation. The limitation of these approaches is the complex nature of the design of the models. We used a unidirectional model which only translates from Ge'ez to Amharic. We designed an NMT encoder-decoder translation model based on attention mechanism that contains two Long Short-Term Memory (LSTM) layers with 500 hidden units both in the encoder and decoder parts. The model takes source sentence as input in the encoder side and generates a target sentence as output in the decoder side, generating a single word at a time. We used attention mechanism to handle long term dependencies in long sentences by paying attention to the parts of the input sentence which contain relevant information in generating a single word in the target sentence. We have collected 50k Ge'ez-Amharic parallel sentences and used different portions of this data for the different experiments. OpenNMT was employed for developing our model and Bilingual Evaluation Under Study (BLEU) is used for evaluating the translation quality of our model. The proposed model was trained using different experiments on Colab and we found the best performing translation with BLEU score of 15.4%. Despite the hungry nature of NMT models for data, the model has performed well with the collected corpus.

Keywords: *Amharic, Ge'ez, Translation, NMT, ANN, attention mechanism*

DOI: https://doi.org/10.20372/pjet.v2i1.1718

Corresponding Author: Esubalew Alemneh Jalew

ICT4D Research Center, Bahir Dar Institute of Technology, Bahir Dar University, P. O. Box 26, Bahir Dar, Ethiopia

Email esubalew@gmail.com

1. Introduction

Communication is the way of transferring information from one to the other using the fundamental element called language. Languages in world are different in syntax and word order even in their phonemes. This is a basic problem in realizing a good global communication through world. Different or /and similar machine translation models that translate from one natural language to the other are developed [1]. However, there are languages that are not yet included in most advanced machine translation systems, one of which is Ge'ez. One of branch of Natural Language Processing (NLP) called Machine translation (MT) supports in communication between machine-to-machine (e.g. question answering) and human-to-human [2].
NLP is a combination of computational methods, which are theory-driven, used for human language representation and analysis. Deep learning is a recently used approach for one of the NLP categories, machine translation. Unlike traditional machine translation, neural machine translation is a better choice for more accurate translation and it also provides better performance [3]. Deep learning which uses neural networks, is a type of learning in which researchers are attracted to the field of machine translation [4]. The basic motive behind NMT is to create a trained model for a system that will work as a translator by having learned from previous experiences and antiquity without the need to use linguistic rules. Translation in its simple definition includes: defining the source text and re-encoding this translation into the target language. The translator fully decodes the meaning of the original text. MT aims to translate text from a source language into a target language, ensuring it reflects the same meaning for bilingual or multilingual systems. Unidirectional bilingual systems translate from the source language to the target language, while bidirectional systems translate both directions. Most bilingual systems are bidirectional, while multilingual systems are unidirectional. Rule-based machine translation (RBMT) is a major MT approach, requiring linguistic professionals to define rules for the translation process. It involves analyzing the morphology, syntax, and semantics of both languages, and generating a text in the target language. We also need the bilingual dictionary of source and target languages [5]. The sub approaches of Rule-based approaches are Direct, Transfer based and Interlingua approaches. The Corpus-based approach is a data-driven machine translation method that automatically extracts rules from a bi-language parallel corpus. It is an alternative to the rule-based approach and uses sub approaches such as Statistical Machine Translation (SMT), Example-based Machine Translations (EBMT), and Natural Language Translation (NLT). Neural Machine Translation (NMT) is a type of the corpus-based MT approach and it is also called sequence to sequence model, data-driven or corpus-driven machine translation [6]. It is a deep neural network model that happened to be the state-of-the-art machine translation nowadays. The statistical models of the machine translation are learned by the use of neural network models. Google Translate, Baidu Translate are well-known examples of NMT that are made available to the public on the Internet.

Neural machine translation technology is currently the cutting-edge technology in machine translation and offers the highest quality translation. The main benefit to using NMT is that the system is trained on the source and target text directly without having to leverage any specialized systems which SMT systems used. According to [7] and [8], neural machine translation builds and trains a single, large neural network that reads a sentence as input and produces a correct translation as output in an end-to-end fashion (mapping from input sentence to its corresponding output sentence). Separate models such as the language model, translation model, and reordering model are not needed, it is just a large single sequence model (neural network) that predicts one word at a time.

In NMT, instead of translating word for word, or looking at shorter strings of words in parallel, it can take whole sentences and more into context at once. The results of this are more fluent, accurate, and usable translations than ever before. NMT doesn't care about the different meanings of letters with the same phonetics, Ge'ez language, for example, has three letters with the same phonetics ሀ፡ ሐ and ኸ, since it is context based.

Ge'ez is an ancient language used in the Horn of Africa, Ethiopia, and Eritrea. It is a "pure" Semitic language compared to Amharic and other Ethio-Semitic languages. It was a major language and linguistic king until the beginning of the twentieth century. Thereafter, it gradually started to disappear and eventually replaced with Tigrigna in northern Ethiopia and with Amharic in Central.

Ge'ez script is an alpha syllabary script also called "Abugida", in which a character represents a consonant and a vowel combination. This is different from an alphabetic script where a character represents one sound either a consonant or a vowel. The alphabet of the Amharic script is unique scripts acquired from Ge'ez and use an alpha syllabary writing system where the consonant and vowel are combined to form a single symbol. Thus, once a person knows all the alphabets, he/she can easily read and write both Ge'ez and Amharic. Scripts in Ge'ez include 26 basic alphabets called 'Fidel' whereas there are 34 alphabets in Amharic scripts [9].

Amharic is a Semitic language spoken in Ethiopia. **It originally came from the northern part of Ethiopia and is spoken by the people who live in an area known as Amhara. Later on, however, the language became an official language that is spoken all over the country. Thus, currently it was declared as a working language across the country, Ethiopia. It has 31.8 million speakers as mother tongue and 25 million speakers as second language** [10]**.**

Amharic has its own writing system. The Amharic alphabets are written from left to right. It contains 34 alphabets and 26 of them are derived from the Ge'ez alphabets. The remaining eight alphabets are modifications of the Ge'ez alphabets which are; ሰ to ሸ ፡ ተ to ቸ ፡ ነ to ኘ ፡ ከ to ኸ ፡ ዘ to ዠ ፡ ደ to ጀ ፡ ጠ to ጨ and በ to ቨ [11]. These 34 consonant symbols, with seven variations, and variations according to the vowel that is paired with a consonant. Each letter or symbol usually represents the entire syllable [12] [13].

There is a lot of literature written in the Ge'ez language which contains ancient but yet very important data. The vast literature written in Ge'ez derives expertise in different varieties of our life. Religious texts such as the Bible, theological, and magical texts, stories saints, religious poetry, and angels are some of the literatures which includes important ideas and philosophies.

Therefore, the main problem here is understanding the very important data or information in this literature. The Ge'ez Amharic MT is supportive for reading and communication purposes, as well as for language education. Translating Ge'ez text documents into Amharic text is currently very difficult. Different approaches have been proposed to address this difficulty. However, MT using deep learning is a state-of-the-art (SOTA) method with an improved quality of translation, fluency, speed, and accuracy. The previous approaches which require a linguistic professional in the translation process. In addition to this the model design process is more complex which needs independent design of components included in the translation.

