Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Amharic Parts-of-Speech Tagger using Neural Word Embeddings as Features

By: Mequanent Argaw

A Thesis Submitted to the School of Graduate Studies

in Partial Fulfillment of

Masters of Science in Computer Engineering

January, 2019

Addis Ababa, Ethiopia

Amharic Parts-of-Speech Tagger using Neural Word Embeddings as Features

By

Mequanent Argaw

MSc Thesis

Submitted in Partial Fulfillment of the Requirements

for Masters of Science in Computer Engineering

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Dr. Surafel Lemma Abebe	Menore Tekeba
Thesis Advisor	Thesis Examiner
Dr. Yalemzewud Negash	Kinde Mekuria
School Dean	Thesis Examiner

January, 2019

Declaration

This thesis is a presentation of my original research work. Wherever contributions of others are involved, every effort is made to indicate this clearly, with proper citation of sources. I, the undersigned, declare that this thesis has not been presented for a degree in any other university.

Mequanent Argaw

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

Dr. Surafel Lemma Abebe

Acknowledgement

First, I would like to thank the Almighty God for blessing and helping me in all of my life and in accomplishing this study.

The next deep gratitude shall be given for Dr. Surafel Lemma Abebe, my thesis advisor. His priceless scheduled time, strict follow up, constructive ideas and comments worth great for my current study and gave me a nice experience for the future. Without his scheduled follow up, this study will not have been finished by this time.

I would like to thank Dagmawi Demissie who was sharing me his experience on coding and understanding of concepts related to deep neural nets with word embeddings. I would also like to thank Michael Gasser, author of HornMorpho morphological analyzer, who was very responsive to answer my questions while I was using his system.

I would like to thank my parents for their invaluable price they paid for me to reach at this position. Finally, thanks to all my friends who were contributing in different aspects of my study.

Abstract

The parts-of-speech (POS) tagging for Amharic language is not matured yet to be used as one important component in other natural language processing (NLP) applications. Previous studies done on Amharic POS tagger used hand-crafted features to develop tagging models. In Amharic language, prepositions and conjunctions usually are attached with the other parts-of-speech. This forces the tags to represent more than one basic information and also decrease the total number of instances in the training corpus. In addition, the manual design of features requires longer time, more labor and linguistic background.

In this study, automatically generated neural word embeddings are used as features for the development of an Amharic POS tagger. Neural word embeddings are multi-dimensional vector representations of words. The vector representations capture syntactic and semantic information about words. Another additional aspect in this study is, prepositions and conjunctions attached with the other parts-of-speech are segmented using HornMorpho morphological analyzer. State-of-the-art deep learning algorithms are also used to develop tagging models. Long Short-Term Memory (LSTM) recurrent neural networks and their bidirectional versions (Bi-LSTM RNNs) are used to develop tagging models from the possible deep learning algorithms.

The maximum evaluation result observed is 93.67% F-measure obtained from the model developed by using Bi-LSTM recurrent neural network. From the results obtained, it can be observed that word embeddings generated by neural networks can replace manually designed features which is an important advantage. Segmenting prepositions and conjunctions attached with the other parts-of-speech also improved the accuracy of the POS tagger by more than 5%. The accuracy improvement of the POS tagger is obtained from the increased total number of instances and decreased number of tags due to segmentation.

Key words: natural language processing, POS tagger, Amharic, neural word embeddings, segmentation, deep learning, recurrent neural networks

Table of Contents

List of Figures

List of Tables

Abbreviations

ELRC	Ethiopian Languages Research Center
WIC	Walta Information Center
NLP	Natural Language Processing
POS	Parts-of-Speech
LSTM	Long Short-Term Memory
Bi-LSTM	Bidirectional Long Short-Term Memory
RNNs	Recurrent Neural Networks
GNU	GNU's Not Unix
KNN	K Nearest Neighbor
CNTK	Cognitive Toolkit
AI	Artificial Intelligence
API	Application Program Interface
CPU	Central Processing Unit
GPU	Graphical Processing Unit
TPU	Tensor Processing Unit
BSD	Berkeley Software Distribution
ARFF	Attribute Relation File Format
CSV	Comma Separated Value

1. Introduction

1.1. Background

In the world which is becoming a single village, the information and knowledge for human languages is becoming abundant. The interaction between each natural (human) language and culture is increasing as technology is advancing. The need to work on and improve natural language technology is becoming necessary than ever before. Natural language processing (NLP) is part of artificial intelligence which is the process of developing software applications that enable computers to understand human languages. NLP may be done at different levels including word level, phrase level, sentence level or semantic level (Gebre, 2010).

Clearly, computers cannot understand human languages simply as humans do so. They cannot understand the syntax of words and their semantics in a sentence. But, as the data of each natural language is being increased, it becomes difficult for us humans to analyze and get the necessary contents manually from it. We need the help of computers to manipulate the existing large amount of data. Such requirement of computers' help leads natural language processing to emerge as an exciting discipline of computer science. Natural language processing, as a field of study, has many applications including parts-of-speech (POS) tagging, named entity recognition (NER), machine translation (MT), information retrieval (IR), information extraction (IE), morphological analysis, syntactic parsing, stemming. Natural language may be presented in different formats such as audio, script and image. The focus of this study is, however, on the written form of natural language.

Parts-of-speech (POS) tagging, one of the NLP applications, is the task of identifying and assigning the appropriate lexical category to words in sentences. Some of the lexical categories are nouns, adjectives, verbs, adverbs, prepositions and conjunctions. POS tagging is a precondition or subcomponent for most of natural language processing tasks rather than being an application to stand by itself. It is mainly used in machine translation, syntactic parsing, information extraction, named entity recognition, information retrieval, spell checker and stemmer.

Developing a parts-of-speech tagger application is not a simple task due to some factors. One of the factors is, the absence of a single method which can solve the POS tagging problem completely for any language. Not only algorithms developed for computers to learn natural languages but also

trained human annotators disagree on the category of words 3-4% of the time due to inconsistency of understanding (Gebre, 2010). The other factor is the ambiguity of natural languages and the deficiency of computers to understand those ambiguities. Computers cannot understand the intended context of a word in a given sentence as humans do so easily. Tagging gives more information to computers so that they can understand human languages in a better way.

The parts-of-speech tagging problem can be solved using three general approaches: rule-based (Brill, 1992), corpus-based (Tachbelie et al., 2009) and hybrid (Mohnot et al., 2014). Rule-based POS tagging approaches use a set of manually designed features and a dictionary to tag words of sentences. Rule-based approaches have two phases of tagging words in which the first phase uses the dictionary to give all the possible tags to words. Then, the second phase puts the tagged words by the first phase on to a rule template developed by linguistic professionals so that each word will get a non-ambiguous single tag by avoiding other tags which violet the rule template. This makes rule-based POS tagging more accurate which have the ability to explain linguistic phenomena and have many and complex kinds of information (Kebede, 2009).

Developing a rule-based POS tagger for a language will be harder if the language is morphologically rich as Amharic language. Linguistic experts, which may be many in number, may require years to design hundreds or thousands of rules for developing a rule-based POS tagger for a single language. Even, after developing the tagger in such a manner, it will not be easily portable to other languages having different morphology. It needs to start developing rules from the ground.

Corpus-based parts-of-speech taggers, on the other hand, build models from the training dataset using one or more algorithms and apply the models on unseen instances of the language. Corpus-based POS tagging approaches include Hidden Markov Model (HMM), Artificial Neural Networks (ANN), Memory Based Taggers (MBT) and Decision Tree (DT) taggers (Kebede, 2009). Hybrid approaches combined the positive aspects rule-based and corpus-based approaches (Mohnot et al., 2014).

1.2. Problem Statement

The driving factor into this study comes from the advantage of POS tagging for many NLP applications. Even if it is one of the basic components in different NLP applications, POS tagging

in Amharic language has not yet matured. Thus, further researches shall be conducted to have a better POS tagging component for the other NLP applications of Amharic.

Previous researches on Amharic POS tagger were done by varying either the size of the corpus (Gebre, 2010) or the algorithms used to develop classifying models (Gamback et al., 2009). The hand-crafted features used in those studies got improved from one study to the next. For example, (Gebre, 2010) added two features on those used in former studies. But there are also some additional issues which are suggested to improve the Amharic POS tagger. One of the issues is related to segmentation of the attached prepositions and conjunctions from the other parts-of-speech. In Amharic, prepositions and conjunctions are frequently attached with the other parts-of-speech. Segmenting them reduces the number of tagset members and increases the total number of instances of the dataset which may help to improve accuracy of the tagger. By applying segmentation, combinational tags such as NP (Noun with Preposition), NC (Noun with Conjunction), NPC (Noun with Preposition and Conjunction) and the like will be avoided and the basic single tags will be used. The effect of segmenting prepositions and conjunctions on Amharic POS taggers is not studied in previous researches. In this study, it is aimed to investigate the effect of segmenting prepositions and conjunctions on Amharic POS taggers.

The other issue is the design and selection of features used to build tagging models. Previous researches such as (Adafre, 2005) and (Gebre, 2010) used to design the required features manually which requires a careful understanding and observation of the morphology of the language. Amharic is one of the under-resourced languages having rich and complex morphology. To consider the important features for POS tagging, it requires time and dedicated linguistic experts. Some important features may also be missed in the manual design of features. It will be better if there is a mechanism to learn the features automatically using either machine learning techniques or any other approach.

The third issue is related to the performance of state-of-the-art algorithms on Amharic POS tagging. Long short-term memory (LSTM) recurrent neural networks and bidirectional long short-term memory recurrent neural networks showed better results (e.g. 97% accuracy for Portuguese (Fonseca et al., 2015)). The state-of-the-art POS tagging approaches for Amharic use machine learning algorithms such as support vector machines (SVM), conditional random fields (CRF) and hidden Markov model (HMM) (Gebre, 2010). However, the maximum performance obtained from

these algorithms is 90.95% accuracy for CRF. In this study, the impact of using LSTM recurrent neural network on Amharic POS tagging is investigated.

This study will try to answer the following research questions in relation with the performance improvement of the Amharic POS tagger.

- RQ1: [***Neural word embeddings for Amharic POS tagging***]:
  Could neural word embeddings improve the performance of Amharic POS tagging?
- RQ2: [***Segmentation for Amharic POS tagging***]:
  Could segmentation of the attached prepositions and conjunctions from the other parts-of-speech help to improve the Amharic POS tagger?

1.3. Objectives

1.3.1. General Objective

The general objective of this study is to examine the impact of segmentation and neural word embedding features on Amharic POS tagger.

1.3.2. Specific Objectives

- To have a cleaned corpus presented in Amharic Fidels for segmentation and word vector generation.
- To increase the total number of instances and decrease the number of tags by segmentation.
- To have automatically generated features as a replacement of hand-crafted features.
- To examine the performance of deep neural nets on Amharic POS tagging using neural word embeddings as features.

1.4. Methodology

The methodology followed to address the problems mentioned above includes the following tasks.

Literature review: done on the researches studied previously on Amharic POS tagging and other related papers to understand the necessary concepts for the study.

Data collection: the annotated corpus is collected from a previous researcher (Demissie, 2017).

Preprocessing: includes cleaning the corpus and the segmentation task.

Word vector generation: is done to represent the syntactic and semantic features of words into limited dimensional real valued vectors.

Feature extraction: component is implemented appropriately according to the existing features.

Model building: is implemented using the existing classifiers on Weka and the new ones that are developed based on deep learning.

Model evaluating: is done to measure the performance of the developed models using precision, recall, F-measure and accuracy.

1.5. Scope of the Study

This study is done on Amharic POS tagging using automatically generated neural word embeddings as features. Despite the possibility of poem, Biblical and other sentence arrangements, a corpus prepared from news articles is only used.

1.6. Contributions

This study has the following contributions.

- It increases the total number of instances in the dataset by adding new prepositions and conjunctions.
- It presents the performance of neural word embeddings in Amharic POS tagging.
- It presents a simpler feature extractor which searches for corresponding features of words from the storage of words along with their tags and features.
- It presents the performance of LSTM and Bi-LSTM deep neural networks on Amharic POS tagging.

1.7. Thesis Organization

The remaining part of the Thesis is organized as follows. Chapter 2 presents a theoretical background about POS tagging and related issues including the possible approaches of its development. Chapter 3 presents the summary of the related previous works. Chapter 4 explains the proposed approach of this study. Chapter 5 presents the experiments done and the results obtained. Finally, Chapter 6 presents the conclusion and possible future works of the study.

2. Theoretical Background

2.1. Parts-of-Speech (POS) Tagging

Parts-of-speech tagging (POS tagging) is a task of assigning the parts-of-speech category or tag to each word of a given text. The parts-of-speech categories include noun, pronoun, adjective, verb, adverb, preposition, conjunction etc. These categories are represented with their short tags. The tags can be done to different details even in a single language including person of pronouns, plurality of nouns and tense of verbs. For example, previous researches done on Amharic POS tagging used different tagsets. As the number of member tags in the tagset increases, the POS tag will give a better information about the word. The following can be an example of POS tagging in Amharic language.

