Jimma University

Jimma Institute of Technology

School of Graduate Studies

Department of Information technology

Part of Speech Tagging Using Artificial Neural Network for Tigrigna

By

Dawit Tekie

October, 2017

A Thesis submitted to Faculty of Computing of Jimma University in Partial Fulfillment of the Requirements for Degree of Master of Science in Information Technology

Principal Advisor: **Mr. GETACHEW M. (Ass Prof)**

Co-Advisor: **Mr. BEHAYLU S.**

October, 2017

Jimma, Ethiopia

Jimma University

Jimma Institute of Technology

School of Graduate Studies

Department of Information technology

Part of Speech Tagging Using Artificial Neural Network for Tigrigna

By

Dawit Tekie

October, 2017

Signature of the Advisor and Co-Advisor for Approval

Name	signature
1. Mr. Getachew M. (**Ass Prof**)	______________
2. Mr. Behaylu S. (Co-Advisor)	______________

Jimma University

Jimma Institute of Technology

School of Graduates Studies

Department of Information Technology

Part of Speech Tagging Using Artificial Neural Network for Tigrigna

By

Dawit Tekie

This thesis is authorized "Part of Speech Tagging Using Artificial Neural Network for Tigrigna" has been read and approved as meeting the School of Graduates Studies in partial fulfillment for the award of the degree of master of Information Technology in Jimma University, Jimma, Ethiopia

Approved Sheet

Mr. Getachew M. (Ass prof)		
Advisor	Signature	Date
Mr. Behaylu S.		
Co-Advisor	Signature	Date
External Examiner	Signature	Date
Internal Examiner	Signature	Date
Faculty Dean	Signature	Date

Dedicated To:

<u>My Mother Freweyni G/mariyam</u>

<u>MOM</u>: you are my base and foundation of my life. You dream to be a man and you realized. You planted the seed that I base my life on, and that is the belief that the ability to achieve starts in my mind.

Acknowledgment

First, I would like to thanks **Almighty GOD** for helping me in day to day my life activity and helping me to finalize my work peacefully.

I am extremely grateful thanks to my advisor **Mr. GETACHEW M.** for his constructive comments, suggestions and full devotion from the beginning to end of my thesis work. I want to thank also to **Mr. TEFERI K.** for his constructive comments when I was starting my thesis work. And thanks to my Co-Advisor **Mr. BEHAILU S.** for his advising on how to do my thesis.

I express my heartfelt gratitude to **Dr. Jason Brownlee** for providing me with necessary help and suggestions throughout the entire Thesis period.

I owe my deepest heartfelt appreciation to my best friend and sees like my brother **HAGOS A.** for his successful contribution in this work especially he helped me in implementation part as much as he can.

It is a pleasure to thank to my bro **MESAY ASCHALEW** who always wants to see my successful and he is always on beside of me to encourage in financial and knowledge case.

I want to thank to all my friends especially **ANDUALEM CHEKOL "idea inventor", TEKLAY MURUTS, ANGOSOM ALMAYOH, WUBETU BARUD**, **MULUGETA ATSBHA and GEREZGHER TSEGAY** for their material support and encouragement to achieve my thesis work.

Last but not least, I want to thank to my **mother, brother** and **sister** for their love and support to conduct this study.

"I LOVE YOU MOM"

Contents

Acknowledgment I

List of Tables V

List of figures VI

Abbreviation, Acronym and Definition VII

Abbreviation VII

Acronym and Definition VIII

Abstract IX

Chapter 1 1

1. Introduction 1

1.1. Statement of the Problem 3

1.2. Objectives 4

1.3. Methods and Methodology 4

1.3.1. Literature Review 4

1.3.2. Data Source and Preparation 4

1.4. Scope of the Study 5

1.5. Application of Results 5

1.6. Organization of the Thesis 6

Chapter 2 7

2. **Literature Review and Related Works** 7

2.1. Introduction 7

2.2. The Human Brain 10

2.3. Artificial Neural Network (ANN) 11

2.3.1. Component of Neural Network 14

2.3.2. Activation Function 17

2.3.3. Architecture of Neural Network 19

2.3.4. Types of Neural Networks 20

2.3.5. Learning in Artificial Neural Network (ANN) 23

2.3.6. Back-propagation Algorithm 24

2.4. Related Works 26

2.5. Summary 33

Chapter 3 34

3. Tigrigna POS and Tag sets 34

3.1. Introduction 34
3.2. Tigrigna Sentences Structure 34
3.3. Word Classification of Tigrigna 35
3.3.1. The Need for Categorizing a Word 35
3.3.2. Tigrigna Word Classes 36
3.4. Tigrigna Tags and Tag-sets 40
3.5. Summary 47
Chapter 4 50
4. Design of Tigrigna POS Tagger 50
4.1. Design Approaches and Techniques 50
4.2. Design Goals 51
4.3. Designing Neural Network Model 52
4.3.1. Feature Extraction 52
4.3.2. Representation of Input for MLP _tagger 54
4.3.3. Representation of Output for MLP_Tagger 54
4.3.4. Back Propagation Algorithm 54
4.4. Summary 57
Chapter 5 58
5. Experimentation and Discussion 58
5.1. Data Preparation 58
5.2. Implementation 61
5.3. Experimentation 64
5.4. Summary 72
Chapter 6 73
6. Conclusion and Future Works 73
6.1. Conclusion 73
6.2. Future Works 73
Reference 75
Appendices 80
Apendex1 80
Sample Manually Annotated Corpus: 80
Sample Word Dictionary: 80
Sample Lexical Probability 81

Sample Preprocessed Lexical Corpus Used for Artificial Neural Network..81
Appendex2..82
Pseudo Code for Sample Steps of Back Propagation Algorithm..82
1. Initialize a Network..82
2. Forward Propagate..83
3. Back Propagate Error..84
4. Train Network..85

List of Tables

Table 3.1 Tigrigna Demonstrative Adjective 39
Table 3.2 Tigrigna Tag-sets 49
Table 4.1 Probability of words given a tag 53
Table 5 Sample Lexical probabilities 59
Table 5.1 Sample representation of the input dataset with their lexical probability and target output 60
Table 5.2 Binary and decimal number representation of tags 61
Table 5.3 Performances of Network Models for different values of parameters 75% 65
Table 5.4 Performances of Network Models for different values of parameters 50% of the dataset 67
Table 5.5 Performances of Network Models for different values of parameters 50% of the dataset 68
Table 5.6 Performances of Network Models for different values of parameters 25% of the dataset 69
Table 5.7 Performances of Network Models for different values of parameters 100% of the dataset 70

List of figures

Figure 2.1   The Common Methods for the POSTaggers [17].. 9
Figure 2.2 Structure of a neural cell in the human brain [24, 26].. 11
Figure 2.3 The artificial neural network [17].. 12
Figure 2.4 Artificial neuron models.. 13
Figure 2.5 INNN and SNNC [24]... 15
Figure 2.6   Threshold functions a) Linear b) Hard-limit [32]... 18
Figure 2.7 Log-Sigmoid Function (left) Tan-Sigmoid Function (right) [32] 19
Figure 2.8 Feed-Forward (left) Feed-Back (middle) Lateral network (right).............................. 20
Figure 2.9 perceptron neurons model 21
Figure 2.10 multilayer perceptron networks.. 21
Figure 2.11 Sample structure back-propagation neural networks 22
Figure 2.12 A feed forward neural network, highlighting the connection from unit i to unit j. ... 24
Figure 3.1 Tag-set concept hierarchies [1] 42
Figure 4.1 Architecture of ANN based Parts of Speech Tagger... 53
Figure 4.2 Back propagation network architecture.. 56
Figure 5.1 Plot of Model Accuracy on Train and Validation Datasets 75% 66
Figure 5.2 Plot of Model Loss on Training and Validation Datasets 75%................................... 66
Figure 5.3 Plot of Model Accuracy on Train and Validation Datasets for 50%........................... 67
Figure 5.4 Plot of Model Loss on Training and Validation Datasets for 50% 68
Figure 5.5 Plot of Model Accuracy on Training and Validation Datasets for 25%...................... 69
Figure 5.6 Plot of Model Loss on Training and Validation Datasets for 25% 70
Figure 5.7 Plot of Model Accuracy on Train and Validation Datasets.. 71
Figure 5.8 Plot of Model Loss on Training and Validation Datasets 71

Abbreviation, Acronym and Definition

Abbreviation

AAND	Annotation of Amharic News Document
ANN	Artificial neural network
CRF	Conditional Random Field
ELRC	Ethiopian Language Research Center
HMM	Hidden Markova Model
INNN	Idealized Neuron of Neural Network
MLP	Multi-Layer Perceptron
NLP	Natural Language Processing
NN	Neural Network
POS	Part of Speech
POST	Part of Speech Tagging
SNNC	Structure of Neural Network and Components

Acronym and Definition

Part of Speech Tagging: In corpus linguistics, part-of-speech tagging (POS tagging or POST), also called grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context i.e., its relationship with adjacent and related words in a phrase, sentence, or paragraph.

Lexical Probability: We use lexical generation probabilities to estimate the probability that a POS tag will generate a particular word, independent of the surrounding words.

Artificial Neural Network: Neural networks or connectionist systems are a computational approach used in computer science other research disciplines, which is based on a large collection of neural units (artificial neurons), loosely mimicking the way a biological brain solves problems with large clusters of biological neurons connected by axons.

Back propagation Neural Network: The backward propagation of errors or back-propagation: is a common method of training artificial neural networks and used in conjunction with an optimization method such as gradient descent. The algorithm repeats a two phase cycle, propagation and weight update. When an input vector is presented to the network, it is propagated forward through the network, layer by layer, until it reaches the output layer. The output of the network is then compared to the desired output, using a loss function, and an error value is calculated for each of the neurons in the output layer. The error values are then propagated backwards, starting from the output, until each neuron has an associated error value which roughly represents its contribution to the original output.

Multilayer Perceptron Neural Network: A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate outputs. An MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one.

Abstract

The interaction between human to computer is a daily occurrences. Due to the rapid change of technologies, for example, HCI (Human computer Interaction) and NLP (Natural language processing) streamline the easiness and effectiveness of computer systems that can better communicate and assist human beings. For instance, Part of speech tagging is one attempt in the effort of understanding human language to identify the role of a word in a given context.