The contribution of this study is that it's results (models) that can be used to develop machine translation software for Ge'ez to Amharic, which will be used to translate huge litterateurs in Ge'ez to Amharic. Researchers who need to take part in achieving the goal of developing an efficient NLP system for the Ge'ez language can greatly benefit from this work. Since NMT is much faster and accurate than a human translator is, an NMT service translates the available Ge'ez contents into the Amharic language quicker than ever before and to inspire students for learning the Ge'ez language.

The remaining sections are organized as follows. Section two describes and summarizes related works. Section three introduces the methodology used in the study. Section four presents the proposed new system called Ge'ez-Amharic NMT Model with attention. The experiment of the study is presented at section five. Section six contains results and discussion and the last section a conclusion of the study.

1. Related Works

In this section, we summarized previously conducted related works on machine translation for local and foreign languages.

English to Amharic SMT

English to Amharic statistical machine translation (EASMT) was conducted by [14]. The main purpose of the study is to develop English to Amharic machine translation using SMT. In the EASMT system, the experiment was conducted in the training corpus of both languages based on the words contained in the parallel documents.

The experiment was conducted on a targeted translation system using 18,432 English-Amharic parallel sentences, with 90% used for training and the remaining 10% for testing and validating the system. In addition to the parallel corpora, the researchers used 254,649 monolingual corpora. The monolingual corpus is used for Language Modeling (LM). Accordingly, the baseline phrase-based BLEU score result was 35.32%.

The preliminary experiment demonstrates EASMT's ability to translate basic English sentences to Amharic sentences, but it has pros and cons, including addressing issues like untranslated words, translation errors, and word segmentation, which require strong aspects.

The researchers in [14] recommended that more experiments and research are needed to further improve EASMT translation accuracy. The experiment done so far is as promotive as the translation is made from less fluent English to morphologically rich Amharic.

Bidirectional English-Amharic MT

English to Amharic machine translation using SMT approach with constrained corpus is conducted by [15]. The experiment involved two corpora with simple and complex sentences, constructing a bidirectional translation model based on the probability of a source sentence generating a target sentence, and using a decoder and expectation-maximization algorithm for word alignment.

According to [15], an experiment was carried out for the first corpus and the second corpus separately. The BLEU result of the experiment conducted using the first corpus for English to Amharic translation was 82.22% and 90.59% for Amharic to English translation was recorded. The result of the experiment using the second corpus was approximately 73.38% and 84.12% for English to Amharic and for Amharic to English respectively. Finally, [15] concluded that with a large amount of dataset available, a good translation performance could be achieved as there would be more words in the corpus which will maximize the probability that a word precedes another.

Amharic to Tigrigna MT

Amharic to Tigrigna MT is done by [16] using a statistical machine translation approach. They used 27,470 parallel Amharic and Tigrigna sentences for training, the selected corpus has been preprocessed and analyzed morphologically using confessor.

Researchers suggest using units of words and morphemes for Amharic and Tigrigna in an experiment, using multi-threaded Giza (MGIZA) software for data alignment. In this work, words or morphemes were used as a translation unit. This study selects sentences with 80 words per sentence without special characters, including colons and exclamation marks. The results are promising, proving the feasibility of an SMT system for implementing translation systems for local languages.

The Bilinguale Evaluation Understudy BLEU score result of the translation from Tigrigna-Amharic and Amharic-Tigrigna is 6.65 and 8.25 respectively regarding to each translation unit. The translation result is increased to be 13.49 and 12.93 for Amharic-Tigrigna and Tigrigna-Amharic respectively using morphemes as units for Amharic and Tigrigna.

Amharic-to-Tigrigna MT Using a Hybrid Approach

In this work, [17] used a hybrid approach i.e., combination of SMT and RBMT. A syntactical reordering approach is proposed to align word order in the source language so that a similar order of words in the target language is achieved. A unidirectional language model is developed in the research to assign a probability that a given input sentence in the source language generates an output sentence in the target language. Two major experiments carried out by [17] came out with BLEU results of 7.02% using the statistical machine translation approach and 17.47% using the hybrid machine translation approach.

Bidirectional Tigrigna – English SMT

This work was conducted in 2017 by [18], which aimed to investigate the development of the Tigrigna - English bidirectional machine translation system using a statistical approach. The main aim of the work was to design a bidirectional Tigrigna to English machine translation. So, to achieve their aim, they prepared a corpus collected from different domains and classified into five sets of corpora and arranged a format appropriate for use in the development process.

The researcher conducted three sets of experiments: baseline (phrase-based machine translation system), morph-based (based on morphemes obtained using the unsupervised method), and post-processed segmented systems (based on morphemes obtained by post-processing the output of the unsupervised segmenter). A free statistical machine translation framework called MOSES which allows automatically training translation model using parallel corpus was used.

The translation evaluation metric used was BLEU, according to their experimental result, the BLEU score of 53.35 % for Tigrigna – English and 22.46 % for English – Tigrigna translations. Finally, [18] recommended that "future research should focus to further improve the BLEU score by applying semi-supervised segmentation to include the remaining linguistic information".

Ge'ez to Amharic MT

Ge'ez to Amharic MT was done by [19], using a statistical machine translation approach. The research came with the aim of addressing Amharic speakers getting knowledge decoded in Ge'ez using automated translation techniques is mandatory. The methodology used was a qualitative experimental method to examine the effect of variables such as normalization, corpus, and test segmentation options of SMT results.

The data used for the experiment were found both from the internet and prepared manually. The data collected by [19] was in a different format so to make those different formats suitable for the experiment, the researcher merges all documents to Ms.-Word format and aligns to verse/sentence level, cleaned for noisy characters, and converted to plain text in UTF-8 format.

As described by [19], they used Old Testament (9 documents), Wudasie Mariam and Arganon. The number of parallel sentences, 12, 860 for both Ge'ez and Amharic languages.
As for data organization, bilinguals, 90% for training, and 10% for the testing were used for the experiment. GIZA ++, IRSTLM, Moses decoder, and BLEU were used to it building a translation model, a linguistic model, an alignment of words, and a Ge'ez evaluation for the Amharic MT system, respectively. The parallel corpus used for the experiment was aligned at the sentence level.

As stated by [19], the translation result was high when test data was taken from psalm as a complete low when the testing data contains sentences from the praise of Saint Mary and part of the Bible using 10-fold cross-validation. The results show inconsistency.

After experimenting, had an average accuracy of translating the BLEU score of 8.26. Using a sufficiently large parallel Ge'ez-Amharic corpus language and collection synthesis tool, it is possible to develop a better translation system for language pairs.

Finally, [19] recommended that Ge’ez and Amharic are related, but morphologically complex and limited research were performed in the morphological segmentation and in the synthesis of the two languages. The researcher suggests extending research using different morphological and synthesis mechanisms to improve performance in language morphological synthesizers and segmentation tools. The NMT approach can resolve the complexity of Ge'ez and Amharic languages by training a single end-to-end system with parallel corpus.