በላይ <N> ዘለቀ <N> ጀግና <ADJ> የኢትዮጵያ <N> አርበኛ <N> ነበር <AUX> ። <PUNC> ➔ *Belay Zeleke was a brave Ethiopian patriot.*

Words of the above sentence are tagged with their appropriate parts-of-speech categories: N, ADJ, and AUX. The short forms represent noun, adjective and auxiliary verb, respectively. Tags are thus short symbolic representations of word categories. A collection of tags representing all classes of words having distinct grammatical behavior is known as a tagset (Gebre, 2010).

Tags give many linguistic information about words and they need to be designed carefully. The details of designing the tags vary according to the details of the information to be obtained from them. In addition to the common parts-of-speech, tags may be designed to give more information about the gender and number of nouns, tenses of verbs, singular and plural forms and other required details. So, the number of tags in most of the languages depends on the purpose of the application being developed and the nature of the language. Even, the design of tags by itself can be a research problem which needs a careful analysis of the nature of the language (Gebre, 2010).

2.2. Applications of POS Tagging

POS tagging is an important subcomponent for the following natural language processing applications.

Machine Translation

Machine Translation (MT) is an application used to translate a text in one language to its equivalent meaning in another language. It is one of the widely used and very important natural language processing applications in today's world of technology. Machine translation uses a bilingual dataset and other resources to do the translation task.

Parts-of-speech tagging is the first essential part in preprocessing for machine translation. POS tagging is used to determine the grammatical structure of the concerned languages. For example, a subject-verb-object (SVO) sentence structure of English should be changed to an SOV sentence structure of Amharic to be meaningful if the translation is from English to Amharic (Zewgneh, 2017). To determine these sentence structures, POS tagging is required.

Information Extraction

Information Extraction (IE) is another natural language processing application which is the task of extracting structured facts from unstructured dataset automatically. It involves some steps including identifying a predefined set of concepts and deciding whether a text is relevant for a certain domain. It is indicated in (Hirpassa, 2017) that information extraction involves six different main tasks. These tasks are: POS tagging, named entity recognition, syntax analysis, co-references and discourse analysis, extraction patterns and bootstrapping.

Named Entity Recognition

Named Entity Recognition (NER) is the task of classifying proper nouns in a given text into their corresponding predefined categories. The categories of proper nouns include person, location, organization, date, and others (Demissie, 2017). POS tagging is a prerequisite for named entity recognition to extract nouns from a given annotated corpus. Proper nouns then can be further refined from the collection of nouns to be used in the NER application development.

Syntactic Parsing:

Parsing is the process of converting a given flat sentence into a hierarchical structure that corresponds to the units of meaning in the sentence. It is a procedure of finding ways on how to combine grammatical rules that can generate a tree representing the structure of the input sentence. Having a set of grammatical rules is one of the requirements of a functional syntactic parser. Then,

the set of grammatical rules enabled the system to classify words into their corresponding grammatical categories or parts-of-speech (Megersa, 2002).

2.3. Overview of the Amharic Language

Amharic is the official working language of the Federal State of Ethiopia having its own script originated from Ge'ez, the language being used in Ethiopian Orthodox Tewahido Church for spiritual services. Amharic is also the working language of some regional states such as Southern Nations Nationalities and Peoples, Gambella and Benshanguel Gumuz.

Amharic is one of the Semitic languages which belongs to the Afro-Asiatic super family. It is a morphologically rich language and related to Hebrew and Arabic since they belong to the same family. As other Semitic languages, Amharic also has root words made up of consonants with their own lexical meaning and other words can be formed by inserting vowels among the consonants. In Amharic language, nouns, adjectives and adverbs can be derived from other nouns, adverbs, adjectives and verbs. The unusual situation is the possibility of finding derived verbs from the other parts-of-speech (Gebre, 2010). Different forms of verbs are derived from their own root verbs.

In Amharic, there are 34 basic consonant and vowel characters which have six additional different forms. The seven forms are based on the seven vowels (ኧ, ኡ, ኢ, ኣ, ኤ, እ, ኦ). The 34 basic characters are ሀ, ለ, ሐ, መ, ሠ, ረ, ሰ, ሸ, ቀ, በ, ተ, ቸ, ኀ, ነ, ኘ, አ, ከ, ኸ, ወ, ዐ, ዘ, ዠ, የ, ደ, ጀ, ገ, ጠ, ጨ, ጰ, ጸ, ፀ, ፈ, ፐ. These characters have the first vowel in their form implicitly. When one of the remaining six vowels follow these basic characters, they change their shape into some predefined forms.

For example, the second vowel forces the consonants to add a horizontal like shape in their right side and at their medium height (e.g. ሁ, ሉ, ሑ, ሙ for ሀ, ለ, ሐ, መ). The third vowel, on the other hand, forces the addition of the previous shape at the right bottom, sometimes with some additional lengthening vertical lines (e.g. ሂ, ሊ, ሒ, ሚ for ሀ, ለ, ሐ, መ). But there are exceptional characters which are not suitable to add such shapes. The full form of the characters can be observed from the set of Amharic Fidel Gebeta presented in appendices A and B.

The seven base characters ሸ (Se), ቨ (ve), ቸ (ce), ኘ (Ne), ዠ (Ze), ጀ (je) and ጨ (Ce) are found in the Amharic alphabet only. No word in Ge'ez language consist of these letters while they are

frequently used alphabets of Amharic language. There are also some special characters which represent the special labial sounds in addition to the 238 characters. These special characters of Amharic include ሏ (lua), ሟ (mua), ሯ (rua), ሷ (sua), ሿ (Sua) …. These characters are also not used in Ge'ez language. This shows that Amharic added its own derived characters from Ge'ez which by themselves are not members of the Ge'ez alphabet.

All the above 34 basic characters are not unique in their pronunciation or sound representation. Some of them represent similar sounds, even if all of them have their own usage. The sets {ሀ, ሐ, ኀ}, {ሠ, ሰ}, {አ, ዐ} and {ጸ, ፀ} represent similar sounds. The trend taken from Ge'ez in using these similar letters can make the case easier to understand. Writing words such as ነገሰ, ንጉሰ and መንግሰት[1] this way is out of convention. Rather ነገሠ, ንጉሥ and መንግሥት are acceptable since the character ሠ and its other six forms are expected to be used in words derived from the root word 'neggese' (ነገሠ). The ሠ is referred to as '*nigusu se*' (ንጉሡ ሠ) to differentiate it from the other character (ሰ), which represents the same sound. The character ሰ, on the other hand, is known as '*esatu se*' (እሳቱ ሰ) which is used to write words such as እሳት, እሳቱ. To write other words using these characters, it needs to know such conventions and the task of creating awareness on such issues may be for linguistic experts. It will be better to understand such issues on characters representing similar sounds rather than trying to have only one character for one sound and vanish the other characters.

2.4. Challenges in Amharic POS Tagging

There are some challenges in Amharic POS tagging which does not exist in the other languages. For example, English has capitalization for proper nouns which helps to identify them in classification by considering the case as one important feature. The case is different in Amharic language which does not have such special features for nouns rather they are written in a similar manner as other parts-of-speech words.

Another challenge is, the morphological features in Amharic are more complex. Even, some parts-of-speech are attached with the other parts-of-speech. Prepositions and conjunctions are usually attached with the other parts-of-speech as shown by the following example.

E.g. *To Abebe ➔ ለአበበ (le-Abebe), Abebe and Kebede ➔አበበና ከቤዴ (Abebe-na Kebede)*

[1] The horizontal lines are underlines for giving emphasis and are not part of the characters.

Thus, prepositions mostly come as prefixes of the other parts-of-speech if they didn't stand by themselves while conjunctions are mostly attached as suffixes of the other parts-of-speech. The plural forms, person forms etc., on the other hand, do not have a simple pattern which makes the design of morphological features more complex. For example, the plural form of a noun ሰው (person) is ሰዎች (persons) while the plural form of another noun መንግሥት (state) is መንግሥታት (states).

Another challenge in Amharic is that there are words which have different meanings and act as different parts-of-speech according to the context in the given sentence. For example, a single word በለጠ (*belete*) may act as a noun and a verb according to its context. It can be a name of a person if it is used as a subject or as an object in the sentence. The same word can also be used as a verb to mean *it/ he became greater or more than some other thing.*

2.5. POS Tagging Approaches

Different approaches have been applied so far to develop parts-of-speech taggers. The most common ones are: rule-based, corpus-based and hybrid approaches. The corpus-based approach includes statistical and artificial neural network approaches. In rule-based POS tagging approach, different grammatical and morphological features are considered to determine the words' categories. In the stochastic or statistical approach, probabilistic models are applied to assign the most likely tags to words of sentences. Artificial Neural Networks, on the other hand, use the training corpus and adaptively learn properties of words to pick the appropriate tag for a word in a given sentence. Hybrid approaches combine the strengths of either rule-based and statistical approaches or rule-based and artificial neural network-based approaches. More discussion on the possible approaches of developing a POS tagger is presented as follows.

2.5.1. Rule-based POS Tagging Approach

Rule-based parts-of-speech tagging approach uses a set of rules containing morphological, syntactical and lexical information. The set of rules developed for POS tagging of one language are highly dependent on the specific language and cannot be easily applied to the same purpose on another language. Rules to be used in the tagging process are either manually hand-crafted by linguistic professionals or automatically generated using machine learning algorithms (Abreha, 2010).

The manual approach is time taking, tedious, and error prone approach. The quality of the rules also depends on the skills of linguistic professionals. The second alternative of getting rules is automatic which learns and stores a set of rules from its input corpus without any linguistic professional requirement. The automatic way of generating rules is known as Brill Transformation based approach (Mekuria, 2013). In addition to machine learned and hand-crafted set of rules, rule-based approach can use contextual information rules, usually known as context frame rules.

Rule-based approach has its own advantages and disadvantages. One of the advantages is, it is not dependent on the amount of the data in hand which enables it to work well on small amount of data. It also gives better performance when applied on POS taggers developed for limited domains. Besides, it can be used with both well-prepared and ill-prepared input formats. The disadvantages of rule-based approach include: inflexibility of adding and removing a rule from the set of rules defined, time taking, tedious and linguistic professional requirement.

2.5.2. Stochastic POS Tagging Approach

Stochastic parts-of-speech tagging approach is mostly applied since it is simpler than rule-based tagger. It uses probabilistic model to calculate the most appropriate tag based on the word and its neighbor tag and choose the most suitable tag to a word. Stochastic POS tagging can be either supervised or unsupervised based on the corpus used. It will be supervised if the corpus it uses for training is annotated (Gamback et al., 2009) and unsupervised if it uses unannotated corpus (Teichert and Duame, n.d.) to learn information about tagset. Stochastic parts-of-speech tagger can be done based on methods such as maximum likelihood, N-gram, and Hidden Markov Model (HMM) (Mekuria, 2013).

Maximum-Likelihood (Most Frequent Tag) Method

Maximum likelihood method is the simplest one in stochastic approach which assigns the most frequent parts-of-speech tag found in the training data to a text in a non-tagged data. The probability of a tag to be assigned for a word can be calculated by counting every word with a specific tag and dividing it with the number of occurrences for this particular tag (Mekuria, 2013). This gives the conditional probability of the word given the tag. The major problem of maximum likelihood method is that it does not consider the context of a word in a sentence when assigning the most appropriate tag (Mamo, 2009).

N-gram Method

The N-gram method uses the frequency of parts-of-speech tag, word or both word and corresponding POS tag sequences to produce the probability of a word by considering contextual information within a given sentence. It can calculate the probability of tag sequence given the previous n-1 tags and it can also calculate the probability of the word sequence and the probability of tag and word sequence (Mekuria, 2013).

HMM Method

Hidden Markov Model is widely used stochastic approach for POS tagging. Hidden Markov Model is characterized by finite set of states, set of observations, set of state transition probabilities and the initial state. The finite set of states are associated with a probability distribution and the transitions among the states are determined by transition probabilities. In particular state, a visible observation can be generated based on the associated probability distribution while the states are hidden to the observer (Abreha, 2010).

The basic idea of HMM is that it is possible to choose the highest priority tag for a word given a sequence of words with their own potential tag sequences. Words can have one or more tags and a sequence of tags with the highest probability is chosen. In this process, the sequence of words is directly visible to the observer and the sequence of tags is hidden since it is only estimated. The term 'Hidden Markov Model' is derived from the hidden state, the tag sequence for POS tagging case. The probability of a tag sequence is a function of the probability of a tag following another tag and the probability of a word being assigned with a particular tag from the possible tags (Mamo, 2009). Thus, HMM can be defined as:

$$P(\text{tag}/\text{ previous n tags}) * P(\text{word}/\text{ tag}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

This becomes the combination of the n-gram tag and the most frequent tag.

To generalize the stochastic approach, it does not require linguistic professionals and requires large amount of annotated training data.

2.5.3. Artificial Neural Network (ANN) Approach

Artificial Neural Networks are computer programs which are designed to simulate the biological human brain information processing. The power of a neural network is determined by its component processing elements (neurons) which have weighted inputs, transfer function and an output. The behavior of a neural network, on the other hand, is determined by the transfer functions of its component neurons, the learning rule and the architecture of the network itself. The activation signals of the neurons in the network are the weighted sum of inputs which then passed through the transfer function to produce an output for each neuron (Agatonovic-Kustrin and Beresford, 1999).