There are a significant amount of work have been done to address part of speech disambiguation problems towards sematic languages. For example, Part of speech tagging for Tigrigna has been done using hybrid approach; HMM tagger combined with the rule based tagger. Unlike the aforementioned approaches, a neural network approach is an effective method to achieve a better performance towards part of speech tagger for Tigrigna. Therefore, the main objective of the study is developing an effective part of speech tagger scheme for Tigrigna words to attain a better performance.

To achieve the previously specified objective, a neural network based part of speech tagging for Tigrigna scheme is proposed. The tagger is used Back propagation learning algorithm in order to train the neural network.

In order to evaluate the performance of the proposed scheme, an extensive experiment is conducted using a corpus comprised of the training set and the prepared testing set. The performance of the Tigrigna MLP tagger in the 75% of the dataset and 25% of the dataset is 90.726% and 85.693% using 10-fold cross validation technique respectively while the overall performance of the MLP tagger on the whole datasets which can be used K fold technique is 93.849% which takes a total time about 2 hour to train.

The experiment results demonstrate that MLP tagger achieve an efficient performance using prepared testing set. However, this result would improve better if there is more dataset for training and testing.

Keywords: ANN, POS Tagging, Tagging, Tigrigna, MLP and Back propagation

Chapter 1

1. Introduction

Natural Language Processing (NLP) is a branch of computational linguistics which is concerned with automated, computer processing of natural language such as speech acts or texts. NLP concerns to process and understand natural language using computers. Natural Language Processing is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications. Thus, it performs useful tasks like enabling human-machine communication, improving human-human communication, or simply doing useful processing of text or speech [15][19]. It includes techniques like word stemming (removing suffixes) or a related technique, lemmatization (replacing an inflected word with its base form), multiword phrase grouping, synonym normalization, part-of-speech (POS) tagging (elaborations on noun, verb, preposition etc.), word-sense disambiguation, and role determination (e.g. subject and object) [16]. NLP has many applications including machine translation, speech recognition, question answering, information retrieval system and parts of speech tagging. The above definition uses for all natural languages to develop the part of speech tagging.

Part-of-speech tagging or POS tagging for short. It is the process of assigning a part-of-speech tag like noun, verb, pronoun, preposition, adverb, adjective or other lexical class markers to each word in a text. POS tagging is not useful by itself but it is generally accepted to be the first step to understanding a natural language. Most other tasks and applications heavily depend on it. In addition to that, POS tagging is seen as a prototype problem because any NLP problem can be reduced to a tagging problem. For example, machine translation can be seen as the tagging of words in a given language by words of another language; speech recognition can be seen as the tagging of signals by letters and so on. In general, the input-output relationship can be as complex as sequences, sets, trees and others that can be imagined [10, 19].

The purpose of tagging is to give the computer as much information and knowledge as necessary to enable it to assign each word the correct tag as used in the given context. So thus part of speech tagging is necessary for Tigrigna and other languages.

Ethiopia is a multi-lingual country which includes more than 85 nations and nationalities with different speaking and morphologically, semantically, syntactically and lexically different languages. Tigrinya is a member of the Ethiopic branch of Semitic languages with about 6 million speakers mainly in the Tigre region of Ethiopia and in Central Eritrea. There are also large immigrant communities of Tigrinya speakers in Sudan, Saudi Arabia, the USA, Germany, Italy the UK, Canada and Sweden, as well as in other countries. Tigrinya is written with a version of the **Ge'ez script** and first appeared in writing during the 13th century in a text on the local laws for the district of Logosarda in southern Eritrea [9, 18].The Tigrigna language has its own distinct way of grammar construction, character representation called ፊደል (Fidel) and sentence formation as revealed by [1][9][18]. This paper applies to the new state of art to part of speech tagging method and tests their effectiveness of the artificial neural network method with the previous work of methods done for the Tigrigna part of speech tagging, morphological rich language.

The main advantage of part of speech tagging is to solve the ambiguity of different languages by assigning the proper tag. So Tigrinya is ambiguous like other languages so developing the best part of speech tagger will solve such kind of ambiguity problem.

In Tigrigna there are a lot of complexities of word categorization. For example let us see the following two sentences. Example 1:

በሪሁ ናይ ካልኣይ ክፍሊ ተምሃራይ እዩ:: /**berihu** Nay Kal^ay kfli temhaRay ^Yu / Berihu is grade two student.

ናይ ገዛና መብርሃቲ/ኣምፖል በሪሁ:: /Nay GeZaNa mebrhati/^ampol **berihu** /our house bulb is switched on.

In the first example the word Tigrigna በሪሁ "Berihu" becomes in a noun and verb condition based on the sentences given. Most of the time in Tigrigna noun words is become on the first position of the sentences whereas the verb becomes at the end of the sentences. So in the first example and in the first sentences the word "በሪሁ" indicates a noun which can be shows the name of a person whose grade is grade two. But in the second sentences the word "በሪሁ"is indicate a verb which shows the bulb is switched on. In this example the ambiguity is based on the word category.

1.1. Statement of the Problem

There are different researches done on human language technologies especially for the foreign languages. But these technologies are not sufficient because of the ongoing and novel idea are creating by different researchers. Though for the local languages or under resourced languages are not sufficiently implemented such technologies even if there are different trial on the language processing for these languages like Amharic, Oromo, Tigrigna and so on. Part of speech tagging for Tigrigna language is done by Teklay Gebregzabiher using rule based, hidden Markov model and hybrid methods with comparable performance for each algorithms. This is the first paper on part of speech tagging by comparing the three algorithms performance. He was done well part of speech tagger for Tigrigna language using the three methods.

But, using the previous methods that was done for part of speech Tigrigna has different problems based on the nature of the methods. Rule based approach uses rules to disambiguate of words the rules are based on knowledge of the specific languages which may consists of a large number of morphological, lexical and syntactical information [1, 10, 15]. These rules are obtained manually that are handcrafted by linguistic professionals or through machine learning.

So using rule based approach requires manual work experts so a huge work and cost is requires. And it is tedious and time consuming, not easy to obtain high coverage of the linguistic knowledge and some changes may be hard to accommodate. Stochastic approach also called statistical approach is based on a probabilistic pattern to assign a probable part of speech tag to a given text from a given training text corpus. Stochastic approach (Hidden Markova Model) requires less work and cost than the rule based approach. And the tag to the unknown words cannot be found. Using artificial neural network is fast, tolerant of imperfect data.

Another work on part of speech tagger for Tigrigna was done by Mulgeta Atsbeha using hybrid approach (rule based and average perceptron neural network). This algorithm is used for classification problem. Average perceptron model consists of feature functions paired with feature weight which is estimated during training. Average perceptron algorithm considers one example at a time instead of considering the entire data at once during training. Second it is error-driven, which means as long as it is doing very well, it doesn’t bother updating its parameters.

Part of speech tagging for Tigrigna using artificial neural network with back propagation algorithm is new idea for the language. And also using this method is solves the problems with rule based and stochastic approaches.

The absence or limited of part of speech tagging for Tigrigna limits researches on machine translation, speech recognition, information retrieval, information extraction, corpus analysis and text to speech synthesis system. Hence, conducting research and developing an automatic Part of Speech Tagger for Tigrigna language worth paramount significance.

1.2. Objectives

The general objective of the study is to propose and design part of speech tagging for Tigrigna using artificial neural network and analyzing the performance of the method.

To achieve the general objective of the study the following specific objectives are accomplished.

- ✓ To review, analyze and study the basic word category of the Tigrigna language with the aim of customizing the POS tags done on [1].
- ✓ To review different researches done around the part of speech tagging for local and nonlocal languages to analyze what the difference among part of speech tagging methods so that used as an input for this study.
- ✓ To analyze and study the morphological property of Tigrigna language to perform the tagging process.
- ✓ To develop and design the model for part of speech tagger using the artificial neural network
- ✓ To test and analyze the performance of the algorithm
- ✓ To put the conclusion and recommendation based upon the experimental result

1.3. Methods and Methodology

1.3.1. Literature Review

For this research we have reviewed different related literatures that were help to analyze and model of the part of speech tagging for Tigrigna language. And also uses to understand the morphological and grammatical structure of the language. To understand the different algorithms and methods used for part of speech tagging.

1.3.2. Data Source and Preparation

Data needed for Parts of speech tagging are different sentences with different phonemes. The data set corpus is prepared from different sources such as fictions of Tigrigna, student textbooks of preparatory and secondary schools, Tigrigna magazines and grammars of Tigrigna books and from prepared corpus of Tigrigna. The corpus is tagged by linguistics for the purpose of training and testing data. And the part of speech tagsets are customized from the work of [1].

1.3.3. Modeling Techniques

For this research we introduce a new idea or method for part of speech tagging for Tigrigna using the artificial neural network with back propagation algorithm. During training the neural network we must consider what value to place on what feature. Because neural network is working using a numeric value as an input and gives an output as numeric value. The features are extracted from the tagged corpus and pre-defined tagsets. Mainly in this thesis we consider two features those are the POS information and POS and word information. During designing the neural tagger we must consider how to deciding network topology, deciding the learning algorithm and encoding schema. Words, tags, and associated tags are very essential information in POS tagging process. Each word, W, from the corpus is encoded as n-element vector $W = (P_1, P_2 \dots P_m \dots P_n)$ where n corresponds to the total number of tags and P_m is the prior probabilities (lexical probabilities) that the word W corresponds to the tag T_m. P_m is estimated $P = (Word/tag)$.

Multi-layer feed forward networks, also known as multi-layer perceptron (MLPs) consist of a certain number of neurons which are split into several disjoint sets, so called layers. Every MLP contains an input layer, to which the input vector is fed, and an output layer, which produces the vector of output values. Back-propagation is a supervised learning algorithm and is mainly used by Multi-Layer-Perceptron to change the weights connected to the network's hidden neuron layers. The back-propagation uses a computed output error to change the weight values in backward direction.

1.4. Scope of the Study

The scope of this paper is to design part of speech tagger system for Tigrigna using artificial neural network which can be tag the word class category only. The corpus is collected from different resources so that used for training and testing the system.