Morpheme-Based Bi-directional Ge'ez to Amharic MT

Bidirectional Ge'ez to Amharic MT using Morpheme-Based approach was done by [13]. It is the second Ge'ez to Amharic MT next to [19]. As stated by [13], the aim of the research was to address the problem observed in the manual translation are time-consuming, resource-intensive, and linguistic knowledge language is required. The tools used are MGIZA++, which is a software-based on the famous word-alignment software GIZA++, and morfessor for word-level and BLEU for evaluating the result.

Two experiments were conducted in this study, using unsupervised segmentation and rule-based segmentation. The BLEU score of the dataset prepared by using unsupervised segmentation was 14.54% and 14.88% from Ge'ez to Amharic and from Amharic to Ge'ez respectively.

The BLEU score of the dataset prepared by using rule-based segmentation was 15.14 and 16.15 from Ge'ez to Amharic and from Amharic to Ge'ez. [13] recommended that the alignment of the Ge'ez to Amharic text is a challenging task because of many to many correspondences between words/ morphemes of the two languages. Therefore, it is necessary to identify the ideal alignment for the Ge'ez to Amharic Machine translation. The alignment challenge caused by the many to many correspondences between words of both languages is not an issue in applying an NMT approach because a contextual meaning of words in a sentence is extracted using an attention mechanism.

In previous works that we reviewed them were used morpheme-Based approach, rule-based machine translation, statistical machine translation, and / or hybrid approaches. Therefore, the aim of our study is that to apply NMT approach with attention mechanism.

2. METHODOLOGY

In this section we discussed about data collection processes, necessary tools used and the proposed model.

Data collection

For experimental purposes, a corpus of 50k parallel sentences is collected from religious institutions and other related sources such as Ethiopian Orthodox Tewahdo church religious books[1] including the holy bible i.e., old and new testaments, Wudasie Maryam, Drsane Gebriel, Drsane Michael. The correctness of parallel sentences (Geez and

[1] www.ethiopicbible.com

Amharic) in the meaning was validated by two Ge'ez experts. We divided the collected data into training, validation, and testing set as shown in Table 1. Generally, proportion is selected to make the training data set relatively large for the different experiments we conducted. The rest of the proportion is used for validating and testing our model. The data used for validation is used to check whether the training converges after certain number steps or not. The amount of data used in the validation and testing is almost similar except in our second experiment which contained 2k sentences for validation and 1k sentences for testing.

Table 1: Size of the total data used in the study for the different experiments

Experiment	Training data		Validation data		Testing data		Total	
	Geez	Amharic	Geez	Amharic	Geez	Amharic	Geez	Amharic
1	46k	46k	2k	2k	2k	2k	50k	50k
2	47k	47k	2k	2k	1k	1k		
3	48k	48k	1k	1k	1k	1k		
4	48k	48k	1k	1k	1k	1k		

Tools

OpenNMT

Neural machine translation uses an open-source toolset known as OpenNMT. Its two state-of-the-art deep learning foundations are OpenNMT-py (implemented in PyTorch), and OpenNMT.tf (implemented using TensorFlow). Due to its highly customizable model architectures and efficient training procedures, we prefer OpenNMT-py to OpenNMT-tf. This open-source translation toolkit also includes many modules such as CTranslate2 (inference engine), Tokenizer (C++ tokenization library), and a docker interface (for training and translating using docker containers) to facilitate the overall translation procedures. The fact that NMT systems take from days to weeks to train is a factor for considering the training efficiency in the design of OpenNMT-py. We used python programming language since Python is an efficient, technical language that allows developers to complete more work with fewer lines of code and is easily recognizable by humans.

Evaluation tool

OpenNMT provides baseline performance for output measurement parameters like BLEU, TER, and DLRATIO, which can be used for training or independent tools. BLEU is a popular quality review tool at MT Systems. BLEU scores can be computed either at a document level or at a sentence level [20] [21]. They range between 0 or 0% is the lowest quality and it is completely irrelevant to reference and 1 or 100% refers to the highest quality means the same as a reference.

Ge'ez-Amharic NMT Model with attention

In this section, we first briefly describe how our translation model with attention mechanism works with the generic encoder-decoder architecture of the NMT system (Figure 3: 2 and Figure 3), and components included in the architecture are also discussed separately.

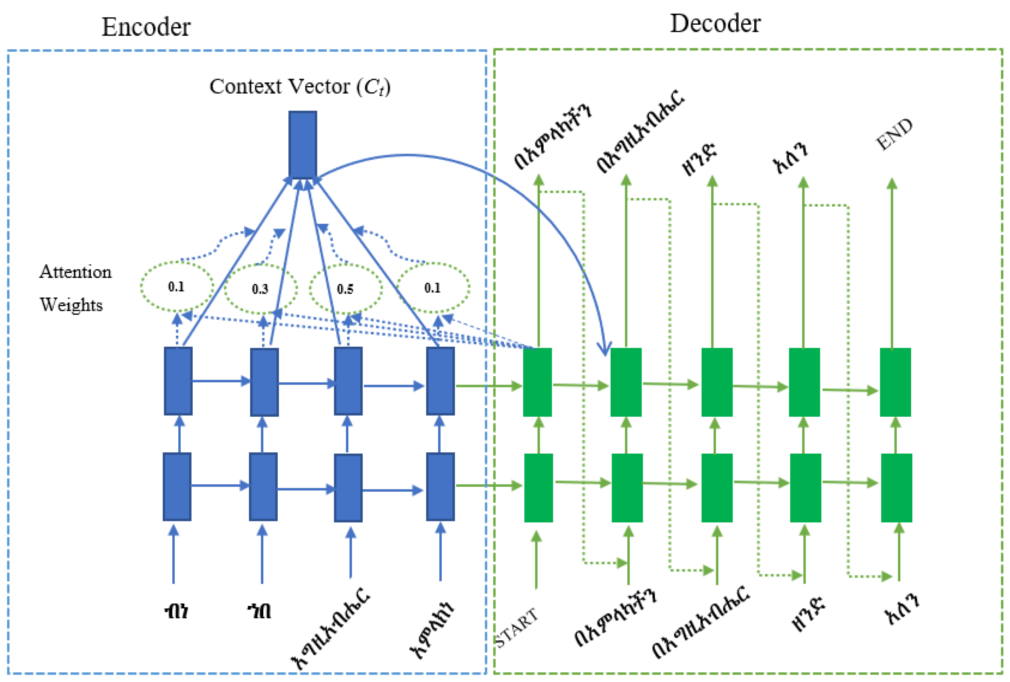

Figure 2: Architecture of our translation model

The translation model contains a 2-layer LSTM network with 500 hidden layers on both the encoder and the decoder side. This Ge'ez – Amharic model is different from other translation models proposed by previous Ge'ez-Amharic translations in that it is a single end-to-end model that is trained to translate input sentence into its corresponding target sentence without having to construct multiple independent components such as language model, translation model, and others.

As shown in Figure 2:, our translation model contains two main components, the encoder, and the decoder. The Ge'ez sentence (ብነ ኅበ እግዚአብሔር አምላክነ) is fed into the LSTM based RNN (blue color) of the encoder as source sentence. Then the encoder RNN computes the representation of the input sentence in the form of hidden states (pool of source states) in the different layers of its network. The decoder RNN (green color) then generates the Amharic sentence or target sentence (በአምላካችን በእግዚአብሔር ዘንድ አለን) one word at a time by looking back into the pool of source states in the encoder every time it generates a word.