ANNs learn from experience of patterns and relationships obtained in the input data and not from programming. Hence, it is possible to call the ANN learning process as an adaptive learning since the network is able to find properties from the presented data. The network does not require to be told how to react to each data input separately like conventional programming (Abreha, 2010).

Most of artificial neural networks have input layer, hidden layer and output layer. The input layer is the first layer which takes the original information as an input of the network. The hidden layer is the layer between the input and output layers and the output layer represents the output of the neural network. The activities of both hidden and output layers depend on the activities of their previous layer neurons and the connection weights with those neurons (Mekuria, 2013).

To use artificial neural networks for parts-of-speech tagging, it needs to preprocess the input data. The preprocessing may include tokenizing, feature generation, and feature extraction etc. By such preprocessing tasks, the data is prepared in a format required by the neural network and then given to it as an input.

Artificial neural networks have their own varieties including the common neural networks of input layer, hidden layer and output layer. ANNs also include different deep neural networks having more than one hidden layer. Deep neural network (deep learning) is about learning multiple levels of representation and abstraction that help to make sense of data such as images, sound, and text.

2.5.4. Hybrid Approach

Hybrid approach is the combination of two approaches by taking positive aspects of both. The positive aspects of stochastic approach may be combined with the positive aspects of rule-based approach to have a better POS tagger. The positive aspects of artificial neural network approach and rule-based approach can also be combined to have a better performing POS tagger. Such combinations of two approaches are thus known as hybrid approaches.

2.6. Neural Word Embeddings

The word representations determined by using neural networks are very interesting because the learned vectors explicitly encode many linguistic regularities and patterns about words. Neural word embeddings are the possible methods of representing words so that words having similar meanings can be represented similarly. Neural word embeddings are distributed representations of a text which enable deep learning methods to perform well on the challenging natural language processing problems (Brownlee, 2017). One of the basic advantages of neural word embeddings is the reduction of out-of-vocabulary impact. This is possible because words will not be completely unknown as far as they have feature vectors even if they may not be seen in the training dataset (Fonseca et al., 2015).

Words can be encoded into limited vector spaces or neural word embeddings in one of the two methods, the continuous bag of words (CBOW) method and the skip-gram method (Word2Vec in Deeplearning4j, n.d.). Continuous bag of words takes the average of the possible contexts of a word in representing it in a limited dimensional vector space. For example, ‘Apple’ can refer to either the name of the company or the type of fruit. Continuous bag of words places the vector of the word ‘Apple’ to a medium position of the two contexts.

Skip-gram model was introduced by (Mikolov et al., 2013) which is an efficient method for learning high quality vector representations of words from large amounts of unstructured text data. The authors of Skip-gram model stated as it is efficient and more accurate than CBOW in that it does not involve dense matrix multiplications. Skip-gram method, on the other hand, can assign two vector values for the above two contexts of ‘Apple’ (NSS, 2017). Based on the above preferences presented by (Mikolov et al., 2013) and (NSS, 2017), Skip-gram model is used in this study.

Continuous bag of words (CBOW) and Skip-gram models are represented more clearly in Figure 2.1 and Figure 2.2, respectively.

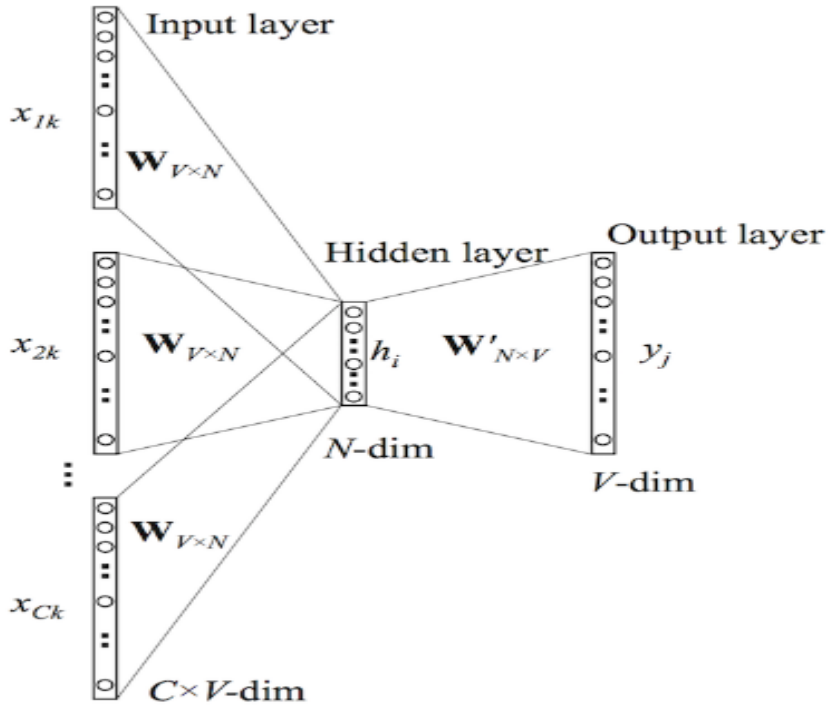

Figure 2-1: Continuous Bag of Words (CBOW) model (picture credit:(Olejnik, 2017))

Notations: x is the one-hot encoded vector of a given word, W is the weight matrix between layers of given dimensions, h is the hidden layer vector and y is the output vector.

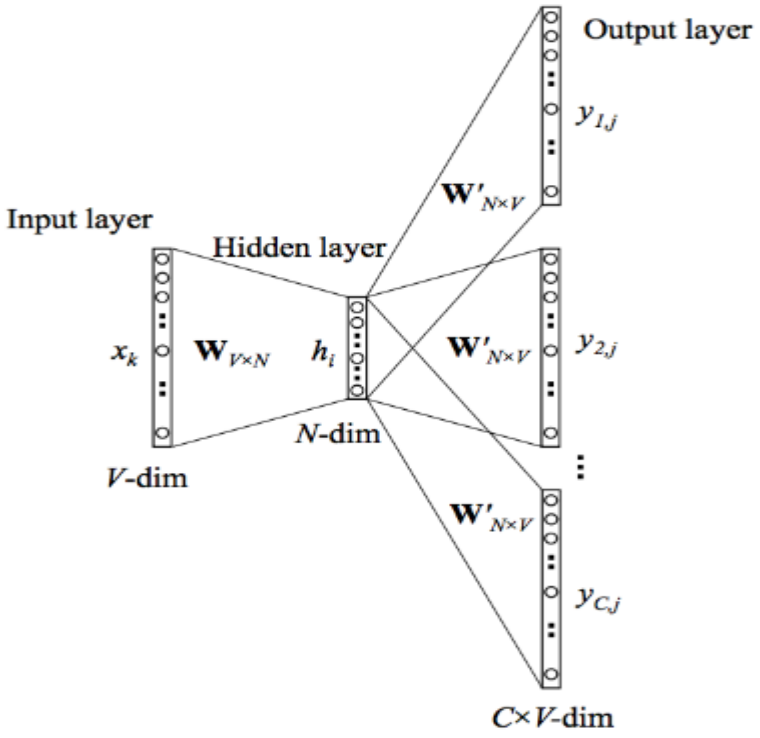

Figure 2-2: Skip-gram model (picture credit: (Olejnik, 2017))

2.7. Deep Learning

Deep learning, usually known as hierarchical learning, is a set of machine learning algorithms suitable to learn layered model inputs. It can learn representations of multiple level abstraction of data through its multiple processing layer components. The algorithms in deep learning use the back propagation algorithm and can discover complex structures from large datasets. Deep neural network architectures including recurrent neural networks, convolutional neural networks and deep belief networks. The architectures are used in different areas including natural language processing applications and they are giving encouraging results (Demissie, 2017). From the possible architectures of deep neural networks, recurrent neural networks having long short-term memories are used in this study. Better performance (97.40% accuracy) is observed when they are applied on English with word embeddings as features (Wang et al., 2015). Here they are tested with Amharic.

2.7.1. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are best suit artificial neural networks for sequence labeling applications. A sequence of vectors is given for the recurrent neural network and returns another sequence. Its preference for using it in sequence labeling comes from its ability to connect the previous information with the present task. A recurrent neural network can be understood as repeated copies of the same network put in sequence so that each network passes a message to its successor (Olah, 2015). This is demonstrated using Figure 2.3.

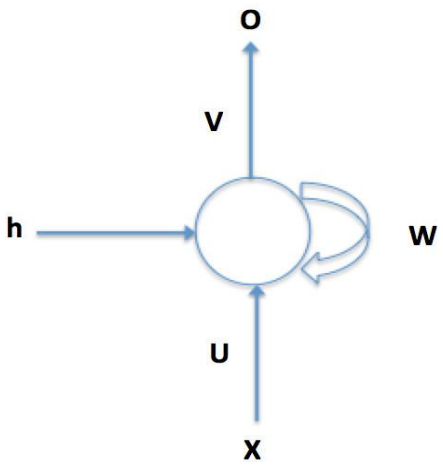

Figure 2-3: Typical representation of RNN (picture credit: (Tripathi, 2017)).

The loop is signifying that the output is again fed back to the same unit. A recurrent neural network when unrolled looks the below Figure 2.4.

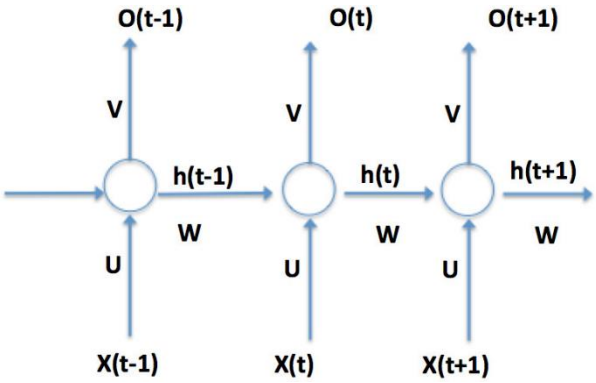

Figure 2-4: Unrolled representation of RNN (picture credit: (Tripathi, 2017)).

The notations U, V and W represent the weight matrix for each time step. t is the time and X, h and O, on the other hand, represent the input layer, hidden layer and output layer vectors.

2.7.2. Long Short-Term Memory Networks

Even if recurrent neural networks are theoretically absolute in handling long-term dependencies, practically they do not seem to perform to such extent. Long short-term memory networks (LSTMs), special kinds of recurrent neural networks, are better than the standard RNNs to handle long-term dependencies in sequence labeling applications. LSTMs do not struggle to learn the long-term dependencies, rather their practical default behavior is designed to remember information for a long time. This behavior of LSTMs solves the long-term dependency problem in a better way. Long short-term memory networks have the same chain like structure as recurrent neural networks, with a different structure repeating module in LSTMs. The repeating module of LSTMs has four layers interacting in a very special way unlike that of the standard recurrent neural networks (Olah, 2015).

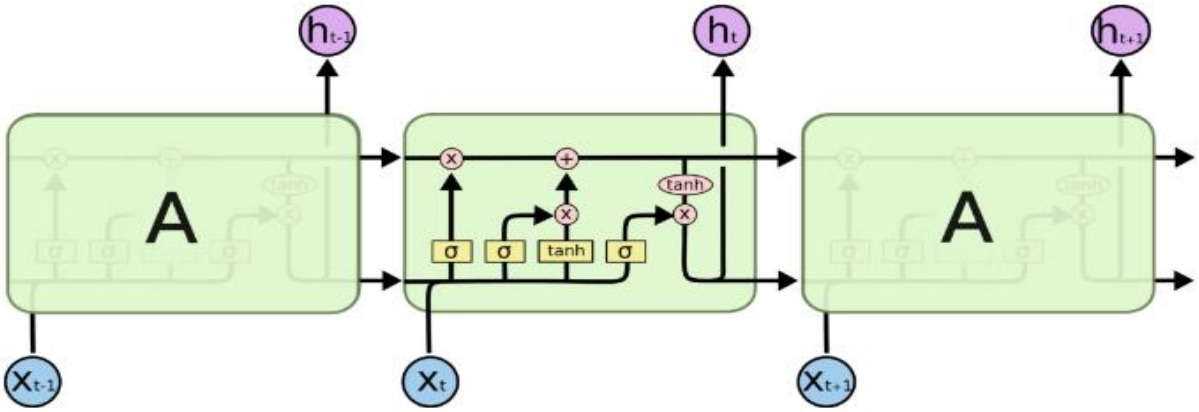

Figure 2-5: The repeating module in LSTM. (credit: (Olah, 2015))

As shown in Figure 2.5, the top horizontal line in the repeating module of LSTM is its key part which is called a cell state. The cell state acts as a conveyer belt and continues through the entire chain with small linear interactions. Information can be let through or denied from passing through the cell state of LSTMs by using gates made from a sigmoid network layer and a pointwise multiplication operation. Gates carefully regulate information to either let through the cell state or reject it. The sigmoid layer of gates gives an output having a value of either zero or one, where zero prohibits the flow of information through the cell state and one lets every information to pass through the cell state.