1.5. Application of Results

Part of speech tagging is usually the first step in linguistic analysis and also it is a very important intermediate step to build many natural language processing applications. It could be used in machine translation, spell checking and correcting, speech recognition, information retrieval, information extraction, corpus analysis, syntactic parsing and text-to-speech synthesis systems [1] [16] [17].

1.6. Organization of the Thesis

The rest of the thesis covers a broad range of topics that are organized in six chapters. **Chapter two** presents literature review on part-of-speech, importance of POS tagging, and various approaches; like statistical, rule-based, artificial neural network and hybrid, with their advantages and limitations. And also we review about components, activation function, and architecture, types and learning method of artificial neural network. As well as on related works, specifically on Amharic language, Tigrigna, Arabic, Affan Oromo, English and Hindi part-of-speech tagger that had been done in different POS approaches are also addressed in this chapter. Besides, **chapter three** sets up the classification mechanism of Tigrigna word categories which are important for building the POS system a more detailed description of tag sets which are necessary and important for the design of the POS tagger. After the part-of-speech and tag sets for Tigrigna language are identified, the next step is to design Tigrigna POS tagger. **Chapter four** discusses in detail the design process, algorithm and architecture of the Tigrigna POS system. Design goals, approaches, and techniques are also included in this chapter. **Chapter five** presents the data collection process and procedures of the experiments. Training and testing procedures are also given attention. The **last chapter** gives about the conclusion and recommendation of the thesis.

Chapter 2

2. Literature Review and Related Works

2.1. Introduction

Words are divided into different classes called parts of speech (POS), word classes, morphological classes, or lexical tags. Part-of-speech tagging (POS tagging or POST), also called grammatical tagging, is the process of marking up the words in a text as corresponding to a particular part of speech, based on both its definition, as well as its context i.e., relationship with adjacent and related words in a phrase, sentence, or paragraph [21]. Part-of-speech (POS) tagging is often considered as the first phase of a more complex natural language processing application.

POS tagging can be handled either at word-level or sentence-level. At word-level POS tagging is posed as a classification problem in which an appropriate tag for a word is found whereas at sentence level a series of tags corresponding to the sequence of words are obtained [12]. The task of POS-tagging is attaching appropriate grammatical or morpho-syntactical category labels to every token, and even to punctuation marks, symbols, abbreviations,… etc. in a corpus [17]. Part of speech tagging is labeling or assigning of part of speech tag like noun, verb, and adjective… etc to words that are found in the corpus. For example if we consider the following sentence in Tigrigna language.

አበበ/NN **ትማሊ**/ADV **ካብ**/PREP **ጅማ**/NN **መፂኡ**/VB::/PUNC ‘Abebe came yesterday from jimma.’

The following sentence represents the translated sentence to Latin word that we present in the above example.

Abebe/NN Tmali/ADV Kab/PREP Jma/NN Metsiu/VB። /PUNC

In the above example the sentence is correctly tagged with the appropriate part of speech tagging to every tokens with the right grammatical and punctuation mark. The lexical categories that are represented to the above sentence of the Tigrigna language are noun, adverb, preposition, verb and punctuation. The symbols NN, ADV, PREP, VB and PUNC represents for noun, adverb, preposition, verb and punctuation respectively.

The process of tagging takes a sentence as input, assigns a POS tag to the word or to each word in a sentence or in a corpus, and produces the tagged text as output [20].

In principle, any collection of more than one text can be called a corpus, (corpus being Latin for "body", hence a corpus is anybody of text) [22]. A corpus, plural corpora, is a special collection of textual material collected according to a certain set of criteria. The criteria main contain such as sampling and representativeness, finite size, machine-readable form, a standard reference [22].

A corpus can be categorized as un-annotated and annotated corpus. Un-annotated corpus is a collection of raw texts that does not contain additional linguistic information or labels on words. For instance, the following sentence is un-annotated or raw texts.

አበበ ናይ ካልኣይ ክፍሊ ተምሃራይ እዩ ፡፡ /Abebe NaY Kalay kfli temharay eyu/Abebe is grade two student. In this example there is no any additional information about the sentence. In this sentence example there is no any part of speech tag identification that distinguishes which one is noun, verb, adverb etc… and the following example is a sentence which can represent an example of annotated corpus.

አበበ/NN **ትማሊ**/ADV **ካብ**/PREP **ጅማ**/NN **መጺኡ**/VB ::/PUNC Abebe/NN Tmali/ADV Kab/PREP Jma/NN Metsiu/VB ።/PUNC ‘Abebe came yesterday from jimma.’

The above example is representing an annotated corpus because there is representation of word class to each word to assign the appropriate part of speech tag. In the above example the word **አበበ**/NN or Abebe/NN is indicated that it is a noun. So every word are represented with the correct or appropriate word class so any person can easily identify or understand the part of speech tag of words.

Annotated corpus has many applications in natural language processing. It enables to retrieve the frequency lists and indexes of various words or other structures of language within it. Since corpus is available with explicitly linguistics information, data retrieval from the corpus can be easier and more than with un-annotated data. It provides the most reliable source of data on language as it is actually used because it is naturalistic collection of data. It is an essential tool to develop part-of-speech taggers and parsers [20]. The annotation of corpus is done either manually or automatically. Manually annotating a corpus is more expensive, time talking and needs fair amount of handworks.

There are so many researches done on different part of speech tagging approaches. The most tagging algorithms fall into one of two classes: rule-based taggers and stochastic taggers. Rule-based taggers generally involve a large database of hand-written disambiguation rule which specify, for example, that an ambiguous word is a noun rather than a verb if it follows a determiner [21]. The part of speech tagging approaches are summarized as the following diagram represents.

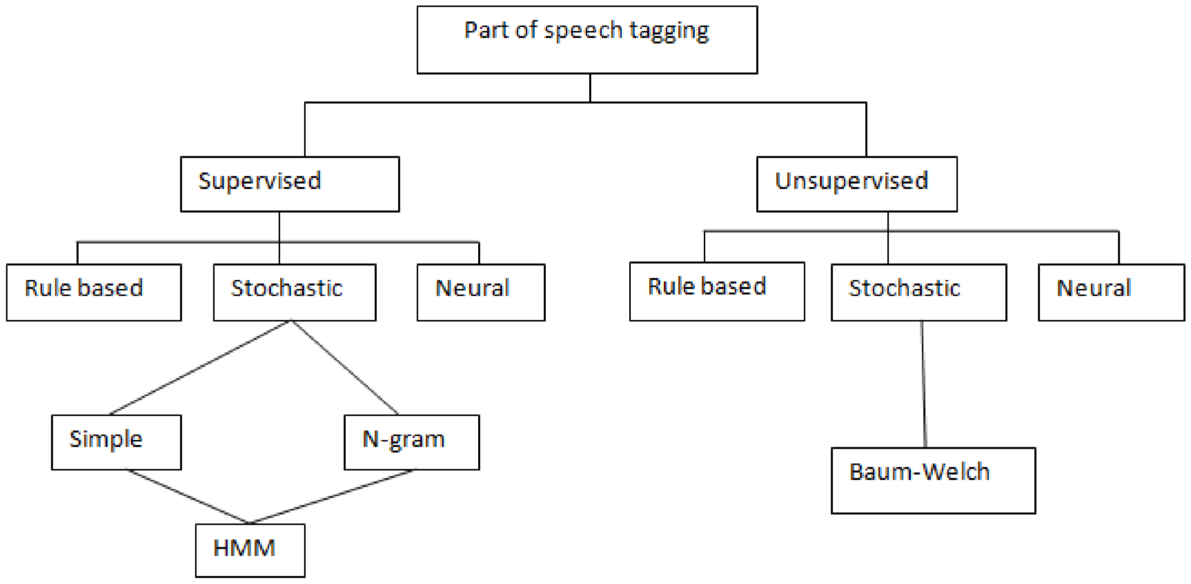

Figure 1.1 The Common Methods for the POSTaggers [17]

Rule based approach is based on the linguistic knowledge during the tagging process. Words are tagged based on the contextual information of the word by observing the right and left side of the words of sentences that are going to be tagged. The contextual of the word can be gained from the annotated corpora-dictionaries. Dictionaries provide a list of word with their lexical meanings. These rules could determine, for example, that a word following a determiner and an adjective must be a noun. In this approach, what researchers should do is set and check rules properly with the help of language professionals or to make the taggers learn rule during the training phase of the system as it is in the case of the Brill tagger [1].

Stochastic approach is based on a probabilistic pattern to assign a probable part of speech tag to a given text from a given training text corpus. The stochastic approach can be divided in to supervised and unsupervised tagging techniques. Supervised statistical tagging techniques are used tagged corpus for their training though it requires large amount of tagged data so that high level of accuracy can be achieved. Unsupervised statistical technique which do not require pre tagged corpus but instead use sophisticated computational methods to automatically induce word groupings (i.e. tag set) [1].

According to different researchers there are many automatic part of speech annotators for different language. However, these automatic annotators are not useful to tag for Tigrigna language. This paper explores the use of artificial neural networks (ANNs) for the problem of part of speech tagging for Tigrigna language. Using of ANNs is a new approach in Tigrigna natural language processing. This chapter reviews the artificial neural network technique or approach used for part of speech tagging of Tigrigna language.

2.2. The Human Brain

The human brain consists of a large number (more than a billion) of neural cells that process information. Each cell works like a simple processor and only the massive interaction between all cells and their parallel processing makes the brains abilities possible. The neuron sends out spikes of electricity activity through a long, thin strand known as axon**,** which split into thousands of branches.

As the figure indicates, a neuron consists of a core, dendrites for incoming information and an axon with dendrites for outgoing information that is passed to connected neurons. Information is transported between neurons in form of electrical stimulations along the dendrites.

Incoming information’s that reach the neuron's dendrites is added up and then delivered along the neuron's axon to the dendrites at its end, where the information is passed to other neurons if the stimulation has exceeded a certain threshold. In this case, the neuron is said to be activated. If the incoming stimulation had been too low, the information will not be transported any further. In this case, the neuron is said to be inhibited.