As it described in the *attention mechanism* section, a fixed-dimensional representation of the input sentence is a problem when translating longer sentences. On the other hand, using the attention mechanism will help us to make the entire pool of source states available to the decoder in the translation process.[2]

When translating any particular word, the decoder needs to work out which one, out of the pool, it wants to draw from. So effectively, the pool of source states is a random-access memory also known as LSTM which the neural network

[2] Most commonly, the highest level of hidden state in the encoder RNN is considered for attention.

is then going to be able to retrieve as needed when it wants to do its translation. Attention for neural machine translation is one specific example of this. The attention model specifically tells which part of the source sentence to translate next. To generate the next word, we need to use the hidden states from the LSTM cells in the encoder (blue part of Figure 2:) and do a comparison with the hidden state of the previously generated word in the decoder (green part of Figure 2:). Upon comparison, each component of the LSTM is scored with an attention function shown in equation 3.

$$a_{t(s)} = \frac{e^{score(h_t ,\bar{\boldsymbol{h}}_{\boldsymbol{s}})}}{\sum_s e^{score(h_t,\bar{\boldsymbol{h}}_{\boldsymbol{s}})}} \qquad 1$$

We then do a combined representation of all the memories weighted by the score which comes to be the alignment weights or *attention weights*. A SoftMax function can easily do this for us (equation 1). At each time step t, the model computes a variable-length attention weight vector a_t from the previous hidden sate $\boldsymbol{h_{t-1}}$ and all hidden states of the source $\bar{\boldsymbol{h}}_{\boldsymbol{s}}$. A context vector c_t is computed as a weighted sum of all source states (see equation 2).

$$c_t = \sum_s a_t\,(s)\bar{\boldsymbol{h}}_{\boldsymbol{s}} \qquad 2$$

To put a score for each LSTM in the encoder, we used a bilinear attention function shown in equation 3, where $\boldsymbol{h_t}$ and $\bar{\boldsymbol{h}}_{\boldsymbol{s}}$ are the decoder and encoder hidden states respectively and $\boldsymbol{W_a}$ is a matrix that determines how much weight we want to put on the **dot product** of the two vectors.

$$score(h_t, \bar{h}_s) = \boldsymbol{h_t}^{\boldsymbol{T}}\boldsymbol{W_a}\bar{\boldsymbol{h}}_{\boldsymbol{s}} \qquad 3$$

Encoder-Decoder Model of NMT

At the very early stages of NMT, multilayer perceptron neural network models were used for translation where fixed-length input sentence is taken and output sentence of the same length is produced. These multilayer perceptron models have been hugely improved by using recurrent neural networks embedded into encoder-decoder architecture making it possible to use variable-length input and output sentences.

The architecture of NMT contains a network of encoders and decoders, where the encoder part of the network encodes a source sentence into a fixed-size vector from which different target translations can be made [22]. The decoder part of the network takes the fixed-size vector of the encoder then generates a translation in the target language. Both the encoder and the decoder are jointly trained to maximize correct translation probability [7].

To generate the fixed-size vector, the encoder RNN (Figure 3:) reads the input sentence as a sequence of vectors $X = (x1, \cdots, x_{T_x})$ *(equation 4)* and convert it into a vector *c (equation 5*

$$h_t = f(x_t, h_{t-1}) \qquad 4$$

$$c = q(\{h_1, \cdots, h_{T_x}\}) \qquad 5$$

where h_t is a hidden state at time t and c is a vector generated from a sequence of hidden states which we use as an input sentence representation.

The decoder is then trained to predict the next word y_t, given the vector c and all the words predicted previously (equation 6). The decoder, on the other hand, computes probability over translation y partitioning the joint probability into ordered conditionals.

$$p(y) = \prod_{t=1}^{T} p(y_t \mid \{y_t, \cdots, y_{t'-1}\}, c) \tag{6}$$

The fixed-length vector generated by the encoder part of the neural network introduces an issue into the field of machine translation, being a bottleneck for the performance achievement of the encoder-decoder architecture. It makes it difficult for the neural network to deal with long sentences, mainly for sentences that are longer than sentences in the training data.

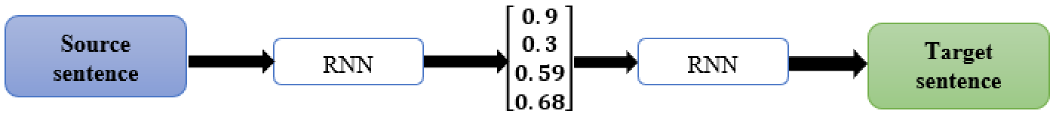

Figure 3: Architecture of NMT

An extension of this encoder-decoder architecture that learns to jointly translate and align (attend) was introduced by [7] as a solution for the difficulty. This, extended, architecture of the encoder-decoder, searches for parts of the source sentence that provided relevant information while generating the target word at each time step. Generally, the issue of long-term dependencies among words of a sentence is resolved with a mechanism known as the *attention mechanism*.

The basic idea behind the NMT system is to predict the target language string Y = (y1, ..., yt) for a particular source language string X = (x1, ..., xs). It is a conditional distribution modeled with an RNN-based encoder-decoder architecture. The encoder extracts a variable-length sentence from the source language and converts it to a fixed-length vector. This constant vector is called the sentence embedding and contains the meaning of the inserted sentence. The decoder then takes care of recording that sentence and starts guessing the words in the output, taking into account the context of each word as input.

A simplified example of a Ge'ez to Amharic machine translation: "እስመ አልቦ ነገር ዘ ይስአኖ ለ እግዚአብሔር "is encoded into numbers 251, 3245, 953, 2, 552, 2388, 3. The numbers 251, 3245, 953, 2, 552, 2388, and 3 are input into a neural translation model and results in output 2241, 9242, 98, 6342, 2241 is then decoded into the Amharic translation "ለ እግዚአብሔር እሚሳነው ነገር የለም" (each number in the input and output represents a word in the Ge'ez and Amharic dictionary and are always encoded and decoded accordingly) [23].

Recurrent Neural Networks (RNN)**Error! Bookmark not defined.** is one of the algorithms behind the remarkable results that h ave been observed in deep learning for the past few years. The most powerful and robust neural network algorithms which are guaranteed to be the only neural networks with internal memory are RNNs.

LSTM: Long-Short term memory is one of the topologies of recurrent artificial neural networks. Unlike basic repetitive artificial neural networks, it can learn from its experience to process, classify, and predict time series with very long delays of unknown magnitude between important events. Because of this, long-term memory outperforms other recurrent neural networks, hidden Markov models, and other sequencing methods. The artificial neural network with long-term memory consists of blocks of long-term memory that can memorize the value for a certain period of time. This is achieved with gates that determine when the input is meaningful enough to remember when to remember or forget, and when to display the value [24] [25].

Because of their internal memory, RNNs can remember important things about the input they received, this allows them to be very accurate in predicting what will happen next. RNNs also is known as recurrent because they perform the same task for every element of a sequence, with the output being dependent on the previous computations (equation.*4*).

For our MT model, we use a sequence-to-sequence model consists of two recurrent neural networks: an **encoder** that processes the input and a **decoder** that produces the output.