2.7.3. Bidirectional Long Short-Term Memory Networks

Both LSTM and RNN can get information of the previous context only. But sometimes, information about the future context may also be necessary for the current task. Bidirectional recurrent neural networks (Bi-RNNs) are designed to remember and handle both the previous and the future information required for the current task. When the repeating module of the Bi-RNNs is made of LSTMs, the resulting neural network becomes a bidirectional long short-term memory recurrent neural network (Bi-LSTM RNN). In Bi-LSTM RNN, the LSTM is used as a storage and the Bi-RNN is used to access information from the past and the future. This helps the Bi-LSTM to have the advantage of LSTM with feedback for the next layer (Yulita and Fanany, 2017). Bi-LSTM RNN can be demonstrated using Figure 2.6 below.

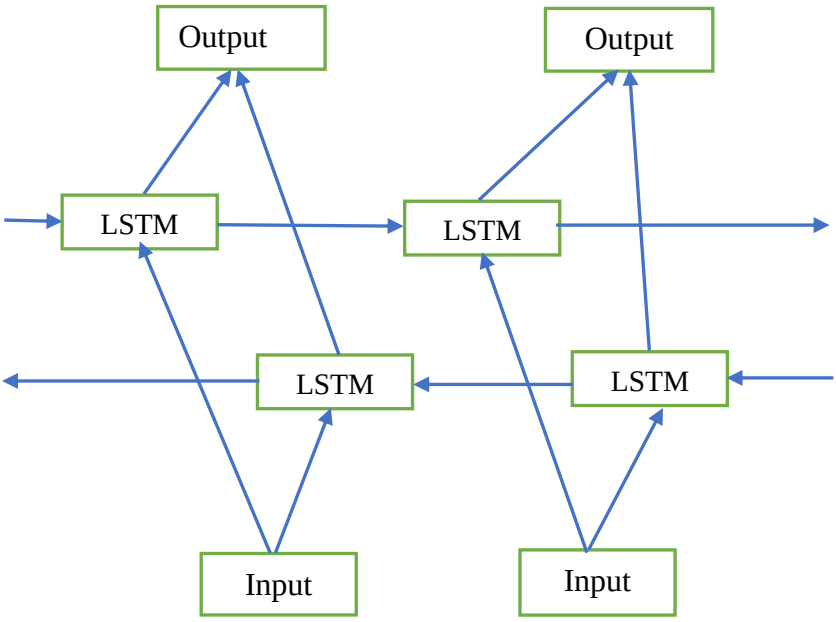

Figure 2-6: The Bi-LSTM RNN model

3. Literature Review

3.1. Parts-of-Speech Tagging on Amharic Language

A limited number of studies have been conducted on parts-of-speech tagger for Amharic. This study is a continuation of the previous studies with some proposed additional features.

Adafre (Adafre**,** 2005) worked on Amharic POS tagging using Conditional Random Fields (CRF). The author used 1000 words from five news articles by manually annotating the words with their respective part-of-speech tags and applied hand-crafted features. Adafre used features composed of character features, morphological features, dictionary features, the previous tag and character binary features. By using these features, the author obtained an overall accuracy of 74%.

Gamback et al. (Gamback et al., 2009) studied on methods for Amharic POS Tagging using the Ethiopian Languages Research Center (ELRC) corpus with manually designed features. Some of the features included in the authors design are:

- tag of $Word_n$
- prefixes of $Word_n$
- postfixes of $Word_n$
- is $Word_n$ capitalized?
- is $Word_n$ all digit?

The authors applied three state-of-the-art POS taggers to Amharic using three different tagsets. The taggers they used are HMM based TnT (Trigram'n'Tags), SVM based SVMTool and Maximum Entropy based MALLET. The three tagsets used are: the original 30-tagset developed by the ELRC, 11 basic tags the authors mapped to the corpus, and the 10 tags used by Adafre (Adafre, 2005) (referred as SISAY). SVM resulted in a relatively better performance from the three taggers. The accuracies achieved by SVM for the three tagsets are 88.30%, 92.77% and 92.80% for the ELRC, 11-basic and SISAY tagsets respectively. These results can indicate the effect of the number of tags on the accuracy of the POS tagger.

Another researcher (Kebede, 2009) also worked on Amharic POS tagging using decision tree classifiers. The author used 19,634 words selected from the ELRC corpus for manual processing of the dataset. The rules developed by decision tree classifiers are stated as they are human

understandable and subject to further improvements unlike that of the other statistical taggers. This implies that the rules generated by decision tree classifiers can be checked and evaluated linguistically to improve the accuracy of the classifiers. The maximum accuracy obtained in the study is 84.9% using J-48 decision tree algorithm (Weka's implementation of C4.5 algorithm).

Tachbelie et al. (Tachbelie et al., 2009) developed Amharic POS tagger for Factored Language Modeling. The dataset was the ELRC corpus and the taggers used were HMM and SVM on the respective tools of TnT and SVMTool. The overall accuracies of the taggers are presented as 82.99% and 85.50%, respectively. The main aim in the paper was to determine how the addition of POS information affects the quality of a language model to be used in different NLP applications. They developed language models, based on their POS tagger, for Amharic speech recognition.

Gebre **(**Gebre**,** 2010), performed POS tagging for Amharic on the ELRC corpus by taking possible features from previous researchers and adding his own features. The author used some important features based on the nature of the language. The features used include morphological features, syntactic features and semantic features. In the morphological features, the author considered suffixes and prefixes that will be added for each parts-of-speech. Some affixes used in one POS tag will not be applied for others and such cases help in the classification process. Totally, the author used 11 basic morphological, syntactic and semantic features. Of the 11 features, vowel patterns and radicals are stated as the novel features set by the study. The author used four state-of-the-art machine learning algorithms (CRF, SVM, TnT, Brill). The maximum accuracy was obtained from the CRF based classifier which is 90.95%.

Lately, (Tachbelie et al., 2011) developed POS tagging for Amharic to identify the best method for under-resourced and morphologically rich languages. They used the tagsets of ELRC to derive a new POS tagset having 11 basic classes. The algorithms used include CRF, SVM, MBT and TnT. They observed that Memory Based Tagging (MBT) is a good strategy for under-resourced languages as its accuracy is less affected by the amount of the training data as compared to the other methods. The HMM based TnT, on the other hand, was demonstrated as it is sensitive to the amount of the training data. It performed worse on a small amount of data and its performance is improved when the size of the dataset is increased.

3.2. Parts-of-Speech Tagging on Foreign Languages

Fonseca et al. (Fonseca et al., 2015) developed a POS tagger for Portuguese language for the purpose of evaluating a revised Mac-Morpho corpus. The authors used word embeddings as features and multilayer perceptron neural network as a model builder. The results obtained are 97.57% overall accuracy on the original Mac-Morpho corpus, 97.48% on the authors' first revised version, and 97.33% on the second revision.

Wang et al. (Wang et al., 2015) developed a POS tagger for English by using Bidirectional Long Short-Term Memory Recurrent Neural Network with word embeddings as features. The approach used by the authors achieved state-of-the-art performance of 97.40% tagging accuracy tested on a Penn Treebank WSJ (Wall Street Journal) test set.

Silva et al. (Silva et al., 2013) proposed an approach to the POS tagging problem that divides it into two different tasks: a learning task and an optimization task. The authors tackled each of the tasks with evolutionary computation techniques: genetic algorithms (GA) and a particle swarm optimizer (PSO). The results obtained are 96.749 % accuracy for PSO-Tagger using a swarm of 20 particles evolving during 50 generations and 96.728% accuracy for GA-Tagger having 100 individual genes evolving during 20 generations.

Modi and Nain (Modi and Nain, 2017) developed a parts-of-speech tagger for Hindi using rule-based approach with 29 standard tags including two special tags for date and time. The authors used a Hindi corpus of around 9000 words to increase tagging and rule-based approach to decrease the size of already trained corpus. The authors' proposed system yields 91.84% of average precision and 85.45% of average accuracy.

3.3. Summary

The existing works made an effort to improve the accuracy of Amharic POS tagger by using different corpus-based algorithms. However, the existing approaches have the following limitations.

- The use of manually designed features which may lead to miss some important features, in case of low observation of the morphology of the language. The manual design of features also requires linguistic expertise and it is time taking.

- Choice of the classifiers used. Different state-of-the-art methods were applied in the previous researches. Currently, there are additional state-of-the-art methods which may help to improve the classification accuracy. Hence, different versions of recurrent deep neural networks are used to develop the tagging models of this study.
- The use of prepositions and conjunctions as they are attached with the other parts-of-speech. For example, a preposition (PREP) and a noun (N) attached together form a word which becomes member of a different tag, noun with preposition (NP). The combination of the two words form a single word and an additional tag different from the existing tags for the new word. This case forces the corpus to have lower number of instances and a greater number of tags.

In this research, in line with the existing approaches, it is aimed to improve the accuracy of Amharic POS tagger. In particular, it is aimed to address the above problems to get a better accuracy.

4. Proposed Approach

The architecture of the proposed system has some subcomponents (enclosed by broken lined shapes) which make it different from the previous researches. One of the components is, the use of a morphological analyzer to segment prepositions and conjunctions attached with the other parts-of-speech. The other is, the use of automatically generated features of words instead of hand-crafted features to ease both the design of features and the complexity of the feature extractor. The classifiers are also developed using state-of-the-art neural networks (recurrent deep neural networks having a long-term memory) which can be considered as different tasks from previous studies.

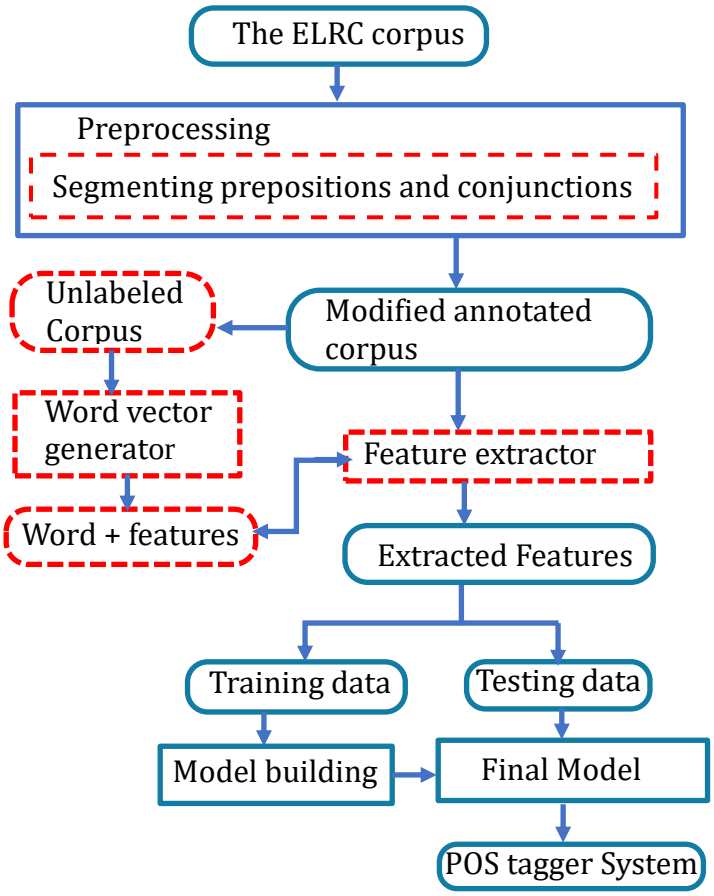

Figure 4-1 Proposed approach of this study

4.1. Data Collection

The dataset used in this study is prepared by the Ethiopian Languages Research Center (ELRC) of Addis Ababa University as a project called "The Annotation of Amharic News Documents" (Demeke and Getachew, 2006). About 1065 news articles were taken from the Walta Information Center (WIC) in preparing the corpus. The annotated corpus is presented in Amharic *Fidel* (Ethiopic[2]) and in its transliterated form using Latin alphabets. The ELRC corpus contains some unwanted (for this study) non-Amharic tags such as date, title and body since it is presented in xml form. Figure 4.2 shows the screenshot of the first part of the original corpus.

```
<?xml version="1.0" ?>
<!DOCTYPE amnews94 (View Source for full doctype...)>
<!-- -->
<amnews94>
<document>
<filename>mes01a1.htm</filename>
<title>ልዩ <ADJ> ዞኑ <N> ያስገነባቸው <VREL> ፕሮጀክቶች <N> 50ሺ <NUMCR> ነዋሪዎችን <N>
<dateline place="አዳማ" month="መስከረም" date="1/1994/(WIC)/" />
<body>

በአዳማ <NP> ልዩ <ADJ> ዞን <N> በተጠናቀቁ <VP> የበጀት <NP> አመት <N> በ6 ነጥብ 8ሚሊየን

<copyright>copyright 1998 - 2002 Walta Information Center</copyright>
</body>
```

Figure 4-2 Part of the Original ELRC corpus

The unwanted parts for this study need to be removed from the dataset. Even if much time and effort is put in preparing the corpus, it has many spelling errors and tagging inconsistencies. Extensive work is done by this study to fix these problems.