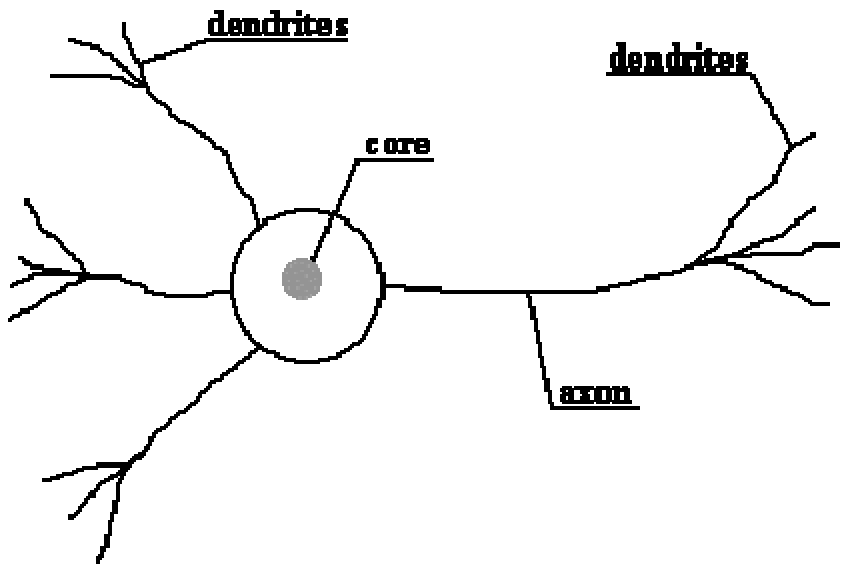

Figure 2.2 Structure of a neural cell in the human brain [24, 26]

2.3. Artificial Neural Network (ANN)

Artificial Neural Networks (ANNs) model is an information processing paradigm that is inspired by the way biological nervous systems. A neural network is interconnected set of simple units or nodes, whose functionality is based on biological neuron found in human brain. They are also called connectionist models [17, 25]. A neural network is an artificial representation of the human brain that tries to simulate the learning process. The term “artificial” means that neural networks are implemented in computer programs that are able to handle the large number of necessary calculations during the learning process [24].

“A neural network is a massively parallel distributed processor made up of simple processing units. Which has a natural property for storing experimental knowledge and making it available for use, it resemble human brain in two respects [1, 26, 28]:

1. Knowledge is acquired by the network from its environment through a learning process.

2. Interneuron connection strength known as synaptic weights is used to store the acquired knowledge.”

According [1, 29] *A neural network is a system composed of many simple processing elements operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes.*

Every artificial neural network has three layers. These layers are input layer, hidden layer and output layer. The input layer receives input data from an external source, the output layer transmit the result of the neural network processing, and the hidden layer provides the internal relations between input and output layers [17].

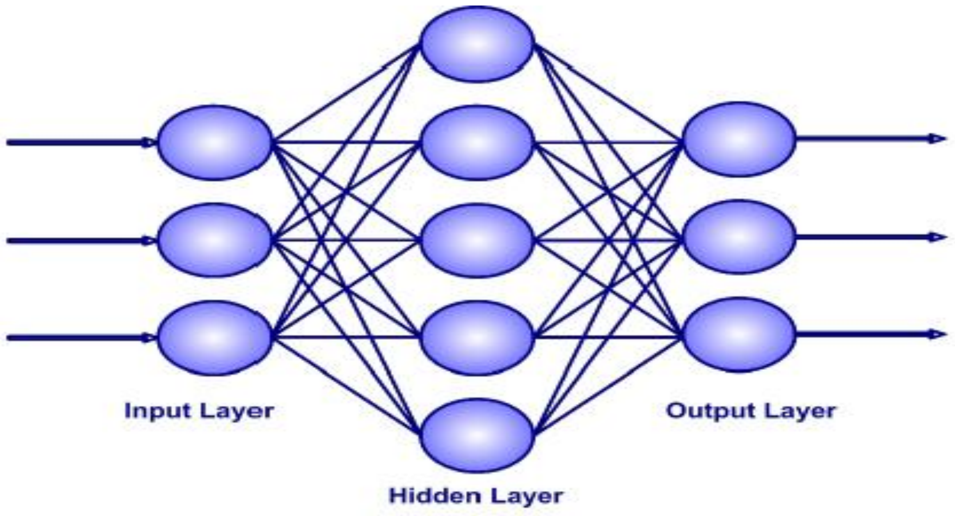

Figure 2.3 The artificial neural network [17]

When creating a functional model of the biological neuron, there are three components of importance. First, the synapses of the neuron are modeled as weights. The strength of the connection between an input and a neuron is noted by the value of the weight. The next two components model the actual activity within the neuron cell. The summation function adds all the input values with the multiplication of the appropriate weight. This activity is referred to linear combination. Finally, an activation function controls the amplitude of the output of the neuron. An acceptable range of output is usually between 0 and 1, or -1 and 1 [26].

If all values of the input pattern are zero, the weights in the first weight matrix would never be changed for this pattern and the network could not be learn it. Due to this fact, a pseudo input which is called bias is added to the network to learn the network if the input patterns are zero and that has a constant value 1. The bias is added to the hidden layer and the output layers, and layers have initial values and are changed in the same way as the other weights. By sending a constant output 1 to following neurons it is guaranteed that the input values of those neurons are different from zero.

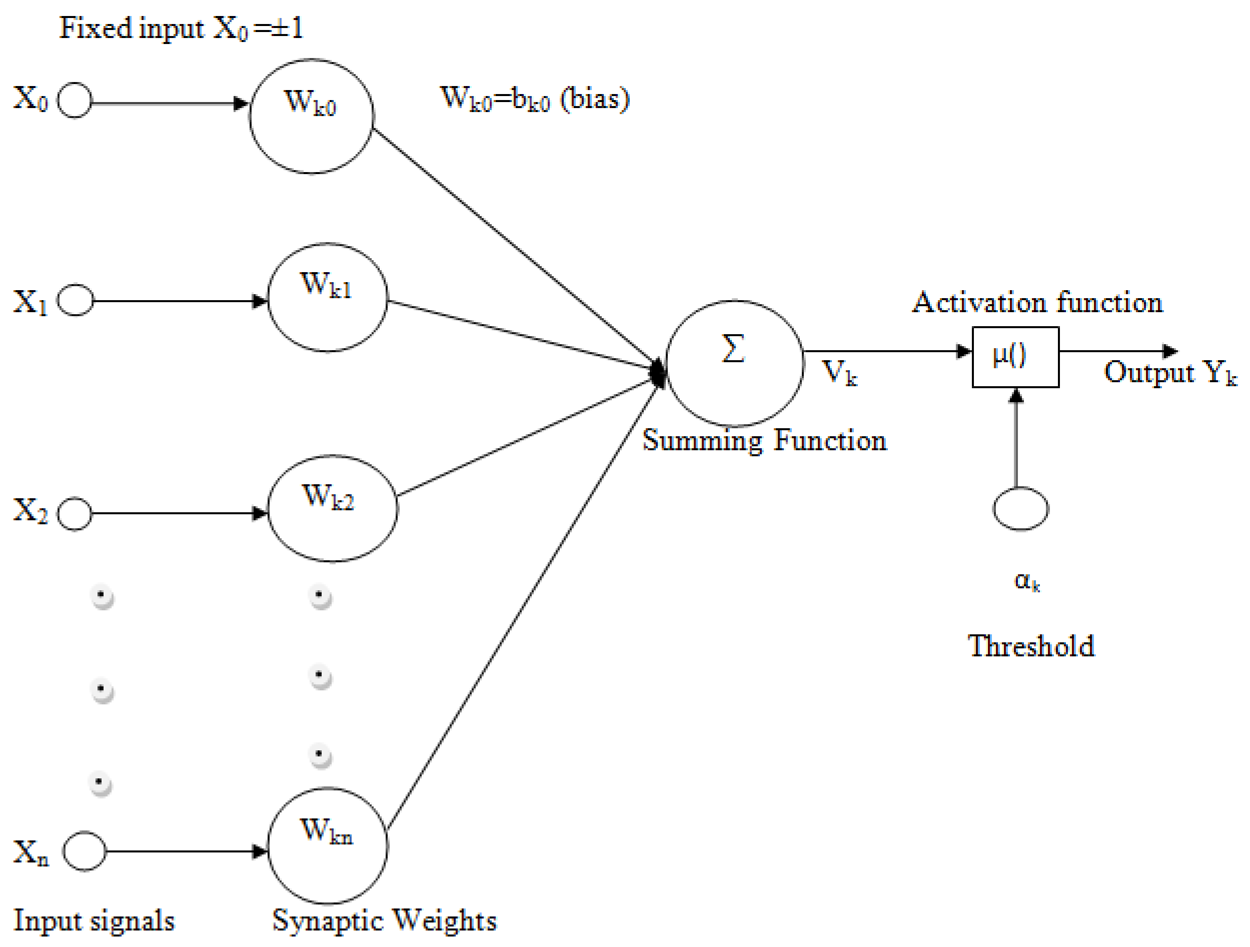

Figure 2.4 Artificial neuron models

Artificial neural network have the following elements:

- ✓ A number of input values $(X_1, X_2, X_3 \cdots X_n)$
- ✓ Weights which indicate the degree of the input value on the processing unit $(W_{k1}, W_{k2}, W_{k3,\ldots} W_{kn})$.
- ✓ A summation function that produces a weighted sum of the input values$(X_0W_{k0} + X_1W_{k1} + X_2W_{k2} + X_3W_{k3} + \cdots + X_nW_{kn})$
- ✓ A threshold value that determines whether a signal should be transmitted or not
- ✓ The output signal/values $(Y_{1,}Y_{2,}Y_{3,} \cdots Y_n)$

The summation function of the input value with the multiplication of the appropriate weight is expressed as follows:

$$V_k = \sum = X_0W_{k0} + X_1W_{k1} + X_2W_{k2} + X_3W_{k3} + \cdots + X_nW_{kn}$$

From the above equation $X_0W_{k0}$is the bias that could be added to the hidden layer and output layer. The output of the V_k value can be expressed as follows

$$Vk = \sum_{j=0}^{n} X_j W_{kj}$$

The overall output of the neural network can be calculated or Y_k can be expressed as follows: $Y_k$can be expressed according to the threshold value or activation that could be applied to the neural network. So the Y_k can be:

$$Y_{k=} \begin{cases} 1 & if\ V_{k \geq \alpha_k} \\ 0 & Otherwise \end{cases}$$

Whereα_k, is the threshold value that uses to activate the neuron if it satisfies the above equation, unless it cannot pass to the next neuron to process the learning process, and this threshold value is called linear threshold value [25].