A broader description of the encoder-decoder model

The encoder, as its name indicates, encodes the input sequence into a fixed-size vector. The decoder takes the vector as input and produces a translation after processing the vector [26].

Encoder: The task of the encoder is to convert the given source sentence into numbers, more specifically vectors (Figure 3:) and metrics which are just an assortment of numbers representing data since computers do not understand sentences as humans do. The input words are processed using RNNs resulting in hidden states (equation *4*). The hidden states encode each word with all the preceding words i.e., left context. An RNN that processes the words from right to left is also built to produce the right context, generally having two RNNs running in two different directions. This is called bidirectional RNNs.

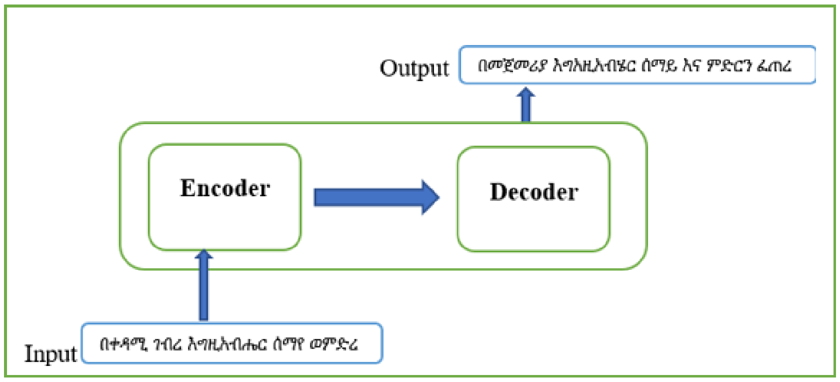

Figure 1: The encoder-decoder model

The encoder Architecture: The encoder architecture is built using multiple (deep) LSTM layers which can learn in multiple ways.

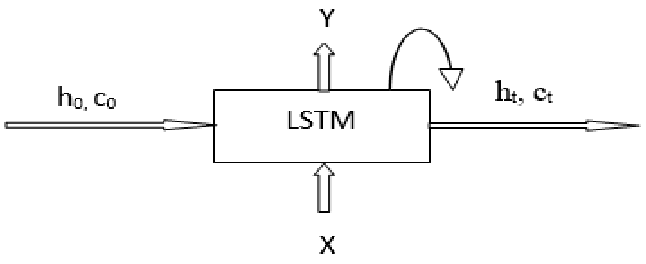

Figure 2: Single LSTM Unit

However, for simplicity purpose, let us describe how a single LSTM unit works. A single LSTM (Figure 2) unit receives three parameters as input and produces three outputs. X is a word from the input sentence, h0 and c0 are hidden state and vector state respectively.

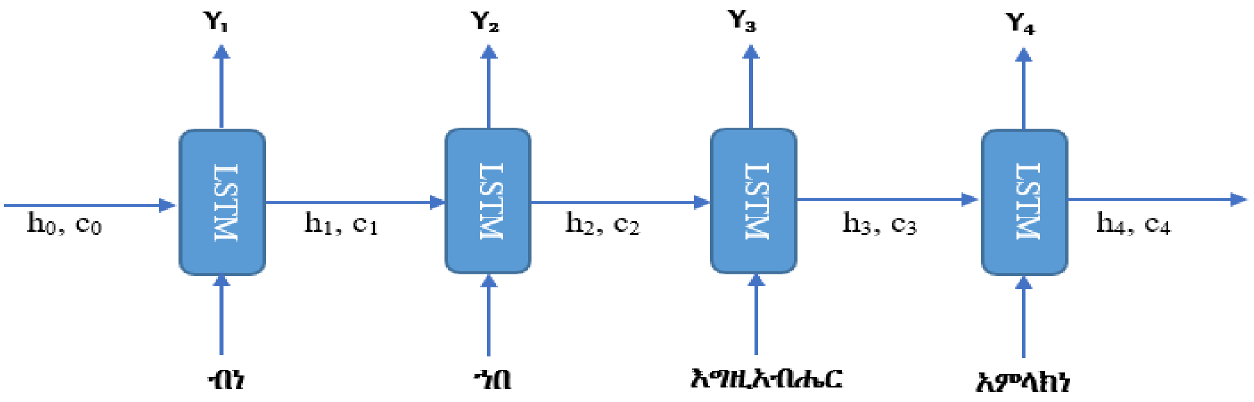

Figure 3: Architecture of encoder

The first input is a word from the sequence which we want to convert and the other two inputs are two vectors, cell state, and hidden state. Let us consider the sentence "ብነ ጎበ እግዚአብሔር አምላክነ" as our input sequence, and the encoder LSTM will process each word in the input sequence at every time step. The word "ብነ ", is the input word which is represented by x1, the input state vector i.e. h_0, c_0 are randomly initialized at the beginning. The output vector y1, the state vectors h1, and c1 are the output of the given inputs.

The state vectors, h1and c1 have the information of the previous work **"ብነ "** which is inserted at time step t_0. Then at time step t_1, the next LSTM receives the state vectors h1, c1, and the next word from the input sentence **"ብነ "** as input. And similarly, the next LSTM receives the state vectors h2, c2, and the next word from the input sentence "**ጎበ "** as input.

The third LSTM also takes the state vectors h3, c3, and the next word from the input sentence **"እግዚአብሔር "** as input. So, at the last time in step 5, the last state vectors, h4, and c4 have the information of the whole input sentence, **"ብነ ጎበ እግዚአብሔር አምላክነ",** so we don't need the vectors y1 to y5, we only need the output state vectors since these contain the information of the entire input sentence. So, there is no random input, instead, the encoder LSTM initializes the decoder with h4, and c4, these are the last vector states of the encoder. The reason behind this is the decoder should have the idea of the given input sentences to decode it.

Decoder: The decoder part of the NMT is also an RNN, which takes the contextual representation of the input sequence with the prediction of the previous hidden state and output word, and it generates the prediction of the new hidden state of decoder and output word.
The LSTM encoder and decoder of NMT has similar architecture. But they have different input and output [27].
The decoder's h0 and c0 are not random input, rather they initialized with the final hidden states of the encoder these are h4 and c4. So, here in the decoder, we have to add the symbol _START_ at the beginning of the target word and the symbol _END_ at the end of the target sentence. Finally, the last row will be "_START_ **በአምላካችን በእግዚአብሔር ዘንድ አለን** _ END_".