4.2. Preprocessing

Preprocessing includes cleaning, tokenization and segmentation. Transliteration is not one of the preprocessing tasks in this study since the Ethiopic version of the corpus is used for training and testing.

[2] ***Fidel*** is the set of alphabets of Amharic language and each character is also called a *Fidel*. The Amharic alphabet (Fidel) is usually known as Ethiopic.

Cleaning: includes the task of removing unwanted (for this study) non-Amharic characters, correcting many misspelled words and fixing tagging inconsistencies. More than 6937 misspelled words were corrected based on the errors obtained in the quarter of the corpus.

Tokenization: is splitting sentences and words from one another to make them suitable for analysis. Tokenization is part of some processes such as segmentation, word vector generating and feature extraction.

Segmentation: is done to separate prepositions and conjunctions attached with the other parts-of-speech. This step increased the number of prepositions, conjunctions and other parts-of-speech while it decreased the number of tags. For example, the Amharic words like ለአበበ meaning 'to Abebe' are tagged, in the original corpus, as NP which indicates a preposition is attached with a noun. After segmentation, ለ meaning 'to' is added to the category of prepositions (PREP) and አበበ (Abebe) is added to the category of nouns (N) which contributes to the increment of the number of instances in PREP and N categories. The segmented prepositions and conjunctions then stand by themselves as other words and they can be passed to the word vector generating component together with the other words. This helps to have their own representation of features to be used for training and testing.

4.3. Development Tools

Different programming languages and tools were used in the progress of the study and they are discussed as follows.

Text Processor: is a tool developed in this study using Java. The tool is used to clean the tags from the corpus and enables to get a raw text. The other significant task done using the text processor is the extraction of the required features from the output of HornMorpho. The Text Processor is also used to convert the ARFF file format to csv file format, input of deep learning algorithms used in this study. Deeplearning4j library is also used with Java to generate word vectors on another IDE (IntelliJ IDEA Community Edition 2017.3.2).

Python:

Python is used to apply segmentation since the morphological analyzer is a python library. It is also used to implement the deep learning neural networks.

HornMorpho:

HornMorpho is a python library developed for Amharic, Tigrigna and Afan Oromo to do morphological analysis, of which the output can be used in different NLP applications (Gasser, 2011). HornMorpho's output has many important features such as the part-of-speech of the given word, its root or stem, subject (gender, plural), preposition, conjunction etc. But, the POS feature for each word in HornMorpho is a noun, a verb or a copula which ignores words of the other parts-of-speech and groups them to one of the three. Hence, the POS feature, in the HornMorpho output, is not matured to be used for other applications.

The required features from the output of HornMorpho for this study are the stem or root of the given word and the preposition or the conjunction segmented from it, if it does have any.

Weka:

Weka (Waikato Environment for Knowledge Analysis) is an open source software under the GNU General Public License. Weka is developed at the University of Waikato in New Zealand and it is written using Java. Weka provides implementations of state-of-the-art data mining and machine learning algorithms. It contains modules for data preprocessing, classification, clustering and association rule extraction ("Introduction to Weka – A Toolkit for Machine Learning," n.d.).

TensorFlow:

TensorFlow is developed by researchers and engineers from the Google Brain team of Google's AI organization. It strongly supports machine learning and deep learning and has the flexible numerical computation core which can be used across many other scientific domains. TensorFlow is one of the possible backends of Keras which are TensorFlow itself, Cognitive Toolkit (CNTK) and Theano ("TensorFlow: An Open Source Machine Learning Framework for Everyone," n.d.). In this study, TensorFlow is used as a backend of the Keras application program interface (API).

Keras:

Keras is a high-level neural network API developed in Python and uses TensorFlow, CNTK or Theano as a backend to run properly. Keras allows for easy and fast prototyping through its user friendliness, modularity, and extensibility. It is compatible with Python 2.7-3.6 and supports both convolutional and recurrent neural networks as well as their combinations ("Keras: The Python

Deep Learning Library," n.d.). In this study, Keras is used to develop the recurrent neural networks easily.

Pandas:

Pandas is a Python package which presents an easy way of working with relational or labeled data by providing fast, flexible, and expressive data structures. It has two primary data structures: Series (1-dimensional) and DataFrame (2-dimensional) (McKinney and PyData Development Team, 2018). The 2-dimensional data structure is used in this study which then is converted into 3-dimensional data by using the NumPy package of Python. The LSTM and Bi-LSTM deep neural networks require a 3-dimensional input; hence, the 2-dimensional data was changed into a 3-dimensional data.

Scikit-learn:

Scikit-learn is a Python module distributed under the simplified BSD license which focuses on bringing machine learning to non-specialists. The library is focused on modeling data rather than manipulating data. In this study, it is used for evaluating the models developed using the Keras package (Fabian et al., 2011).

4.4. Dataset Preparation

After cleaning and fixing problems of the corpus, the tags were removed from a copy of the dataset to have a raw text since the segmentation is done on the words only. More explanation on segmentation is presented below in section 4.4.1.

A modified tagged corpus is prepared by relating the tagged corpus before segmentation and the extracted features of the HornMorpho output. A copy of the modified tagged corpus was then used to prepare another raw text to be used as input of the word vector generator. The output of the word vector generator represents the features of words in the given corpus. Then, by extracting features, the dataset was prepared into two formats. The first format is the ARFF (Attribute Relation File Format) format which was used in Weka tool that has many built-in classifiers. The second file format is the csv (Comma Separated Value) format which is used as input of the deep neural network.

4.4.1. Data Preparation by Segmentation

The raw text prepared from the cleaned original annotated corpus was given to the HornMorpho morphological analyzer. The morphological analyzer gives an output having the following format shown below in Figure 4.3.

```
word: በአዳማ
?POS: noun, stem: አዳማ
 preposition: be

word: ልዩ
POS: noun, stem: ልዩ

?word: ዞን

word: በተጠናቀቀው
POS: verb, root: <Tnqq>, citation: ተጠናቀቀ
 subject: 3, sing, masc
 object: 3, sing, masc
 grammar: perfective, reciprocal, passive, relative, definite
 preposition: be
POS: verb, root: <Tnqq>, citation: ተጠናቀቀ
 subject: 3, sing, masc
 grammar: perfective, reciprocal, passive, relative, definite
 preposition: be
```

Figure 4-3 Part of the Output of HornMorpho

As it can be observed from Figure 4.3, some words have no analysis result while others have one or more analysis results. The result of the morphological analyzer shows that the first word በአዳማ has a preposition በ (be) (see Figure 4.3). The prepositions and conjunctions attached in such a manner are presented in a transliterated format, not in Amharic Fidels (Ethiopic). Thus, they need to be converted back to Ethiopic as this version of the corpus is used in the experiments. It is not difficult to turn these transliterated prepositions and conjunctions into Ethiopic since they are known and small in number.

The tag assigned for በአዳማ (*be'Adama*) in the original annotated corpus is NP, to mean, a preposition is attached with a noun. After the preposition is segmented from the noun, the two words will get their own different tags. Thus, በ (*be*) gets the PREP tag and አዳማ (*Adama*) gets the N tag. By doing this, the tag NP in the original ELRC annotated corpus is removed from the tagset

membership in the new modified annotated corpus. PREP and N are already existing tags and assigning words በ and አዳማ, respectively, to the tags increases their number of instances. The increment of instances per each category (tag) and the decrement of the tagset members is the expected aspect to contribute positively to the POS tagger. As a result, the total number of instances in the modified corpus also increases.

The other words in the above screenshot such as ልዩ and ዞን either don't have a preposition (or conjunction) or no analysis result. In such cases, the words are taken simply with their own original tags. Words like በተጠናቀቀ, on the other hand, has many possible analysis results as shown above. Such cases are resolved by taking the first analysis result from the possible lists.

Words such as የኢትዮጵያ (Ethiopian) are tagged as NP in the original ELRC corpus, saying that a preposition የ (*ye*) is attached with the noun ኢትዮጵያ (Ethiopia). But, the author of HornMorpho, Michael Gasser, argued that የ (*ye*) is genitive indicator rather than a preposition. The author's argument is accepted in this research and action is taken according to it. Thus, words such as የኢትዮጵያ (*ye'Ethiopia* – Ethiopian), which were tagged as NP in the original ELRC corpus, are retagged as N in the modification of the corpus by this study.

The original ELRC tagset that have 30 tags is presented in Table 4.1.

Table 4.1: The ELRC tagset having 30 tags.

No	Tag	Definition
1	VN	Verbal Noun
2	NP	Noun with preposition
3	NC	Noun with conjunction
4	NPC	Noun with preposition and conjunction
5	N	Any other noun
6	AUX	Auxiliary verb
7	VREL	Relative verb
8	VP	Verb with preposition
9	VC	Verb with conjunction
10	VPC	Verb with preposition and conjunction
11	V	Any other verb

12	ADJP	Adjective with preposition
13	ADJC	Adjective with conjunction
14	ADJPC	Adjective with preposition and conjunction
15	ADJ	Any other adjective
16	PRONP	Pronoun with preposition
17	PRONC	Pronoun with conjunction
18	PRONPC	Pronoun with preposition and conjunction
19	PRON	Any other conjunction
20	NUMCR	Cardinal numbers
21	NUMOR	Ordinal numbers
22	NUMP	Numeral with preposition
23	NUMC	Numeral with conjunction
24	NUMPC	Numeral with preposition and conjunction
25	PREP	Prepositions
26	CONJ	Conjunctions
27	ADV	Adverbs
28	INT	Interjections
29	PUNC	Punctuation
30	UNC	Unclassified

The ELRC tagset shown above in Table 4.1 is designed based on 11 basic POS tags to provide more linguistic information. The 11 basic tags are noun, pronoun, verb, adjective, numeral, preposition, conjunction, adverb, interjection, punctuation and unclassified. The ELRC tagset is suggested to be expressed by a simplified tagset having 18 tags (Gebre, 2010). The suggested tagset includes P, C, ADJ, ADV, AUX, VREL, V, PRON, CONJ, INT, N, VN, NUMOR, NUMCR, NUM, PREP, PUNC, UNC. The suggested tagset avoids the multiple information represented in tags such as NPC, ADJPC, VPC, etc. and enables a tag to represent one basic information. But, some kinds of information are still missing in the ELRC tagset. For example, there is no reflection of gender and number of Amharic nouns. The details of the ELRC tagset also missed the representation of proper nouns which would have a significant contribution for NER.

The ELRC tagset is mapped into a new tagset having 14 members to be used in this study. Four tags from the above suggested 18 tags are not included here. The first two tags P and C are merged with their closet tags PREP and CONJ, respectively. The other tag not included in this study is INT because only two words (*መካከል* and *እጅግ*), which are members of other tags PREP and ADV, are tagged as INT in the original corpus. Thus, the two words are retagged by their respective tags and INT is removed from tagset membership. The fourth tag excluded from the above 18 tags is UNC. Words tagged as UNC in one sentence of the corpus are tagged with other tags in a different sentence of the same corpus with no context difference. Thus, by cross-checking such cases, words tagged as UNC are retagged by other tags assigned in a different place for those same words. Hence, a tagset used in this study having 14 tags is summarized as shown below in Table 4.2.

Table 4.2: The tagset used in this study

No	Tag	Definition
1	VN	Verbal Noun
2	N	Any other noun
3	AUX	Auxiliary verb
4	VREL	Relative verb
5	V	Any other verb
6	ADJ	Adjectives
7	PRON	Pronouns
8	NUMCR	Cardinal numbers
9	NUMOR	Ordinal numbers
10	NUM	Numerals
11	PREP	Prepositions
12	CONJ	Conjunctions
13	ADV	Adverbs
14	PUNC	Punctuation

Thus, a modified corpus (a third text file) is prepared from two text files: one containing the original ELRC annotated corpus and the other containing the output of the HornMorpho. Words and their corresponding tags in the original annotated corpus are structured into two ArrayLists of

equal length. Then, by taking a word from the ArrayList of words, its analysis result is searched from the output of the HornMorpho file. If the analysis result for a given word has either a preposition or a conjunction, then the word and its tag will be modified as described above (see Figure 4.3). If the analysis result of the word at a given index of the ArrayList of words has no preposition or conjunction, then the word itself will be written to the third file followed by the tag located at the same index in the ArrayList of tags.

Using such processes, the original ELRC annotated corpus is modified (see Figure 4.4).