2.3.1. Component of Neural Network

There are many different types of neural nets, but they all have nearly the same components. If one wants to simulate the human brain using a neural net, it is obviously that some drastic simplifications have to be made: First of all, it is impossible to "copy" the true parallel processing of all neural cells. Although there are computers that have the ability of parallel processing, the large number of processors that would be necessary to realize it can't be afforded by today's hardware. Another limitation is that a computer's internal structure can't be changed while performing any tasks [24].

These facts lead to an idealized model for simulation purposes. Like the human brain, a neural network also consists of neurons and connections between them. The neurons are transporting incoming information on their outgoing connections to other neurons. In neural net terms these connections are called weights. The "electrical" information is simulated with specific values stored in those weights. By simply changing these weight values the changing of the connection structure can also be simulated [24].

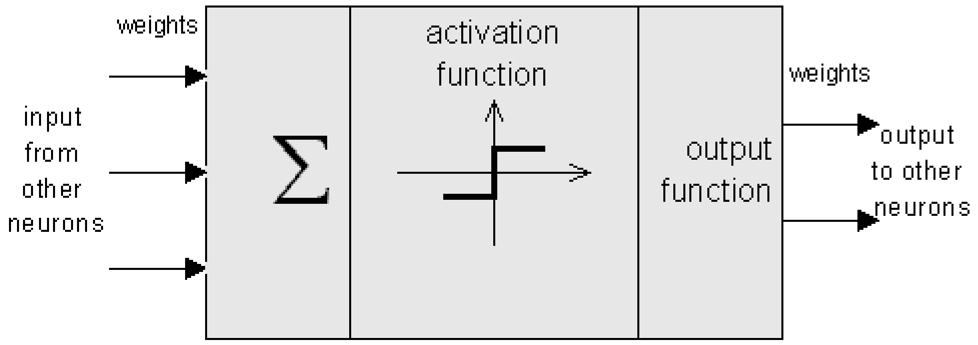

Figure 2.5 INNN and SNNC [24]

As you can see, an artificial neuron looks similar to a biological neural cell. And it works in the same way. Information (called the input) is sent to the neuron on its incoming weights. This input is processed by a propagation function that adds up the values of all incoming weights. The resulting value is compared with a certain threshold value by the neuron's activation function. If the input exceeds the threshold value, the neuron will be activated, otherwise it will be inhibited. If activated, the neuron sends an output on its outgoing weights to all connected neurons and so on. In a neural net, the neurons are grouped in layers, called neuron layers. Usually each neuron of one layer is connected to all neurons of the preceding and the following layer (except the input layer and the output layer of the net). The information given to a neural net is propagated layer-by-layer from input layer to output layer through none, one or more hidden layers. Depending on the learning algorithm, it is also possible that information is propagated backwards through the net [24].

According to [26, 31] there are seven types major components which make up an artificial neuron. These components are valid whether the neuron is used for input, output, or is in one of the hidden layers.

2.3.1.1. Weighting Factors

A neuron usually receives many simultaneous inputs. Each input has its own relative weight which gives the input the impact that it needs on the processing element's summation function. Weights are adaptive coefficients within the network that determine the intensity of the input signal as registered by the artificial neuron. They are a measure of an input's connection strength. These strengths can be modified in response to various training sets and according to a network’s specific topology or through its learning rules.

2.3.1.2. Summation Function

The first step in processing element's operation is to compute the weighted sum of all of the inputs. Mathematically, the inputs and the corresponding weight are vectors which can be represented as $(X_1, X_2 \cdots X_m)$ and $(W_1, W_2 \cdots W_m)$.The total input signal is the dot, or inner product of these two vectors. This simplistic summation function is found by multiplying each component of the X vector by the corresponding component of the W vector and then adding up all the products. $Input_1 = X_1 * W_1, Input_2 = X_2 * W_2 \cdots$Are added as $Input_1 + Input_2 + \cdots + Input_m$ then the result is a single number, not a multi-element vector.

2.3.1.3. Transfer Function

In transfer function, the sum total can be compared with some threshold to determine the neural input. If the sum is greater than the threshold value, the processing elements generate a signal. If the sum is less than the threshold, no signal is generated. The threshold, or transfer function, is generally non-linear. Linear (straight-line) functions are limited because the output is simply proportional to the input. Linear functions are not very useful.

2.3.1.4. Scaling and Limiting

After the processing element's transfer function, the result can pass through additional processes which scale and limit. This scaling simply multiplies a scale factor times the transfer value, and then adds an offset. Limiting is the mechanism which insures that the scaled result does not exceed an upper or lower bound. This limiting is in addition to the hard limits that the original transfer function may have performed.

2.3.1.5. Output Function (Competition)

Each processing element is allowed to output one signal, which it may send to hundreds of other neurons. This is just like the biological neuron, where there are many inputs and only one output. Normally, the output is directly equivalent to the transfer function's result. Some network topologies, however, modify the transfer result to incorporate competition among neighboring processing elements. Neurons are allowed to compete with each other, inhibiting processing elements unless they have great strength. Competition can occur at one or both of two levels. First, competition determines which artificial neuron will be active, or provide an output.

Second, competitive inputs help to determine which processing elements will participate in the learning or adaptation process.

2.3.1.6. Error Function and Back Propagated Value

In most learning networks, the difference between the current output and the desired output is calculated .This raw error is then transferred by the error function to match the particular network architecture. The artificial neuron’s error is then typically propagated into learning functions of another processing element. This error term is sometimes called the current error. The current error is typically propagated backward to a previous layer. Yet, this back propagated value can be either the current error, the current error scaled in some manner or some desired output depending on the network type. Normally, this back-propagated value, after being scaled by the learning function, is multiplied against each of the incoming connection weight to modify them before the next learning cycle.

2.3.1.7. Learning Function

The purpose of the learning function is to modify the variable connection weights on the inputs of each processing elements according to some neural based algorithms. This process of changing the weights of the input connections to achieve some desired result could also be called adaptation functions, as well as learning mode.

2.3.2. Activation Function

The output of neuron depends on the type of activation function used. There are four most commonly used activation functions in neural network. These are linear, hard limit, sigmoid and tan-sigmoid; these functions are classified in two according to [25, 32, 34], discrete and continuous.

2.3.2.1. Discrete Activation Function

I. Linear Transfer Function

The linear transfer function reproduces the input fed into it as the output itself.

$$F\,(x) = \begin{Bmatrix} 1, x > b \\ x, a < x < b \\ -1, x < a \end{Bmatrix}$$

II. Hard-limit Transfer Function

Hard-limit activation function gives the output 0 if the total summed output is less than 0, and 1 if greater than 0. Mathematically, hard-limit transfer function is defined as [32, 33, 34]:

$$F\,(x) = \begin{Bmatrix} 1, & x \geq 0 \\ 0, & x < 0 \end{Bmatrix}$$

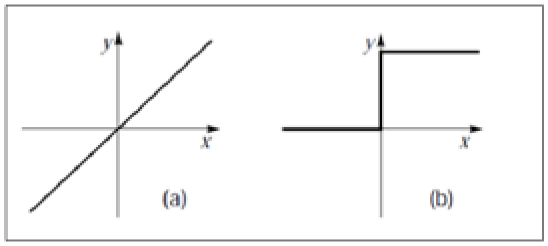

Figure 2.6 Threshold functions a) Linear b) Hard-limit [32]

2.3.2.2. Continuous Activations Function

I. Log-Sigmoid Transfer Function

Sigmoid function is differentiable and its value lies between 0 and 1 according to the below expression. It is used in learning algorithms like back propagation learning and the like.

$$\varphi(\mathrm{v_k}) = \frac{1}{(1+\mathrm{e}^{-\mathrm{v_k}})}$$

II. Tan-Sigmoid Transfer Function

This transfer function takes the inputs and which have the output values between plus and mines of infinity, and limits between -1 and 1 values.

$$\varphi(v_k) = \tan h(v_k)$$

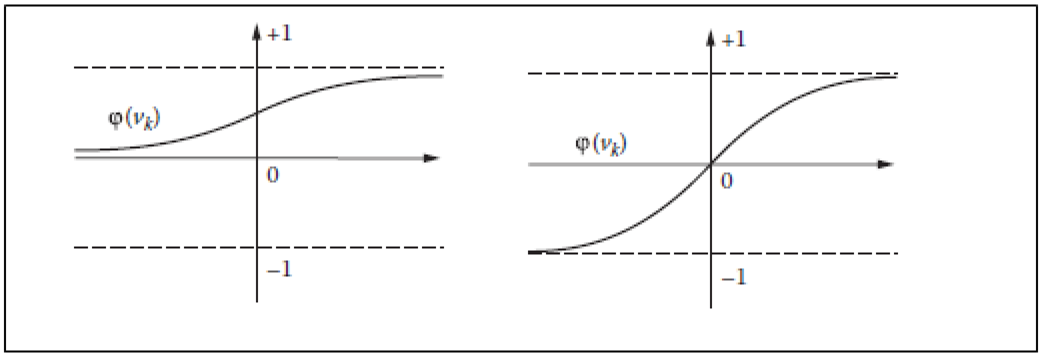

Figure 2.7 Log-Sigmoid Function (left) Tan-Sigmoid Function (right) [32]

2.3.3. Architecture of Neural Network

Architecture refers to the connectivity of each neuron node in the neural network design. Depending on the interconnectivity of the neural network the architecture of neural network can be divided in to three [34].

I. Feed-forward Networks

This architecture does not contain feed-back to the previous network node. The flow takes place from one layer to the next. Generally in the feed-forward networks the flow of information is in the forward direction.