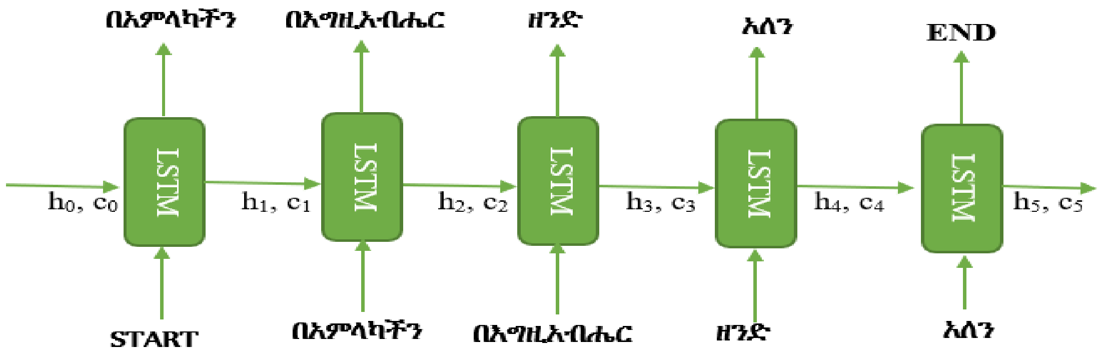

Figure 4: Architecture of decoder

Where h0 of the decoder is h4 of the encoder and c0 of the decoder is c4 of the decoder, x1 is the symbol _START_, and y1 is the first word in the target sequence. The state vectors h1 and c1 will be inserted into the LSTM decoder in the next time step. The output y1 is the essential certainty of the decoder. This task will be done until the model gets the symbol, _END_. Finally, at the last time step, we only need y's for output and we just rejected the final state vectors of the decoder. The training architecture of both the encoder and decoder is illustrated in Figure 5.

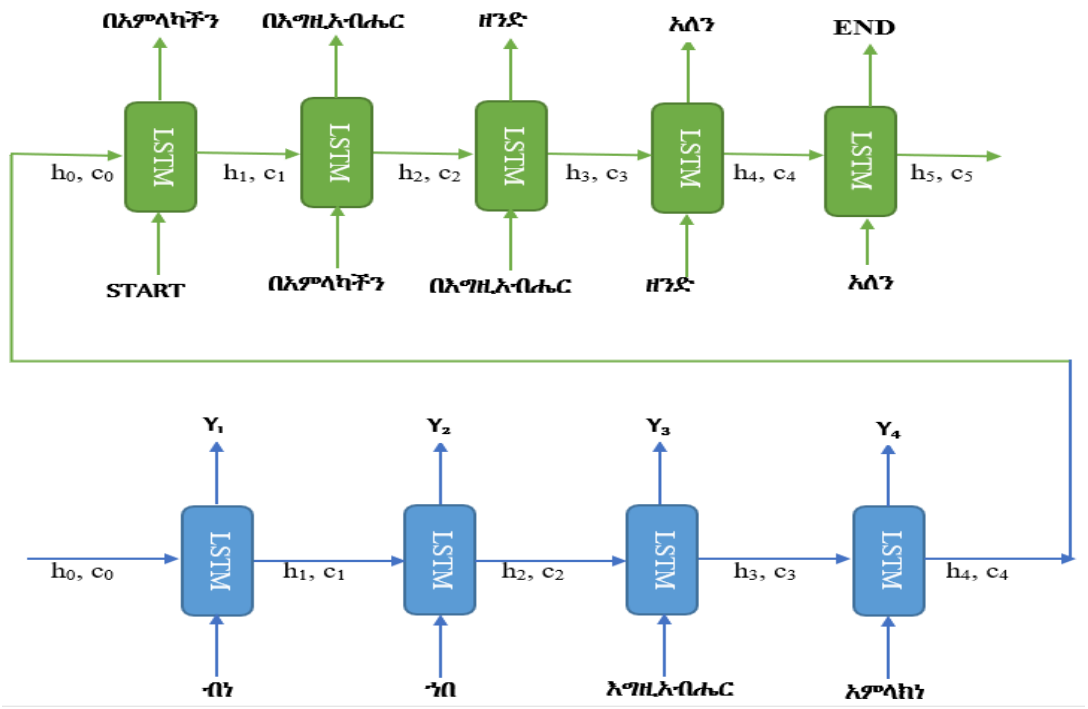

Figure 5: Encoder-Decoder architecture

Attention mechanism

The main disadvantage of the sequential encoder-decoder model for sequencing recurrent neural networks is that it only works with short sequences. The encoder model finds it difficult to remember long sentences and convert them into fixed-length vectors. Besides, the decoder receives only information that is the last hidden state of the encoder. Therefore, it is difficult for the decoder to summarize a large input sequence at once. So, to solve this problem, we have to use the attention mechanism.

Capturing semantic details of very long sentences using fixed-size vector is difficult. An effective approach that resolves this difficulty is reading the whole sentence at once and focusing on different parts of the source sentence that contain relevant information on the word being predicted. This architecture is called encoder-decoder RNN with attention. This architecture the core of Google Neural Machine Translation, or GNMT which they used in their "*google translate*" service.

Attention is probably one of the most powerful concepts in the field of deep learning today. It is based on a common sensory intuition to which we "attend to" a certain part while processing a large amount of information. The attention model can extract the most important words, even if the sentence is long and complex. Most of the state-of-the-art neural translation systems employ attention mechanism.

Generally, an encoder which computes a representation *c* for each source sentence and a decoder which generates translation one word at a time work based on conditional probability as stated in Eq. *6*. This conditional probability equation can also be written as:

$$log\, p(y|x) = \sum_{t=1}^{T} log\, p(y_t|y < t, \boldsymbol{c}) \qquad 7$$

$$p(y_t|y < t, c) = \boldsymbol{softmax}\big(g(h_t)\big) \qquad 8$$

h_t is modeled as in Eq. (*4*).
Our training model is then described as:

$$J_t = \sum\nolimits_{(x,y)\epsilon D} -log\, p(y|x) \qquad 9$$

D is parallel training corpus and *g* is a function that outputs a vocabulary size vector, *h* is RNN hidden unit, and *f in equation* 4 *computes the current hidden state given the previous hidden states. The SoftMax function normalizes the vocabulary size vector into values between 0 and 1.*

Eventually, NMT directly calculates:

$$p(y|x) = p(y_1|x)p(y_2|y_1, x)p(y_3|y_1, y_2, x) \cdots\ (y_T|y_1, \cdots, y_{T-1}, x) \qquad 10$$

There are generally two types of attention mechanisms, global and local attention. The global attention mechanism is the one in which at each translation step, all the source hidden states of used for context vector computation. Whereas the local attention mechanism is one in which only some of the hidden states of the encoder are used for context vector computation.

NMT produces results that significantly reduce the total editing performance on the best phrase-based machine translation (PBMT) system. It surpasses PBMT systems in all meaningful lengths, although production length deteriorates faster than its competitors. Additionally, its output contains fewer morphology, fewer lexical, and substantially fewer word order errors.

3. Experimental Setup

The open-source toolkit we used for our translation experiment is OpenNMT[3]. We used Colab[4] to train our model. Followings are major steps during the experiments:

[3] www.opennmt.net
[4] https://colab.research.google.com/

Cleaning the collected data: The collected data is sorted to align sentences in both languages and noisy symbols like special characters are removed. The corpus is split into space-separated words or tokens.

Tokenization: In the tokenization process, we performed two operations. N**ormalization**, which transforms the sequence to identify and protect specific characters like types of quotes, Unicode variants into unique representation to make the translation simpler, and **tokenization**, which converts the normalized sequence into space-separated token strings using byte pair encoding, dividing the corpus by root word and affix.

Preprocess the text-data: Preprocessing is the initial step in creating a machine learning model, ensuring data is organized and clean. It involves assigning tokens an index, filtering long sentences, and creating a vocabulary list. A new file is generated from the tokenized corpus for training.