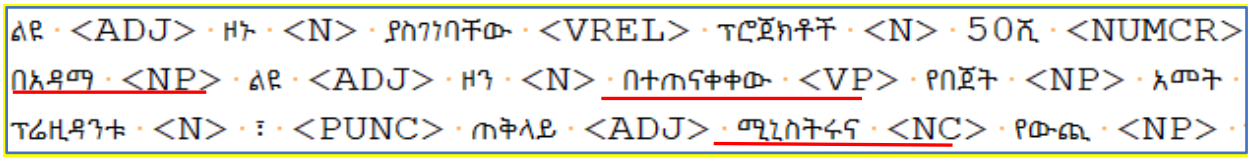

Figure 4-4-a The cleaned ELRC corpus

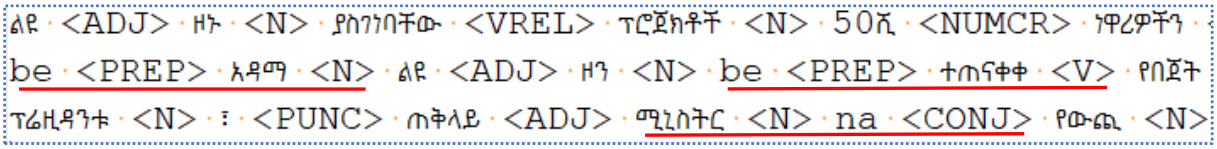

Figure 4-4-b The modified corpus containing transliterated prepositions and conjunctions

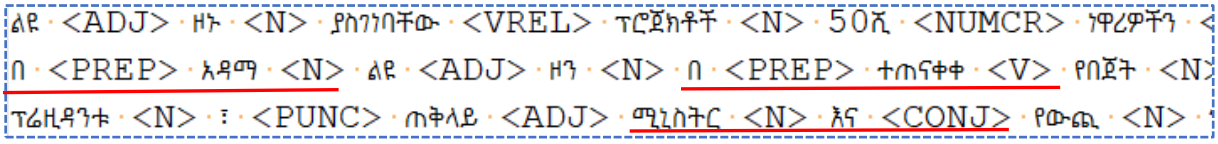

Figure 4-4-c The modified corpus containing prepositions and conjunctions in Amharic Fidel

Figure 4.4.a shows part of the original ELRC after passing the cleaning of spelling errors and tagging inconsistencies. Figure 4.4.b shows part of the modified corpus by taking prepositions and conjunctions directly from the HornMorpho output in their transliterated form. Figure 4.4.c shows part of the modified corpus after converting back the transliterated prepositions and conjunctions into Amharic alphabets (Ethiopic).

By applying segmentation on the attached prepositions and conjunctions, the size of the dataset has increased and the number of tagset members decreased. As an example, the number of prepositions increased from 5,525 to 20,929, conjunctions from 1,336 to 3,945 and Nouns from 68,721 to 112,771. Verbs and other parts-of-speech also increased significantly since the segmented prepositions and conjunctions freed them from tags such as VPC, ADJPC etc.

4.4.2. Data Preparation for Training

After having the modified corpus by using HornMorpho morphological analyzer, as shown in figure 4.4.c, the tags are removed from its copy to get a raw text. This raw text becomes an input for the word vector generator.

The data used has 220,260 instances including punctuation marks. It is a highly imbalanced data which may affect the classifier to predict to the highly abundant category. The maximum category is N(noun) which has 112,771 instances and the minimum is NUMOR (ordinal numbers) which has only 257 instances. Even, the second abundant category is the collection of verbs having 28,851 instances, which is close to one-fourth of the first abundant category. This makes difficult to make balancing between the categories which have far apart number of instances. Trying to balance them may distort the dataset significantly which may lead the neural network to the failure of learning real-world proportions. The gap between the number of instances in each category can be clearly understood from Table 4.3.

Table 4.3. The data obtained after segmentation

No	POS Tag	Number of Instances
1	VN	5331
2	N	112,771
3	PRON	2708
4	AUX	995
5	VREL	7448
6	V	28,851
7	ADJ	12,600
8	PREP	20,929
9	CONJ	3945
10	ADV	2415
11	NUMCR	5445
12	NUMOR	257
13	NUM	2,663
14	PUNC	13,902
Total		220,260

Then, it was tried to bring the smaller categories into the maximum category by using SMOTE (Syntactic Minority Oversampling TEchnique) tool of Weka. SMOTE is used to oversample the

least abundant category into the required percent of the most abundant category. The balancing led all the categories to have more than 100,000 instances each. To see the effect of balancing the dataset an experiment is conducted (experiment 4). This experiment uses the dataset prepared in such a manner having a total of 1,570,021 instances. The value of each category in the data used for experiment 4 is shown in Table 4.4.

Table 4.4. Dataset obtained by using SMOTE tool of Weka for balancing

No	POS Tag	Number of Instances
1	VN	111,951
2	N	112,771
3	PRON	113,736
4	AUX	111,440
5	VREL	111,720
6	V	115,404
7	ADJ	113,400
8	PREP	104,645
9	CONJ	114,405
10	ADV	111,090
11	NUMCR	114,345
12	NUMOR	112,052
13	NUM	111,846
14	PUNC	111,216
Total		1,570,021

The balancing of the dataset shown in Table 4.4 is not fair, because a category having only 257 instances is forced to have a closer number of instances (112,052) to the maximum category. Hence, another option is proposed to minimize such unfairness and the maximum category is under-sampled into a closer number of instances (30,840) to the second largest category. After under-sampling the largest class, SMOTE is applied for balancing which is better than the previous one. The total number of instances obtained from such a process become 388,173 as shown below in Table 4.5.

Table 4.5. The dataset obtained from undersampling and oversampling

No	POS Tag	Number of Instances
1	VN	27,934
2	N	30,840

3	PRON	27,080
4	AUX	28,855
5	VREL	27,557
6	V	28,851
7	ADJ	28,224
8	PREP	27,207
9	CONJ	24,604
10	ADV	27,772
11	NUMCR	28,858
12	NUMOR	25,957
13	NUM	26,630
14	PUNC	27,804
Total		388,173

4.2. Word Vector Generating

The vectors which represent words of the corpus are generated using tools such as Word2Vec which are developed by using neural networks. Word2Vec is part of the Deeplearning4j package for Java and Scala ("Word2Vec in Deeplearning4j", n.d.). Due to this, the generated features are also known as neural word embeddings. Word2Vec is a two-layer neural network designed to process an input text and give a set of vectors as an output. Word2Vec by itself is not a deep neural network but it maps a given text into a suitable numerical format that the deep nets can understand.

Unannotated corpus (raw text) is given to the Word2Vec tool so that each word is represented in vectors of real numbers with limited dimensional spaces. Then, the words are stored with their corresponding vector representations to be used for feature extraction.

4.3. Feature Extraction

The following algorithm is designed and used for the feature extraction task.

1. *Arrange the words and corresponding tags in annotated corpus into ArrayLists of 'Words' and 'Tags' keeping the indices the same.*
2. *Take a word from the ArrayList of 'Words'*
3. *Use the word to search its corresponding feature from the generated features*
4. *Copy the line containing the word with its features*

5. *Append the tag located at the same index at the end of the copied line*
6. *Write the line into a third file containing the extracted features.*
7. *Loop to step 2 until the last index of the ArrayLists*

The feature extractor searches for the corresponding features of the words in the annotated corpus from the storage of words along with their features. Tokenization is done on the modified annotated corpus shown below using Figure 4.5. Words of this modified annotated corpus are used as searching indices of their own word vectors from the storage of word vectors depicted in Figure 4.6.

ልዩ <ADJ> ዞኑ <N> ያስገኝባቸው <VREL> ፕሮጀክቶች <N> 50ሺ <NUMCR> ነዋሪዎችን <N> ተጠቃሚ <ADJ> አደረጉ <V> ። <PUNC>
በ <PREP> አዳማ <N> ልዩ <ADJ> ዞን <N> በ <PREP> ተጠናቀቀ <V> የበጀት <N> አመት <N> በ6 ነጥብ 8ሚሊየን <NUM> ብር <N>
ፕሬዚዳንቱ <N> ፣ <PUNC> ጠቅላይ <ADJ> ሚኒስትር <N> እና <CONJ> የውጪ <N> ጉዳይ <N> ሚኒስትር <N> አደጋው <N> የኢትዮጵያን ‹
በ <PREP> አሜሪካ <N> ኒውዮርክና <N> ዋሽንግተን <N> ትናንት <ADV> የደረሱትን <VREL> አራት <NUMCR> የአጥፍቶ <VREL> መጥፋት
ኦህዴድ <N> ከ <PREP> ጉባኤ <N> ተጠናክሮ <V> ወጥቷል <V> ። <PUNC>

Figure 4-5: Part of the modified annotated corpus

When word vectors are generated, the order of words becomes dispersed as shown below using Figure 4.6. Figure 4.5. shows the first lines of the modified annotated corpus while Figure 4.6 shows the first lines of the generated word vectors with corresponding words. The features of words in the modified annotated corpus are searched from such a dispersed storage of word vectors.

የጎደሬ −0.006521324627101421 ... −8.445163257420063E−4
ብቃታቸው −0.011472159065306187 ... 0.003154917620122432 7
ፈንላንድና −5.049884784966707E−4 ... −0.012615488842129707
አሳድሯል 0.0011524289147928357 ... 0.003909721039235592
የወጣው −0.01567731238901615 ... 0.02024085633456707
በፈትም −0.022752445191144943 ... −0.010487488470971584
ዘመኑ −0.022725047543644905 ... 0.006806608289480209

Figure 4-6 Part of the generated word vectors

The line, in Figure 4.6, containing the feature of a word in Figure 4.5 is copied into a third file shown using Figure 4.7. Then, the tag of the currently concerned word is added at the end of the line in the third file and searching continues with the next words.

```
ልዩ 2.4211357231251895E-4 0.0016628832090646029 ... 0.003255050163716078 ADJ
ዞኑ -0.02282402478158474 -0.013213218189775944 ... 0.0017324904911220074 N
ያስገነባቸው 0.013919714838266373 -0.00963563472032547 ... 0.02966431900858879 VREL
ፕሮጀክቶች -0.07076674699783325 -0.07002998143434525 ... -0.060365017503499985 N
50ሺ -0.023166967555880547 -0.011037144809961319 ... 0.010263576172292233 NUMCR
ነዋሪዎችን -0.007632601074874401 0.04421665146946907 ... 0.05906081944704056 N
ተጠቃሚ -0.12058249861001968 0.020284833386540413 0.09878786653280258 0.154811620
```

Figure 4-7 The extracted features with tags at the end

After extracting features for words of the modified annotated corpus, the data is given as input for the classifiers on Weka and those developed by using deep neural networks. The data is divided into training and testing as 80-20 ratio, respectively. A 10-fold cross validation is also used with the J-48 decision tree classifier in the first experiment.

4.4. Model Building

Model building is implemented mainly using recurrent deep neural networks. RandomTree and J-48 decision tree classifiers of Weka are also used for model building and testing. The classifiers learn the categories of words from the training dataset and build models.

4.5. Evaluating the Model

The models built are tested with a testing dataset. Based on their learning, the models predict the categories of the given inputs and those predictions are evaluated using metrics. The metrics used in this study are precision, recall, F-measure and accuracy which are discussed in section 4.5.1.

4.5.1. Evaluation Metrics

The metrics used for evaluating the models developed in this study are accuracy, precision, recall and F-measure. These metrics are defined using some common terms obtained from a confusion matrix.

The measures obtained from confusion matrix and used to determine the other metrics are True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). True Positive represents the number of instances categorized correctly to their corresponding class. True Negative represents the number of instances classified as they are *not* part of a category where they actually are *not members*. False Positive represents the number of instances classified into other categories in which they are not actually members. Finally, False Negative represents the

number of instances that are not classified as member of their corresponding category where they actually are members (Sunasra, 2018).

The measures associated with confusion matrix can be summarized as shown in Table 4.6.

Table 4.6 Summary of measure associated with confusion matrix

		Actual	
		Positives (1)	Negatives (0)
Predicted	Positives (1)	TP	FP
	Negatives (0)	FN	TN

True Positives and True Negatives are the correct predictions since member instances are predicted as members, and non-members are predicted as non-members. After defining the basic measures associated with the confusion matrix, the other performance metrics, which based on these measures, can be defined as follows.

Accuracy: is the number of correct predictions (sum of TP and TN) made by the model over all kinds of predictions made.

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}}$$

Accuracy can be a good performance measure if the number of instances in each category are balanced and it should never be used if there is a majority class from the given categories.

Precision: indicates the percent of correct predictions from correctly and incorrectly predicted instances as members of categories. Precision is viewed together with recall which is defined below.

$$\mathrm{Precision} = \frac{TP}{\mathrm{TP} + \mathrm{FP}}$$

Recall: measures the percentage of the relevant instances that have been retrieved over the total amount of relevant instances.

$$Recall = \frac{TP}{TP + FN}$$

Precision describes the performance of the given classifier as how many it predicts correctly, whereas recall shows how many are missed by the classifier. Minimizing the false negatives improves recall while reducing the false positives improves precision without affecting recall.

F1 Score (F-measure) is a harmonic mean of precision and recall. It is a better evaluation metric for categories having non-balanced number of instances since it gives an appropriate score than a simple arithmetic mean.

$$F - measure = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall}$$

5. Experiments and Results

In this section, the evaluation of the proposed approach will be presented. In particular, it is aimed to answer the following research questions.

RQ1: ***[Neural word embeddings for Amharic POS tagging]:*** Could neural word embeddings improve the performance of Amharic POS tagging?

RQ2: ***[Segmentation for Amharic POS tagging]***: Could segmentation of prepositions and conjunctions attached with the other parts-of-speech improve the Amharic POS tagger?