II. Feed-Back Networks

In this kind of network, the output of a neuron is either directly or indirectly fed back to its input via other linked neurons. This kind of network is frequently used in complex pattern recognition tasks, e.g., speech recognition etc.

III. Lateral Networks

In this kind of network, there exist couplings of neurons within one layer. There is no essentially explicit feedback path amongst the different layers. This can be thought of as a compromise between the forward and feedback network [25].

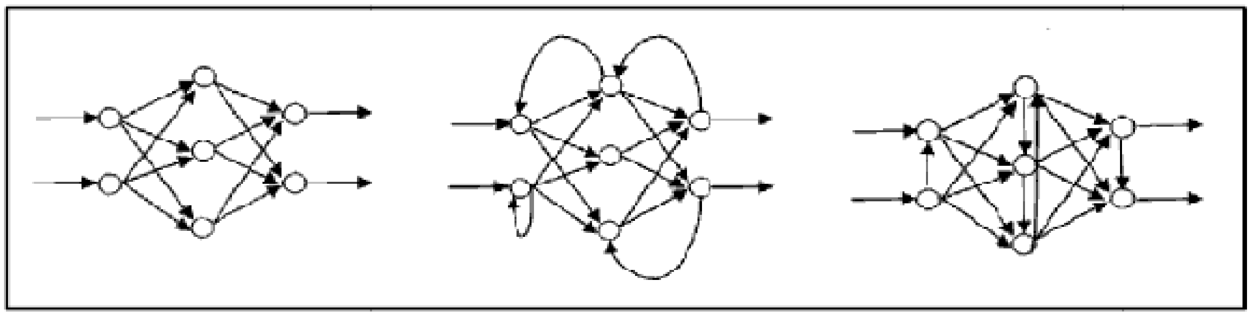

Figure 2.8 Feed-Forward (left) Feed-Back (middle) Lateral network (right)

2.3.4. Types of Neural Networks

There are different most commonly used types of neural networks. Some of them are single layer perceptron (perceptron), multi-layer perceptron, back propagation network, radial basis function network, hopfiled network, adaline, kohonen network. These types can be distinguished by their type of architecture, structure and the type of learning algorithm they use.

2.3.4.1. Single Layer Perceptron

The basic constituent unit of most neural networks is called a perceptron or simply an (artificial) neuron. Perceptron is the simplest single layer neural network whose weight and bias could be trained to produce the target vector in corresponding to their input value.

In the perceptron model, each external input is weighted with an appropriate weight, and the sum of the weighted inputs is sent to the linear transfer function, which also has an input of unity as the bias. The linear transfer function is used to return output as a 0 or 1. The perceptron neuron produces the output 1 if the net input into the transfer function is equal to or greater than 0; otherwise it produces 0 [34].

$$f(x) = \begin{cases} 1 & x \geq 0 \\ 0 & otherwise \end{cases}$$

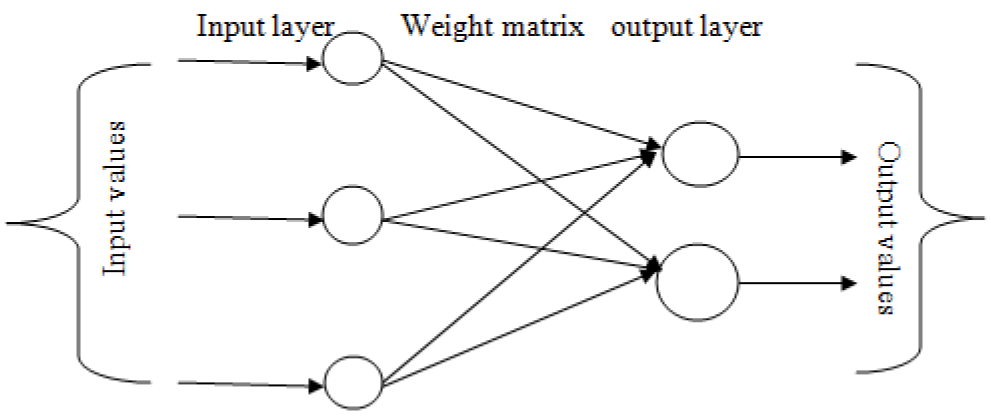

Figure 2.9 perceptron neurons model

The type of architecture of the single neural network is feed-forward and contains one input layer and one output layer. In this neural network it also uses an activation function of hard limiter with the learning mechanism of supervised and an algorithm of hepp learning rule.

2.3.4.2. Multilayer Perceptron

Single layer neural network cannot overcome the real world problems. To solve such kinds of limitations multilayer neural network is proposed by different researchers. The real problem of perceptron neural network is it only solves problems that are linearly separable. But most problems of the real world are non-linear in nature so that this problem opens to the new idea of multilayer perceptron neural network [24, 34]. Multilayer neural network contains one input layer, one or more hidden layers and with one output layer.

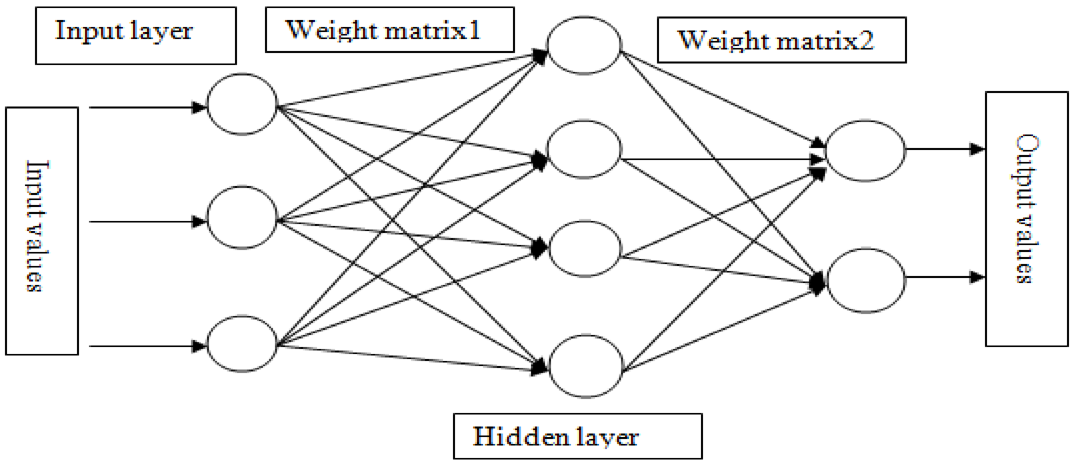

Figure 2.10 multilayer perceptron networks

Multilayer neural network (MLP) is a type of feed-forward architecture which have an input value of binary and it's activation function could be hard-limiter or sigmoid. It is also have a learning algorithm mechanism like delta learning rule or back-propagation.

According [24, 34] the basic multilayer perceptron algorithm could be summarized as follows:

1. Initialize the network's weights with a random value between +1 and -1
2. Present the first training pattern (binary values) to the input network layers
3. Activate each neuron of the following layer:
   I. Multiply the weight values of the connections leading to this neuron with the output values of the preceding neurons
   II. Add up these values
   III. Pass the result to an activation function, which computes the output value of this neuron
4. Repeat this until the output layer is reached
5. Compare the calculated output pattern to the desired target pattern and compute an error value
6. Change all the weights by adding the error value to the (old) weight values
7. Go to step 2
8. The algorithm ends, if all output patterns match their target patterns

2.3.4.3. Back Propagation Neural Network

The back propagation neural network involves multilayer neural network with a back propagation algorithm.

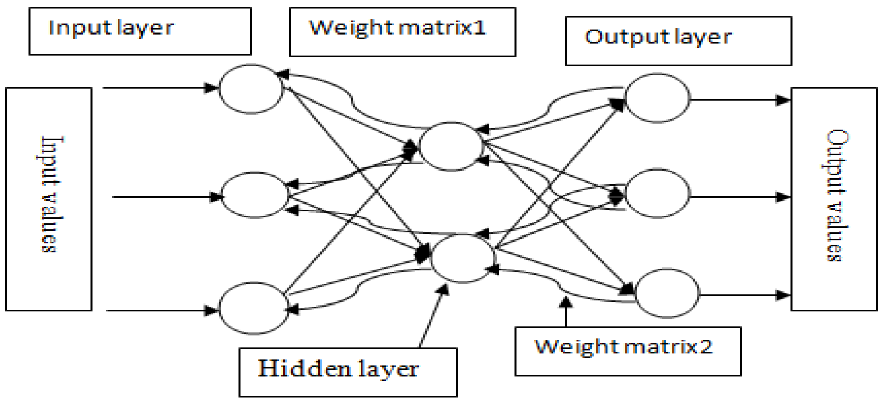

Figure 2.11 Sample structure back-propagation neural networks

Back-propagation neural network is type of feed-forward architecture and contains neural layer of one input layer, one or more hidden layer and one output layer. Its input value is binary with learning algorithm of back-propagation algorithm.

2.3.5. Learning in Artificial Neural Network (ANN)

The most important characteristic of ANN is the ability to learn from data. Artificial neural networks work through the optimized weight values. The method by which the optimized weight values are attained is called learning. In the process of learning we present the neural network with input and output data try to teach the network how to produce an output with the corresponding input values. Because of this the neural network adapt to new environment that don't know before. When learning is complete, the trained neural network, with the updated optimal weights, should be able to produce the output within desired accuracy corresponding to an input pattern [24, 34].

The algorithms used for learning are classified into three [24, 33, 34], supervised, unsupervised and semi-supervised (reinforcement) learning. Supervised learning means guided learning, i.e. when the network is trained by showing the input and the desired result side by-side. A neural network is said to be supervised if the desired output is already known. Supervised learning, in which a teacher provides output targets for each input pattern, and corrects the network's errors explicitly; Examples of supervised learning algorithm are delta rule, gradient descent, back-propagation etc…

In unsupervised learning the network trains by itself from the data without prior knowledge. During this there is no any expected output or it is not known during the learning process. Here there is no teacher that teaches the system and the network must find regularities in the training data by itself. Hebb's rule is an example of unsupervised learning algorithm [33, 34]. Semi-supervised or reinforcement is not always possible to get input data and associated output learning. In such situations, the agent is given a reward for good response and punished for bad ones [25, 33].