Train: Our experiment utilized Google's cloud virtual machine Colaboratory (Colab), which offers free GPUs for computationally intensive tasks, to reduce computational difficulty and improve training speed.

Terms used in training

Epoch: Epoch indicates the number of steps an algorithm takes to view the entire dataset, typically split into batches for large datasets, and completes when all samples are seen. When determining the number of epochs for a model, several parameters must be considered. Increasing the number of epochs doesn't improve accuracy due to overfitting or underfitting issues. **Overfitting** occurs when the model is exposed to training data noise, while **underfitting** occurs when the model can't learn or generalize new data [28].

Validation: is a crucial process in model building, and used to select parameters and avoid overfitting. At each step of the validation, a check is performed whether the training algorithms converge or not.

Translate: The model was trained using preprocessed data and tested to translate 1k source sentences to target sentences, with the quality of the translation evaluated using BLEU.

Detokenization: When we run the detokenization process, it will be returned in the form of the actual statement.

Evaluating: The translation model's quality is assessed by comparing translated sentences with reference sentences in the target language, which are assumed to accurately represent the input test sentences. This process of comparing the sentences is performed using BLEU[5]. The overall steps of our experiments are depicted in Figure 1

[5] *BLEU* compares the n-gram of the candidate translation with n-gram of the reference translation to count the number of matches

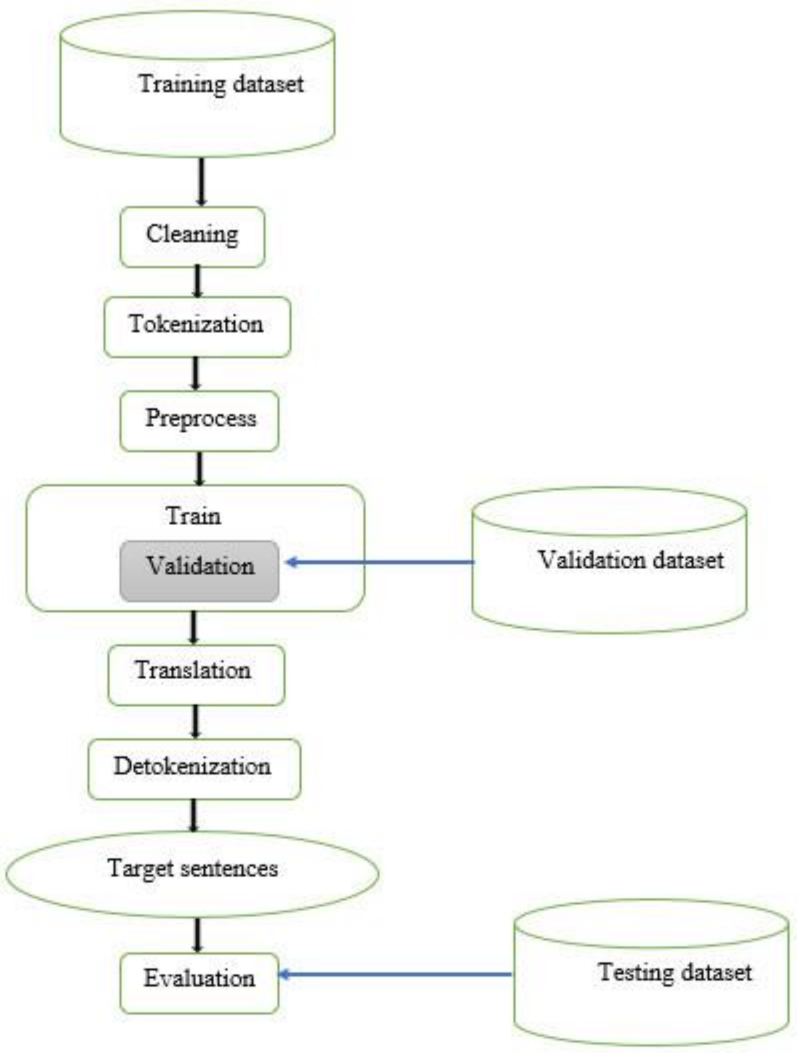

Figure 1: Experimental Steps

4. RESULTS AND DISCUSSION

As described in the previous sections 4, our model was designed to include an LSTM encoder and decoder with an attention mechanism to be able to translate longer sentences and paying attention to context.

The model was trained with four independent experiments with different proportion of the collected and different training parameters. Each of these experiments resulted in unique translation results due to the difference in the training parameters and proportion of the dataset. In the first experiment, the model was trained on 46K pair of sentences of both Geez and Amharic with the rest 2k pair of sentences for validation and the other 2k pair of sentences for testing. The number of training steps (epochs) used in this experiment is 10,000 which took up almost two and half hours to complete training the model. The score the translation result is 6.06 BLEU.

In our second experiment, we trained the model with dataset that is 1k greater than the first experiment and with the training steps doubled. The validation dataset used for this experiment is the same as the first experiment. The time it took to complete training the model is three and half hours. The BLEU of the translation result came to be as better as 7.71 as compared to the result of the first experiment.

In the third experiment, the training dataset is increased to be 48k while the validation and testing dataset are 1k each. The model is trained with 50,000 epochs which took up approximately four hours to complete training the model. This experiment achieved a translation result of 12.3 BLEU. The fourth experiment is quite similar the third experiments except the number of training epoch is doubled to be 100,000. The BLEU of the translation result is 15.4.

As it can be seen from each of the above experiments, the BLEU of the translation result improves on increasing the number of training epochs. However, this doesn't increase indefinitely. Increasing the number of training epochs indefinitely will never be a remedy for a translation result of lower BLEU. To this end, we performed one more final experiment with 120,000 epochs and the BLEU of the translation came to be lower than the previous experiment, which is a sign of overfitting.

According to our experiments, the best BLEU score of the translation result obtained for the model trained with different subsets of the 50k Ge'ez – Amharic parallel corpus in OpenNMT is 15.4%. Despite the hungry nature of NMT models for data and the costly available data for the corpus, our model has performed well with this limited amount of dataset. For comparison, the same model was trained with another English and Amharic parallel corpus which is largely and freely available and the translation result obtained exceeded our language pair translation.