RQ1 helps to examine the performance of Amharic POS tagger developed using word embeddings in place of manually designed features. Neural word embeddings were tested on named entity recognition on the same language (Demissie, 2017) and tested in different natural language processing applications of different languages (Fonseaca et al., 2015), (Wang et al., 2015), (Pham and Le-Hong, 2017). Encouraging results were obtained from these studies.

RQ2 investigates the impact of segmentation on Amharic POS tagger. First, the corpus is modified using HornMorpho (Gasser, 2009) and neural word embeddings generated for words in the modified corpus. The neural word embeddings are then used in the development of the parts-of-speech tagger. To see the effect of segmentation on the Amharic POS tagging, evaluation is done on the corpus before and after segmentation.

5.1. Experimental Setup

Classifications done using the Weka environment run on a desktop computer and those using the deep neural networks were done on another high-performance desktop computer. The details of both computers can be seen from Table 5.1 and Table 5.2, respectively.

Table 5.1 Specifications of the desktop computer used for testing on Weka.

Manufacturer	Dell
Memory	4GB (3.89GB usable)
Processor	Intel® Core™ i7-3770 CPU @ 3.4GHz
Operating System	Windows 10 Pro

Table 5.2 Specifications of desktop computer used for testing deep learning algorithms.

Manufacturer	HP
Memory	16GB
Processor	Intel® Core™ i7 CPU 950 @ 3.4GHz
Operating System	Ubuntu 16.04 LTS

5.2. Experimental Scenarios

To examine the hypothesis of this study, three experimental scenarios were done. The first is, to check if neural word embeddings can encode syntactic and semantic information about words to replace manually designed features. For this scenario, a raw text was made from the modified annotated corpus. Then, the raw text was given to a Word2Vec tool to get word vectors of constant dimension. These word vectors were tested by finding ten closest words to some given words of different parts-of-speech as shown in Table 5.3. The word vectors were also tested by using them on J-48 decision tree and RandomTree classifiers of Weka.

The second experimental scenario is, to test the effect of using segmentation on POS tagging. For this scenario, the raw text file prepared from the original ELRC corpus was used to generate the morphological analysis result of the file. After getting the morphological analysis result, the corpus was modified to have segmented prepositions and conjunctions. By using this modified corpus, word vectors were generated and used in classification. The evaluation result before and after segmentation is presented in Table 5.4. The remaining experiments discussed below are settled based on experiments 1 and 2.

The third experimental scenario is to test the performance of state-of-the-art deep learning algorithms on Amharic POS tagging using neural word embeddings as features. This scenario was implemented by using libraries such as TensorFlow and Keras on Python. The neural word embedding inputs were arranged in the format that the deep learning algorithms require.

The fourth experimental scenario is to test the impact of balancing the number of instances of each category on POS tagging. In this experiment, the dataset having 1,570,021 instances shown in Table 4.4 is used. The fourth experimental scenario differs from the third only with the size of the dataset which is presented in Table 4.4.

The fifth experimental scenario aims to examine the impact of the manner of balancing the dataset on Amharic POS tagging. The dataset used in Experiment 4 has a highly distorted representation of the real-world proportion of categories. Experiment 5 aims to examine the impact of reducing such distorted representation on POS tagging. The details of the dataset used in this experiment are presented in Table 4.5.

5.3. Results

5.3.1. Neural word embeddings for Amharic POS tagging

Experiment 1: was done to answer RQ1. The experiment aims to see if the neural word embeddings can replace manually designed features with comparable or better results. Two experiments were done for this experiment, one using SVM and the other using J-48. *Gebre (Gebre, 2010) obtained an accuracy of **90.43**% from SVM with a 10-fold cross validation on a 206,929 dataset and 30 tags using hand-crafted features. An experiment was done with the same number of tags and data size using neural word embeddings on SVM which resulted in **44.64**% accuracy. Since the performance of SVM on neural word embeddings is worse than on the hand-crafted features, the dataset used for SVM was tested on deep learning algorithms. The deep learning algorithms perform better than the SVM as shown in Table 5.3.*

Table 5.3 Word embeddings with SVM 10-fold cross validation (All values are in percentage.)

	Precision	Recall	F-Measure	Accuracy
MLP	79.65	79.48	79.08	79.48
LSTM	81.32	81.66	81.31	81.66
Bi-LSTM	82.70	82.80	82.68	82.80

On the other hand, J-48 was applied on 19,634 words and achieved 84.9% accuracy by using manually designed features in a previous research (Kebede, 2009). In this study, J-48 with a 10-fold cross validation was applied on the same sized dataset and resulted in 68.84% accuracy by word embedding features.

Table 5.4 Word embeddings vs hand-crafted features (All values are in percentage.)

J-48 (Decision Tree)	Accuracy	Data Size	Number of Tags
Hand-Crafted Features	84.9%	19,634	29
Neural word embeddings	68.84%	19,634	29

The results in Table 5.3 and Table 5.4 show that neural word embeddings can be used to develop Amharic POS tagger. Even if a performance improvement is not observed from this experiment, it gives a hope to replace hand-crafted features with word embeddings to develop Amharic POS taggers. The features used in this experiment were generated automatically from the original annotated corpus, before segmentation was done. The two experiments were done on the same data size and the same number of tags. The only difference is the design of features. The hand-crafted features showed better performance than the automatically generated features. We can observe that using hand-crafted features lowers the dependency on the size of the corpus than neural word embeddings.

5.3.2. Segmentation for Amharic POS tagging

Experiment 2: was done to get answer for RQ2, to examine the importance of segmentation in developing Amharic POS tagger. This was also tested using Weka's RandomTree classifier by splitting the dataset into 80% for training and 20% for testing.

Table 5.5 Results for RQ2 (Segmentation for Amharic POS tagging) (All values are in percentage.)

	Non-segmented	**Segmented**
Precision	83.1	88.4
Recall	83.2	88.3
F-measure	83.1	88.3
Accuracy	83.21	88.32
Dataset Size	194,967	220,260
Number of Tags	30	14

The results in Table 5.5 show that segmenting prepositions and conjunctions from the other parts-of-speech can help to improve the accuracy of the Amharic POS tagger. About a 5% improvement is shown in the result by increasing the total number of instances and decreasing the number of tags.

Experiment 3: was conducted to examine the performance of deep neural networks on the modified data by means of segmentation and using the neural word embeddings as features. It needs to consider the imbalanced proportion observed in the data shown in Table 4.3 which is used in this experiment. There is a huge gap between the categories having maximum (112,771) and

minimum (257) number of instances. The results shown below in Table 5.6 are obtained by dividing the data into 80% for training and the remaining 20% for testing.

Table 5.6 Deep learning algorithms for Amharic POS tagging. (All values are in percentage.)

	MLP	**LSTM**	**Bi-LSTM**
Precision	86.64	88.43	88.82
Recall	86.87	88.53	88.93
F-measure	86.30	88.45	88.82
Accuracy	86.87	88.53	88.93

A deep recurrent neural network of multi-layer perceptron neurons having a total of five layers, 100 neurons on each layer except the output layer having 15 neurons, resulted in 86.87% accuracy.

More advanced deep recurrent neural networks consisting of long short-term memory neurons resulted in a similar accuracy which is about 88.53% (88.45% F-measure) having 100 LSTM units in each of the four layers.

The same dataset was also tested on a bidirectional long short-term memory recurrent neural network (Bi-LSTM RNN) having four layers and 120 smart neurons in each layer. The accuracy obtained is about 88.93% and the F-measure was 88.82%.

Experiment 3 was done on the dataset in Table 4.4 seeking an improved performance for the Amharic POS tagger. But, the categories in the dataset have unbalanced number of instances and the accuracy cannot pass over 88.93% as shown in Table 5.6.

The recurrent neural network structures used in this experiment are applied for the other experiments presented below by changing the size of the dataset.

Experiment 4: was done to examine the performance of deep neural networks on a relatively balanced and larger dataset. The dataset shown in Table 4.4 is used in this experiment. Even if, the data is distorted with respect to the real-world proportion, the evaluation results show improvement over what is observed from experiment 3 (see Table 5.7).

Table 5.7 Deep learning on a balanced data. ***(All values in percentage)***

	MLP	LSTM	Bi-LSTM
Precision	89.00	92.81	93.70
Recall	88.88	92.8	93.70
F-measure	88.81	92.75	93.67
Accuracy	88.88	92.8	93.70

From Table 5.7, balancing the number of instances among the categories increased the performance of the Amharic POS tagger. But, the size of the dataset increased significantly in the process of balancing. This leads to experiment 5 which tries to consider the real-world proportion of the categories when balancing.

Experiment 5: was done to examine the performance of deep neural networks on a relatively better-balanced dataset by under sampling the maximum category. The dataset shown in Table 4.5 is used in this experiment and the result is shown below by Table 5.8.

Table 5.8 Deep learning on a more appropriately balanced data. ***(All values in percentage)***

	MLP	LSTM	Bi-LSTM
Precision	88.33	89.47	90.22
Recall	88.14	89.39	90.18
F-measure	88.11	89.36	90.14
Accuracy	88.14	89.38	90.18

From experiment 4 and 5, it can be understood that the developed classifiers are more sensitive to the amount of data than the balance between each category. As described above, experiment 4 uses a dataset having a total of 1,570,021 instances as shown in Table 4.4 while experiment 5 uses a dataset having 388,173 instances as shown in Table 4.5. The results of experiments 4 and 5 are presented in Table 5.7 and Table 5.8, respectively. From these tables, it can be observed that better results are observed from experiment 4, done on a larger dataset.

5.4. Threats to Validity

5.4.1. Internal Threats to Validity

Internal threats to validity are threats which are caused by instrumentation problems and uncontrolled variables. The instrumentations used in the experiments were computers and other concurrent processes run at the same time the experiments were running on the computers. Two computers were used for two cases: one for generating word vectors and testing on Weka and the other for testing the deep learning algorithms. Testing both cases on a single high-performance computer may give better results. The time required to extract the features was really significant which takes more than a day on the high-performance desktop computer even if time is not the major concern of this study. Trying the feature extraction on a better performing computer may result in lower time requirement.

5.4.2. External Threats to Validity

The dataset used in this study is lower than that of the other resource-rich languages such as English. The quality of the dataset is not also good and the same approach applied for a better dataset, both in terms of amount and quality, may bring better evaluation results. Different results may be observed on different input datasets.

Another threat here is, the use of segmented words directly with no morphological adaptability. By applying segmentation, the morphology of words is changed to their stem or root word which makes these words to have no indication of subject, plurality, gender and other required features. Using the segmented preposition or conjunction with the root word of no morphological change may be difficult to understand the sentence easily.

6. Conclusion and Future Works

6.1. Conclusion

Parts-of-speech tagging is one of the important natural language processing applications which is the identification and labeling of words of a given text with their respective lexical categories. It can be developed using different approaches such as rule-based, stochastic, artificial neural network and hybrid. These approaches have their own advantages and disadvantages in developing the POS tagger.

Studies have been conducted on parts-of-speech tagging for Amharic and other local languages. The POS tagger for Amharic is becoming improved from the beginning through repeated researches. Previous researches invested significant time, energy and linguistic expertise in manually designing the required features.

In this study, two basic tasks having a positive contribution to the Amharic POS tagger were done. The first task is segmenting prepositions and conjunctions attached with the other parts-of-speech. The second task is trying to simplify the design of features by generating them automatically using the Word2Vec tool. The Word2Vec tool gives real valued vectors of limited dimensional space as feature representations of words in the given text. The proposed approach was designed in such a manner that it can use these vectors as features instead of the manually designed features.

Different experiments were done based on the research questions to evaluate the performance of the proposed approach with neural word embeddings as features. The experiments were conducted on some classifiers on Weka environment and others developed using deep learning algorithms. The maximum accuracy values observed in different versions of recurrent deep neural networks are: 88.88% for MLP, 92.8% for LSTM and 93.7% for Bi-LSTM. The F-measure values for these networks are 88.81%, 92.75% and 93.67% respectively.

To summarize the observations obtained from this study:

- Segmentation of prepositions and conjunctions attached with the other parts-of-speech can have a significant impact on the improvement of the Amharic POS tagger.
- Neural word embeddings can encode syntactic and semantic features of words and thus they can replace the manually designed features of words.

- Neural word embeddings are easy to use both with classifiers on Weka environment and other classifiers developed by deep neural networks.
- The deep neural networks can result in better accuracies given a good quality input dataset having larger number of instances.

6.2. Future Works

Based on the observations described above in the conclusion part, the following possible future works can be recommended.

- The dataset used in this research and the others done previously on Amharic language used a medium sized corpus of lower quality. Improving the quality of the existing corpus and increasing its size can be a significantly important task.
- Segmenting the attached prepositions and conjunctions from the other parts-of-speech keeping the necessary information giving morphologies such as gender, person, number etc. can improve the Amharic POS tagger. When segmenting prepositions and conjunctions, the morphologies indicating gender, person, number and other information are missed. The segmented preposition or conjunction and the stem word are taken ignoring such important morphologies. Modifying the segmentation process in a way of keeping such morphologies can be an important task.
- A hybrid of neural word embedding features with manually designed features may also give a better performing Amharic POS tagger.