Running the network is consists of forward pass and backward pass. In the forward pass outputs are calculated and compared with desired outputs. Errors from desired and actual output are calculated. In the backward pass this error is used to alter the weights in the network in order to reduce the size of the error. Forward and backward pass are repeated until the error is low enough (users usually set the value of accepted error) [37]. When training a network, we are feeding to the network which has the input value and the desired output.

2.3.6. Back-propagation Algorithm

Back propagation, also known as Error Back propagation or the Generalized Delta Rule, is the most widely used supervised training algorithm for neural networks. Back propagation uses a calculated output error to change the weight error in the backward direction [24, 33]. The idea of back propagation is output of neural network is evaluated against the desired output. If the output of the neural network is not satisfactory connections (weights) are adjusted repeatedly until error is small enough or accepted. This algorithm is the most important to adjust the weights of the network in backward direction so we begin with a full derivation of the learning rule [33].

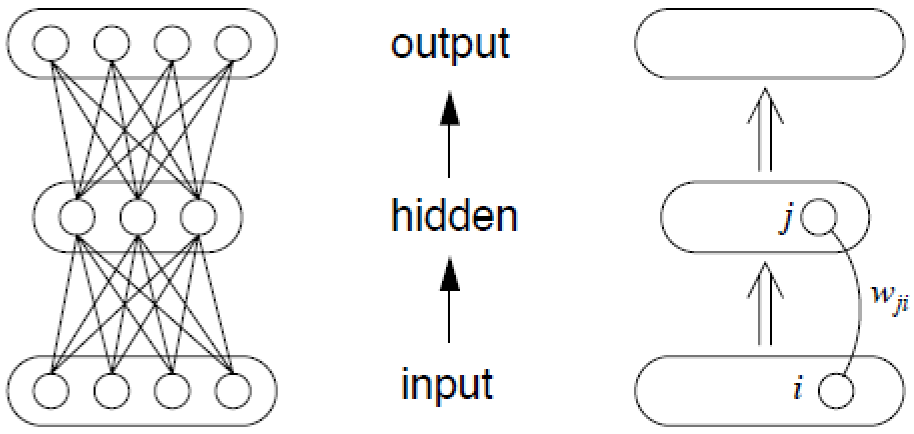

Figure 2.12 A feed forward neural network, highlighting the connection from unit i to unit j.

Suppose the above figure of multilayer neural network which is a feed forward neural type and we want to calculate the desired function from input vector to output vector. Let us define: i, j and k are arbitrary unit indices, O set of output units, h_k is hidden layer input to output layer, p training pattern contains input vector and output target vector, X_j^p Net input to unit j for pattern p, Y_j^p as the activation of unit j for pattern p (expected output), W_{ji} Weight from unit i to unit j, t_j^p target activation for unit j for pattern p (current output) for (j ∈ O), E^p Global output error for training pattern p, E is global error for the entire training set [33].

$$X_j^p = \sum_i W_{ji} Y_j^p$$

Applying this net of neural network to the activation function of sigmoid function

$$Y_j^p = \delta\left(X_j^p\right) = \frac{1}{1 \;+\; e^{X_j^p}}$$

The error of neuron j and the total output error are calculated,

$$E^p = \frac{1}{2}\sum_j \left(Y_j^p - t_j^p\right)^2$$

$$E = \sum_p E^p$$

The basic principle of the back-propagation algorithm is to move error in the backward direction of the neural network architecture, means from the output layer to the hidden layer of the network. So considering this assumption the updated weight of the output layer can be calculated as follows.

For output layer

$$\Delta W_{kj} = \alpha\left(Y_j^p - t_j^p\right)t_j^p\left(1 - t_j^p\right)h_k$$

For hidden layer

$$\Delta W_{ik} = \alpha' \sum\nolimits_j \left(Y_j^p - t_j^p\right)t_p^j\left(1 - t_j^p\right)W_{kj}h_k(1 - h_k)X_i$$

X_i, is input to hidden layer from input layer

The new weights are:

$$W'_{kj} = W_{kj} + \Delta W_{kj}$$

To get the net error of the neural network forward propagation is must done first. While propagating in forward direction, the neurons are being activated using the sigmoid function. The algorithm works as follows [24]:

I. Perform the forward propagation phase for an input pattern and calculate the output error
II. Change all weight values of each weight matrix using the formula

Weight (old) + learning rate * output error * output (neurons i) * output (neurons i + 1) * (1 – output (neurons i + 1))

III. Go to step one

IV. The algorithm ends if all output patterns match their target patterns.

2.4. Related Works

The task of POS-tagging is attaching appropriate grammatical or morpho-syntactical category labels to every token, and even to punctuation marks, symbols, abbreviations, . . . etc. in a corpus. So POS is the first step in NLP that have different important application can be used it. Such as in machine translation, spell checking and correcting, speech recognition, information extraction, information retrieval, corpus analysis and text to speech synthesis system [17]. In this section, researches that are conducted on different approaches of POS tagging for different languages are reviewed. The purpose of this section is to see the development of different POS approaches and take on the crucial tools and implementations for the study.

The first Tigrigna part of speech tagger was done using hybrid (Hidden Markov model and rule based) approaches are done by Teklay Gebregziabher [1]. The hybrid approach was done by combining the hidden Markov model and the rule based approaches. He has used about 24,000 words from around 1000 sentences containing 8000 distinct words were tagged for training and testing purpose. The POS tag-sets used in this research are 36 tag-sets are identified for the tagger to put the appropriate word class. In this research first raw Tigrigna data are tagged using HMM tagger and afterwards using the rule based tagger will be corrected. Viterbi algorithm and Brill Transformation-based Error driven learning are adapted for the HMM and Rule based taggers respectively. He has used both the lexical and contextual probability to find the most probable tag of word. The lexical probability is simply the probability of a word occurrence with a specific tag $\left(P(\mathcal{T}_i|\mathcal{W}_i)\right)$ that can be calculated by dividing the occurrence of the number of appearances of the $\mathcal{W}_i$ and $\mathcal{T}_i$ by the number of occurrences of $\mathcal{W}_i$ in the Text corpus. Contextual probability is the transition probability that can be determined by calculating the probability that the tag occurs with n-previous tags. The researcher has collected data from different sources such as websites of Dimtsi Woyane Tigray, Mekalh Tigray, Woyen Newspaper, FM Mekelle, Tigray Development Association and Ethiopian Radio and Television Agency Tigrigna Department. An experiment of the tagger is done on the 24,000 words for the training and testing purpose. From this amount of word 25% of the corpus was used for the testing purpose. As a result, the experimental performance of this work indicates that 89.13% and 91.8% accuracy for HMM and rule-based tagger, respectively. But the result reaches to 95.88% when the experiment has been conducted on the hybrid tagger.

According to [53] Automatic part of speech tagger for Tigrigna language using hybrid approach (neural network and rule based) was done. He conducted on two steps to accomplish a better performance using average perceptron neural network and followed by rule based for hybrid approach. He was used a total of 22 morpho syntactic tagsets were adapted to prepare the annotated corpus. He collected a corpus containing 3100 sentences, 10000 distinct words and 56,151 total tokens and they are balanced corpus (not a domain specific corpus). The models are trained in 75% of the corpus and tested on the remaining 25% for their robustness and effectiveness. For each model several different experiments have been conducted. In this study state-of-the-art tagging accuracy for morphological rich languages particularly Tigrigna with Averaged perceptron tagger is achieved. The Rule based tagger has found 94.8%, while Averaged perceptron tagger achieved 95.5%. Thus, averaged perceptron tagger and rule based tagger achieved comparable performance; however, the hybrid tagger improves the accuracy to 96.3%. The hybrid tagger works as a sequence of averaged perceptron followed with rule based tagger as error detection and correction sequence. In between the trained averaged perceptron and rule based tagger there is output analyzer with a threshold value as output validation and decision maker.

The first Amharic part of speech tagger was done using hybrid (neural network and rule based) approach by Solomon Asres [38]. The researcher has conducted two steps to accomplish a better performance of tagger. The first step is that the Amharic text is tagged using the Neural Network approach. Afterwards, the second step is, the tagged text is checked for detection and correction of any anomaly using the rule based approach. He has adapted a multilayer perceptron neural network with back-propagation algorithm and transformation based learning method for the development of Amharic tagger. He has used 30 tag-sets and 210,000 words of text corpus. He has used both lexical probability and contextual probability to find the most probable tag of a word. He has used both the lexical and contextual probability to find the most probable tag of word. The lexical probability is simply the probability of a word occurrence with a specific tag

$\left(P(\mathcal{T}_i|\mathcal{W}_i)\right)$ that can be calculated by dividing the occurrence of the number of appearances of the $\mathcal{W}_i$ and $\mathcal{T}_i$ by the number of occurrences of $\mathcal{W}_i$ in the Text corpus. Contextual probability is the transition probability that can be determined by calculating the probability that the tag occurs with n-previous tags. The researcher has collected data from different source such as the Ethiopian Language Research center (ELRC), Addis Ababa. He has collected around 210,000 words from one ELRC project called "The Annotation of Amharic News Document" which is meant to tag each Amharic word in its context with the most appropriate part of speech manually. The project in turn has collected the sentences from Walta Information Center, a private News Agency located in Addis Ababa, Ethiopia, that makes daily news in Amharic and English through its website. To evaluate the proposed method, the researcher has conducted a lot of experiments. Relatively a large number of data is used to train and test the proposed tagger. As a result, the experimental performance of this work indicates that 91% and 94% accuracy for rule-based and neural network tagger, respectively. But the result reaches to 98% when the experiment has been conducted on the hybrid tagger. Though the text corpus taken for this thesis work is not as large size as that of brown corpus etc., it has achieved a higher performance on the hybrid approach.