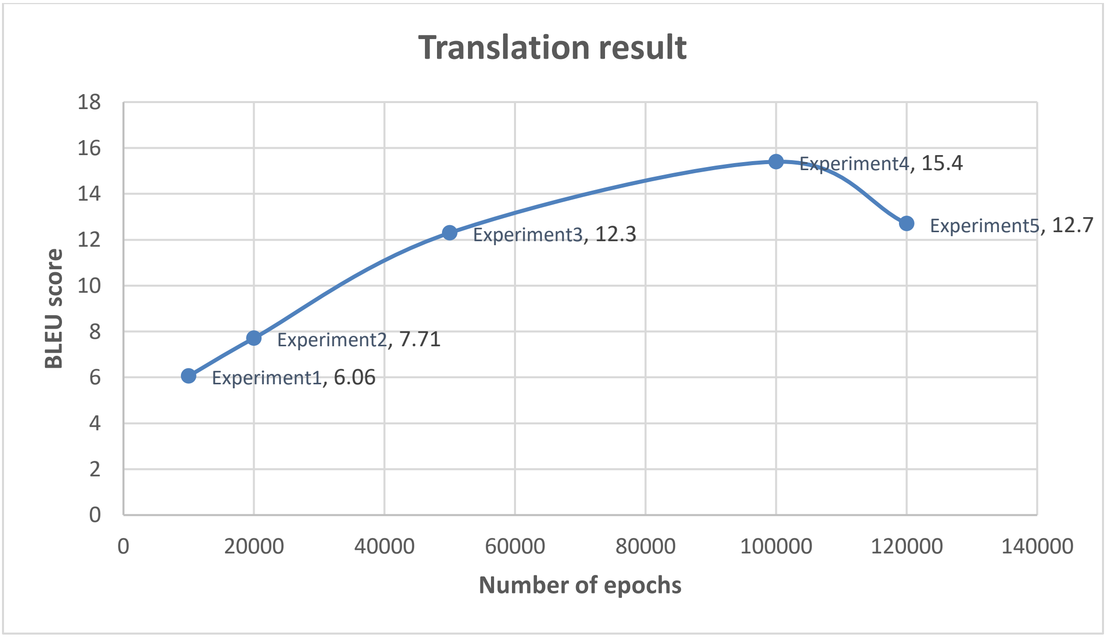

Figure 11: Translation result of experiments

Table 2 Experimental result analysis of our model using different epochs and different sets of the dataset for the training, validation, and testing.

Experiment No.	**Training epochs**	**Training dataset**	**Validation dataset**	**Testing dataset**	**BLEU score(%)**	**LSTM layers**
1	10,000	46k	2k	2k	6.06	2
2	20,000	47k	2k	1k	7.71	2
3	50,000	48k	1k	1k	12.3	2
4	100,000	48k	1k	1k	15.4	2
5	120000	48k	1k	1k	12.7	2

5. Conclusion

The main concern of this study for applying NMT with attention to translate Ge’ez documents into their corresponding Amharic meaning. Mainly, we used an encoder-decoder architecture with LSTM cells on both the encoder and decoder sides to support longer sentence translation. NMT is the state-of-the-art machine translation with better performance and less memory usage than its predecessors such as RBMT and SMT. And it needs a huge number of parallel corpora of both the source language Ge’ez and the target language Amharic. In addition to the huge amount of data it requires, wisely choosing the training parameters such as training epochs, number of LTSM layers, the size of RNN cells in the hidden layer also makes a great difference at the cost of computational resources. Due to the fact that we are able to access only one GPU, all of our experiments are performed using only two layers of LSTM with 500 hidden units. The number of training steps in training a given NMT model is also a determinant factor. Using too large number for the training step will cause overfitting that lowers the translation quality while using too small number will also result in low quality of translation. Our last experiment used three layers of LSTM with 512 hidden layers to train the model with 120,000 training steps which caused the system to total crash. So, choosing optimized values for the parameters used in training the model is required. Therefore, our model selection is from experiment 4 which has shown a better performance. Increasing the amount of standardized training datasets for both languages is our feature work.

6. References

[1] H. Wang, H. Wu, Z. He, L. Huang, and K. W. Church, “Progress in Machine Translation,” *Engineering*, vol. 18, pp. 143–153, 2021.

[2] D. H. S. P. E. C. Tom Young, "Recent Trends in Deep Learning Based Natural Language Processsing," 2018.

[3] H. D. K. S. J. Shashi Pal Singh, "Machine translation using deep learning," 2017.

[4] P. Koehn, Statistical Machine Translation, 2017.

[5] M. Irfan, "Machine Translation," October 2017.

[6] M. L. Forcada, "Making sense of neural machine translation," p. 291–309, December 2017.

[7] D. Bahdanau, "NEURAL MACHINE TRANSLATION," in *Computation and Language*, 2015.

[8] K. Gurney, An introduction to neural networks, UCL Press, 1997.

[9] Harper, K. E. (2018). Machine translation. *Soviet and East European Linguistics*, *October*, 133–142.
https://doi.org/10.4324/9781315678481 -27

[10] N. T. Sinshaw, B. E. Ejigu, B. G. Assefa, and S. K. Mohapatra, "Amharic Handwritten & Machine Printed Character Recognition Using Deep CNN with Random Search Hyperparameter Optimization Algorithm," pp. 0–18, 2023.

[11] ክፍሌ ኪዳነወልድ, መጽሐፈ ሰዋስው ወግስ ወመዝገበ ቃላት ሐዲስ. 1948.

[12] D. Asfawwesen, THE INCEPTIVE CONSTRUCTION AND ASSOCIATED TOPICS IN AMHARIC AND RELATED LANGUAGES, stocholm: Holmbergs, Malmö, 2016.

[13] T. Kassa, "Morpheme-Based Bi-Directional Ge'ez -Amharic Machine Translation," October 2018.

[14] a. L. B. Mulu Gebreegziabher Teshome, "Preliminary experiments on English to Amharic statistical machine translation," in *Third Workshop on Spoken Language Technologies for Under-resourced Languages*, Cape Town, 2012.

[15] E. Teshome, "Bidirectional English-Amharic Machine Translation, An Experiment using constrained corpus," 2013.

[16] M. M. Michael Melese Woldeyohannis, "Experimenting Statistical Machine Translation for Ethiopic Semitic Languages: The Case of Amharic-Tigrigna," July 2018.

[17] A. Gebremariam, "Amharic-to-Tigrigna Machine Translation Using Hybrid Approach," 2017.

[18] M. Hailegebreal, "A Bidirectional Tigrigna – English Statistical Machine Translation," June 2017.

[19] D. MULUGETA, "Geez to Amharic Automatic Machine Translation: A Statistical Approach," pp. 28-31, May 2015.

[20] A. A.-A. Ebtesam H. Almansor, "A Hybrid Neural Machine Translation Technique for Translating Low Resource Languages," pp. 348-355, 2018.

[21] S. R. T. W. W.-J. Z. Kishore Papineni, "BLEU: a Method for Automatic Evaluation of Machine Translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, 2002.

[22] Y. B. A. C. Ian Goodfellow, "Deep Learning," in *Deep Learning: Adaptive Computation and Machine Learning*, The MIT Press, 2016, p. 461.

[23] S. Yip, TranslateFX, [Online]. Available: www.translatefx.com. [Accessed 28 September 2020].

[24] M. A.-S. S. S. Magdi Zakaria, "Artificial Neural Network: A Brief Overview," *Mabrouka AL-Shebany et al. Int. Journal of Engineering Research and Applications* , vol. 4, no. 2, pp. 7-12, 2014.

[25] D. M. J. B. P. S. G. M. D.M. Rodvold, "Introduction to artificial neural networks for physicians: Taking the lid off the black box," *The Prostate,* vol. 46, no. 1, pp. 39-44, 2001.

[26] S. Agnihotri, ""Hyperparameter Optimization on Neural Machine Translation," pp. 4-5, 2019.

[27] P. Koehn, Neural Machine Translation, vol. 24, Cambridge University Press, 2020.

[28] D. S. R. Saahil Afaq, "Significance Of Epochs On Training A Neural Network," *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH ,* vol. 9, no. 06, pp. 485-488, 2020.