Bibliography

Abreha, T. G. (2010). *Parts of Speech Tagger for Tigrigna Language.* MSc Thesis, Addis Ababa University, Ethiopia.

Adafre, S. F. (2005). *Part of Speech Tagging for Amharic using Conditional Random Fields.* University of Amsterdam, The Netherlands.

Agatonovic-Kustrin, S., & Beresford, R. (1999). Basic Concepts of Artificial Neural Network (ANN) Modeling and Its Application in Pharmaceutical Research. *Journal of Pharmaceutical and Biomedical Analysis, 22*(2000), 717-727.

Alemu, B. (2013). *Amharic Named Entity Recognition.* MSc Thesis, Addis Ababa University, Information Science, Addis Ababa, Ethiopia.

Athavale, V., Bharadwaj, S., Pamecha, M., Prabhu, A., & Shrivastava, M. (2016). *Towards Deep Learning in Hindi NER: An Approach to Tackle the Labelled Data Scarcity.*

Bjerva, J. (2013). *Genetic Algorithms in the Brill Tagger - Moving towards language independence.* MA Thesis, Stockholms Universitet.

Brownlee, J. (2014, April 16). *A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library*. Retrieved July 23, 2018, from Machine Learning Mastery: https://machinelearningmastery.com/a-gentle-introduction-to-scikit-learn-a-python-machine-learning-library/

Brownlee, J. (2017, October 11). *What Are Word Embeddings for Text?* Retrieved November 12, 2018, from Machine Learning Mastery: https://machinelearningmastery.com/what-are-word-embeddings/

Demeke, G. A., & Getachew, M. (2006, March). *Manual Annotation of Amharic News Items with Part-of-Speech Tags and its Challenges.* Addis Ababa University, Ethiopian Languages Research Center.

Demissie, D. (2017). *Amharic Named Entity Recognition using Word Embeddig as a Feature.* MSc Thesis, Addis Ababa University, Addis Ababa, Ethiopia.

Fabian, P., Gael, V., Gramfort, A., Michel, V., & Thirion, B. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*.

Fonseaca, E., Rosa, J. L., & Aluisio, S. M. (2015). Evaluating Word Embeddings and Revised Corpus for Parts-of-Speech Tagging in Portuguese. *Journal of the Braziliaan Computer Society*.

Gamback, B., Olsson, F., Alemu, A., & Asker, L. (2009). *Methods for Amharic Part-of-Speech Tagging.*

Gasser, M. (2011). HornMorpho: A System for Morphological Processing of Amharic, Oromo, and Tigrinya. *Conference on Human Language Technology for Development.* Alexandria, Egypt.

Gebre, B. G. (June 2010). *Part of Speech Tagging for Amharic.* MA Thesis, Wolverhampton University, United Kingdom.

Hirpassa, S. (2017, Mar – Apr). Information Extraction System for Amharic Text. *International Journal of Computer Science Trends and Technology (IJCST), 5*(2).

(n.d.). *Introduction to Weka - A Toolkit for Machine Learning.* Winter School.

Kebede, G. (2009). *The Application of Decision Tree for Part of Speech (POS) Tagging for Amharic.* MSc Thesis, Addis Ababa University, Addis Ababa, Ethiopia.

Keras: The Python Deep Learning Library. (n.d.). Retrieved August 18, 2018, from Keras Documentation: www.keras.io

Mamo, G. (2009). *Part-of-Speech Tagging for Afan Oromo Language*. MSc Thesis, Addis Ababa University, Information Science, Addis Ababa, Ethiopia.

Mamo, G. (January 2009). *Part-of-Speech Tagging for Afan Oromo Language.* MSc Thesis, Addis Ababa University, Ethiopia.

McKinney, W., & PyData Development Team. (2018). *Pandas: Powerful Python Data Analysis Tookit.*

Megersa, D. (June 2002). *An Automatic Sentence Parser for Oromo Language using Supervised Learning Technique.* MSc Thesis, Addis Ababa University, Ethiopia.

Mekuria, Z. (2013). *Design and Development of Part-of-speech Tagger for Kafi-noonoo Language.* MSc Thesis, Addis Ababa University, Ethiopia.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality.* Google Inc., Mountain View.

Modi, D., & Nain, N. (2016). Parts-of-Speech Tagging of Hind Corpus Using Rule-Based Method. *Proceedings of the International Conference on Recent Cognizance in Wireless Communication and Image Processing.* Springer India.

NSS. (2017, June 04). *An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec*. Retrieved from Analytics Vidhya: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

Nurwidyantor, A., & Winarko, E. (n.d.). *Parallelization of Maximum Entropy POS Tagging for Bahasa Indonesia with MapReduce.* Universitas Gadjah Mada, Indonesia.

Olah, C. (2015, August 27). *Understanding LSTM Networks*. Retrieved July 14, 2018, from Colah's Blog: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Olejnik, G. (2017, December 03). *Word embeddings: exploration, explanation, and exploitation (with code in Python)*. Retrieved from Towards Data Science, A Medium Corporation: https://towardsdatascience.com/word-embeddings-exploration-explanation-and-exploitation-with-code-in-python-5dac99d5d795

Pham, T.-H., & Le-Hong, P. (2017). *End-to-end Recurrent Neural Network Models for Vietnamese Named Entity Recognition: Word-level vs. Character-level.* Hanoi, Vietnam.

Silva, A. P., Silva, A., & Rodrigues, I. (2013). A New approach to the POS Tagging Problem Using Evolutionary Computation. *Proceedings of Recently Advances in Natural Language Processing*, (pp. 619-625). Hisar, Bulgaria.

Sunasra, M. (2017, November 11). *Performance Metrics for Classification Problems in Machine Learning*. Retrieved August 20, 2018, from A Medium Corporation:

https://medium.com/greyatom/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b

Tachbelie, M. Y., & Menzel, W. (2009). Amharic Part-of-Speech Tagger for Factored Language Modeling. *International Conference RANLP 2009 - Borovets, Bulgaria*, (pp. 428–433). Hamburg, Germany.

Tachbelie, M. Y., Abate, S. T., & Besacier, L. (2011). Part-of-Speech Tagging for Under-Resourced and Morphologically Rich Languages – The Case of Amharic. *Conference on Human Language Technology for Development.* Alexandria, Egypt.

TensorFlow: An Open Source Machine Learning Framework for Everyone. (n.d.). Retrieved August 18, 2018, from TensorFlow: https://www.tensorflow.org

Tripathi, M. (2017, April 1). *Deep Learning for Sequences: RNN & LSTM*. Retrieved September 25, 2018, from LinkedIn: https://www.linkedin.com/pulse/deep-learning-sequences-rnn-lstm-manish-tripathi/

Wang, P., Qian, Y., Soong, F., He, L., & Zhao, H. (2015). *Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network.*

Word2Vec in Deeplearning4j. (n.d.). Retrieved September 12, 2018, from Deep Learning for Java: https://deeplearning4j.org/docs/latest/deeplearning4j-nlp-word2vec

Yulita, I. N., & Fanany, M. I. (2017). Bi-directional Long Short-Term Memory using Quantized Data of Deep Belief Networks for Sleep Stage Classification. *2nd International Conference on Computer Science and Computational Intelligence.* Bali, Indonesia.

Zewgneh, S. (October 2017). *English-Amharic Document Translation Using Hybrid Approach.* MSc Thesis, Addis Ababa University, Ethiopia.

Appendix

The Amharic Alphabet (Fidel)

- The transliterated representations of Amharic Fidels are presented based on the Amharic keyboard presented by Windows 10. There are also other representations such as Power Ge'ez.

A. The 238 Amharic characters

ሀ, he	ሁ, hu	ሂ, hi	ሃ, ha	ሄ, hie	ህ, h	ሆ, ho
ለ, le	ሉ, lu	ሊ, li	ላ, la	ሌ, lie	ል, l	ሎ, lo
ሐ, h.e	ሑ, h.u	ሒ, h.i	ሓ, h.a	ሔ, h.a	ሕ, h.	ሖ, h.o
መ, me	ሙ, mu	ሚ, mi	ማ, ma	ሜ, mie	ም, m	ሞ, mo
ሠ, sse	ሡ, ssu	ሢ, ssi	ሣ, ssa	ሤ, ssie	ሥ, ss	ሦ, sso
ረ, re	ሩ, ru	ሪ, ri	ራ, ra	ሬ, rie	ር, r	ሮ, ro
ሰ, se	ሱ, su	ሲ, si	ሳ, sa	ሴ, sie	ስ, s	ሶ, so
ሸ, Se	ሹ, Su	ሺ, Si	ሻ, Sa	ሼ, Sie	ሽ, S	ሾ, So
ቀ, qe	ቁ, qu	ቂ, qi	ቃ, qa	ቄ, qie	ቅ, q	ቆ, qo
በ, be	ቡ, bu	ቢ, bi	ባ, ba	ቤ, bie	ብ, b	ቦ, bo
ቨ, ve	ቩ, vu	ቪ, vi	ቫ, va	ቬ, vie	ቭ, v	ቮ, vo
ተ, te	ቱ, tu	ቲ, ti	ታ, ta	ቴ, tie	ት, t	ቶ, to
ቸ, ce	ቹ, cu	ቺ, ci	ቻ, ca	ቼ, cie	ች, c	ቾ, co
ኀ, hhhe	ኁ, hhhu	ኂ, hhhi	ኃ, hhha	ኄ, hhhie	ኅ, hhh	ኆ, hhho
ነ, ne	ኑ, nu	ኒ, nu	ና, na	ኔ, nie	ን, n	ኖ, no
ኘ, Ne	ኙ, Nu	ኚ, Ni	ኛ, Na	ኜ, Nie	ኝ, N	ኞ, No
አ, a	ኡ, u	ኢ, i	ኣ, aa	ኤ, ie	እ, e	ኦ, o
ከ, ke	ኩ, ku	ኪ, ki	ካ, ka	ኬ, kie	ክ, k	ኮ, ko
ኸ, Ke	ኹ, Ku	ኺ, Ki	ኻ, Ka	ኼ, Kie	ኽ, K	ኾ, Ko

ወ, we	ዉ, wu	ዊ, wi	ዋ, wa	ዌ, wie	ው, w	ዎ, wo
ዐ, aaa	ዑ, aaau	ዒ, aaai	ዓ, aaaa	ዔ, aaaie	ዕ, eee	ዖ, aaao
ዘ, ze	ዙ, zu	ዚ, zi	ዛ, za	ዜ, zie	ዝ, z	ዞ, zo
ዠ, Ze	ዡ, Zu	ዢ, Zi	ዣ, Za	ዤ, Zie	ዥ, Z	ዦ, Zo
የ, ye	ዩ, yu	ዪ, yi	ያ, ya	ዬ, yie	ይ, y	ዮ, yo
ደ, de	ዱ, du	ዲ, di	ዳ, da	ዴ, die	ድ, d	ዶ, do
ጀ, je	ጁ, ju	ጂ, ji	ጃ, ja	ጄ, jie	ጅ, j	ጆ, jo
ገ, ge	ጉ, gu	ጊ, gi	ጋ, ga	ጌ, gie	ግ, g	ጎ, go
ጠ, Te	ጡ, Tu	ጢ, Ti	ጣ, Ta	ጤ, Tie	ጥ, T	ጦ, To
ጨ, Ce	ጩ, Cu	ጪ, Ci	ጫ, Ca	ጬ, Cie	ጭ, C	ጮ, Co
ጰ, Pe	ጱ, Pu	ጲ, Pi	ጳ, Pa	ጴ, Pie	ጵ, P	ጶ, Po
ጸ, tse	ጹ, tsu	ጺ, tsi	ጻ, tsa	ጼ, tsie	ጽ, ts	ጾ, tso
ፀ, zhe	ፁ, zhu	ፂ, zhi	ፃ, zha	ፄ, zhie	ፅ, zh	ፆ, zho
ፈ, fe	ፉ, fu	ፊ, fi	ፋ, fa	ፌ, fie	ፍ, f	ፎ, fo
ፐ, pe	ፑ, pu	ፒ, pi	ፓ, pa	ፔ, pie	ፕ, p	ፖ, po

B. Additional Labial Amharic Characters

ሯ = ee

ሏ = lua፣ ሗ = h.ua፣ ሟ = mua፣ ሧ = ssua፣ ሯ = rua፣ ሷ = sua፣ ሿ = Sua፣ ቧ = bua፣ ቯ = vua፣ ቷ = tua፣ ቿ = cua፣ ኗ = nua፣ ኟ = Nua፣ ዟ = zua፣ ዧ = Zua፣ ዷ = dua፣ ጇ = jua፣ ጧ = Tua፣ ቋ = Cua፣ ፏ = Pua፣ ፗ = tsua፣ ፏ = fua፣ ፐ = pua።

ኰ	ኩ	ካ	ኴ	ኵ
ቈ	ቍ	ቋ	ቌ	ቍ
ኈ	ኍ	ኋ	ኌ	ኍ
ጐ	ጕ	ጓ	ጔ	ጕ