Part of speech tagging can be used for different applications one of them is used for factor language model [2]. For the purpose of factor language the software like TnT and SVMTool are used to train different taggers. In development of factor language model two important points are considering, the first one is choosing the appropriate factor which can be done based on the linguistic knowledge or using a data driven technique and the second is finding the best statistical model over these factors. The corpus and POS tag set used for the purpose of training and testing of this experiment is from Ethiopian language research center (ELRC) project of the annotation of Amharic news documents (AAND) which contains 210,000 manually tagged each Amharic word in its context. They have developed three TnT based taggers by taking different amount of tokens which are 80%, 90% and 95% of the corpus and the taggers are named as tagger1, tagger2, and tagger3. From the corpus 5% tokens are chosen for the testing purpose of the TnT tagger and for SVMTool for comparable purpose. The overall accuracy of the best performing TnT- and SVM-based taggers is 82.99% and 85.50%, respectively.

According to [3] part of speech tagging for Amharic using conditional random fields was done by implying two tasks the segmentation and part of speech tagging are carried out independently. This aim is to explore recent development in the morphological analysis of related languages, such as Arabic, Hebrew and machine learning approaches and apply them to the Amharic language. The tasks of Amharic word segmentation and part of speech tagging is performed using a small annotated corpus of 1000 words. The given the size of the data and the large number of unknown words in the test corpus (80%), an accuracy of 84% for Amharic word segmentation and 74% for POS tagging.

Another work for Amharic [4] was done by applying the art of part of speech tagging to Amharic using three different tag sets. The taggers are HMM model (TnT), SVMTool, Maximum Entropy (MALLET). The datasets or corpus consists of all 1065 news texts (210,000) words are used from different resources.   It has been morphologically analyzed and manually part of- speech tagged by staff at ELRC, the Ethiopian Languages Research Center at Addis Ababa University. . The best results were obtained using a Maximum Entropy approach, while HMM-based and SVM based taggers got comparable results.

According to [5] is conducted to identify the best method for under resourced and morphologically rich language especially in case of Amharic language. In this research segmentation and tag hypothesis combination have conducted to improve tagging accuracy. Different POS taggers are used for the experiments, Disambig, Moses, CRF++, SVMTool, and MBT (memory based POS tagger generator) and TnT Trigram ‘n’ Tags. The Pos tag-set and the corpus used tokens are 210,000 developed within the AAND at the ELRC has been used for the experiment purpose. The larger the corpus and the higher the accuracy of the training set, the better performance of the tagger. TnT and SVM are affected by the amount of data used in the training and MBT is less affected by the amount of data used in the training. The result justifies that TnT works better with large training data size and MBT is less affected by the size of the data used in the training data set and also segmenting words which are composed of morphemes of different POS and which are assigned compound tags is a mean of improving tagging accuracy for the under resourced and morphologically rich languages. MBT and SVM based taggers have high performance for unknown words depending on the amount of data used in the training data set. The best accuracy was obtained using a Maximum Entropy approach when allowed to create its own folds: 90.1% on a 30 tag tag-set, and 94.6 resp. 94.5% on two reduced sets (11 resp. 10 tags), outperforming an HMM based (TnT) and an SVM-based (SVMTool) tagger.

Another earlier work on Afaan Oromo language part of speech tagger was done by Getachew Mamo in Addis Ababa University in 2009 [20]. In this work, the researcher has used one of the known models, which is the generalization of the stochastic approaches, HMM for tagging Afaan Oromo Texts. He has collected 159 Afaan Oromo sentences (with 1621 distinct words) from different sources and he has used 17 tag-sets to annotate these sentences. He has divided these sentences as training set and test set. The HMM based Afaan Oromo part of speech tagger was trained on the training set in order to compute and store the lexical and contextual probabilities of words in the training. The tagger then takes untagged Afaan Oromo text as an input and tokenizes the sentences into words before actually assigns the part of speech tags sequence. After this, each token in the sentence is assigned with a correct part of speech tag sequence that is done using unigram and bigram models of the Viterbi algorithm by taking the knowledge from lexical and contextual probabilities gained during the training session. The researcher has tested the performance of the tagger by conducting experiments and as a result he has got an accuracy of 87.58% and 91.97% for the unigram and bigram models respectively.

An earlier part of speech tagger using neural network was developed by Helmut Schmid [11]. The developed tagger (Net-tagger) consists of multi-layer perceptron network and lexicons. The structure of the Net-tagger is consists of two layers which means without hidden layer. For each part of speech tag pos_j and each of the word of the previous and subsequent words in the context there is an input unit whose activation in_{ij} represents the probability that $word_i$ has part of speech tag pos_j. In this research the lexical part of speech probability are estimated by dividing the number of times a word occurs with a given tag by the overall number of times the word occurs. In this research also observe the lexicon of the words; during the lookup of a word in the lexicon of the Net-Tagger it has three parts called full form lexicon, a suffix lexicon and a default entry. The network is trained on a tagged corpus which is two million words part of the Penn-Treebank and it was tested on 100,000 words which was not part of the trained corpus. The performance 96.22%, 96.06% and 94.24% was achieved for Net-Tagger, trigram tagger and Hidden Markov model respectively. These experiments demonstrates that the performance of the Net-tagger is comparable to that of trigram tagger and better than that of the HMM tagger. They further show that the performance of the Net-tagger is less affected by a small amount of training data than that of the trigram tagger.

Another earlier part of speech tagging using neural network was done for Hindi language and compares them with two other machine learning approach, HMM and CRF [12]. In this research two network taggers are presented, the first one is single neural tagger is a neural network based POS tagger with fixed length of context chosen empirically is presented first. Then the second type of tagger is multi-neural tagger which consists of multiple single-neural taggers with fixed but different lengths of contexts are presented. Multi neural tagger performs tagging by voting on the output of all single neural tagger. A multi-layer perceptron network with three layers is used as a single neural tagger and is trained in supervised manner with well-known error back propagation learning algorithm. Java programs have been implemented in order to prepare lexicon and implement MLP network and error back propagation learning algorithm and to achieve the results shown in this paper. In this research the corpus is divided in to three corpuses which are called development, training and testing corpus. The size of the corpus was divided into training; development and testing corpus were 187,095, 23,565 and 23,281 words respectively. Percentage of unknown words in the development and testing corpus were 5.33%

and 8.15% respectively. Tools used for the experiment in this research Morfessor Categories-MAP, a tool which finds stem, suffix and prefix from un-annotated text was used to handle unseen words. Brant's TnT HMM based tagger and CRF++, a CRF based tagger were used to compare performance of the presented taggers. The performance evaluations was done on both the development corpus and testing corpus for the three taggers called multi-layer tagger, HMM, CRF and have for HMM tagger the performance was 95.18%, 91.58% for the development and test corpus respectively. The Multi-neural tagger also has 95.78%, and 92.19% respectively. The CRF performance was resulted with 96.05%, 92.92% for the development and test corpus respectively.

Another work [17] which has used Levenberg-Marqurdt learning neural network for POS tagging of Arabic sentences was done by Hassan Muaidi. In this research Levenberg-Marqurdt algorithm was used to train the ANN. Levenberg-Marquardt algorithm is an approximation to the Newton method used for training the neural network. In order to let understand the neural network and use the tokens, all letters should be converted or transformed to numeric value. In this research the binary coded method is used for the coding of the tag-sets. Input vector $\mathcal{X} = \langle X_1, X_2 \dots X_{12} \rangle$ with its corresponding output $\mathcal{Y} = \langle Y_1, Y_2 \dots \rangle$ are presented to the ANN which is considered as experimental data. Using the input vector $\mathcal{X}$, the output $\mathcal{O}$ is calculated this value is differs than the desired output $\mathcal{Y}$ and it is called the actual output or the net output. The difference between the desired and the actual output is computed and is called the error. After the error is calculated using the mean squared error (MSE) the error is propagated backward to change the weights in order to reduce or minimize the error rate. This process is repeated for a series of experiments until the error rates is acceptable. For this research a corpora of 24,810 words are collected and manually tagged to train the neural network and to test the performance of the developed POS-tagger. The tag-set used in this research was adopted the tag-sets done by other researcher that contains a total 161 detailed tags and 28 general tags covering Arabic main POS classes and sub-classes and the Arabic letters are coded to a numeric value. The developed tagger achieved an accuracy of 98.83% when evaluated on the train set and 90.21% on the test set. And also the algorithm also compared with the existing Arabic algorithm called Back-propagation algorithm so it has better performance and efficient than the existing algorithm called Back-propagation algorithm.

Another work of neural network based part of speech tagging for Hindi was done [30]. In this research two steps are done to complete the part of speech tagging; the first step is that the Hindi text are tagged using the rule based approach. Afterwards the second step, the tagged texts are checked for detection and correction of any anomaly using the neural network approach. The proposed tagger is work first raw sentences are passes through the tokenize system. In this step it splits the sentences into words and indexed it as token. The resulting words with tokens pass through rule based POS, simply using the lexicon. For correction and accuracy it finally passes through the ANN based POS tagger using the pattern recognition of corpus. To analyze the effectiveness of the proposed approach, 2600 sentences of news items having 11500 words from various newspapers have been evaluated. The POS tags in the sentences are represented using a numeric value. In this research uses binary representation for the POS tag. During simulations and evaluation, the accuracy up to 91.30% is achieved, which is significantly better in comparison to other existing approaches for Hindi parts of speech tagging.

2.5. Summary

Natural language processing is concerned with the processing and understanding of natural language to understand by a computer. NLP is a general term which has different applications that can be process by a computer such as information retrieval, information extraction, parsing, text to speech synthesis, part of speech tagging etc…are examples of NLP applications. Part of speech tagging is assigning a part of speech tag or a word class like noun, verb, adjective etc… to a word found in a corpus. Several POS taggers are done by using different approaches like rule based, stochastic, artificial neural network and hybrid. These approaches used for part of speech tagging have their own advantage and disadvantage so that why different researchers are doing on part of speech tagging using different approaches.

The statistical approach is requires considerable of training samples to estimate the probability of a word sequences. Rule based approach is another alternative to assign tags to a word using linguistic experts or automatic generation of rules through machine learning process. Neural networks are also interesting mechanisms which can learn general characteristics or patterns from limited sample data.

In this chapter we have tried to elaborate how the ANN is working and describing some of the types of neural network. So that to be applied to part of speech tagging we must know how the ANN is working and the steps to implement using neural network algorithm. Different researchers have different assumptions on the number of datasets used for neural network. In the next chapter we will discuss about the word class of the Tigrigna language because it is a crucial to implement the part of speech tagging for Tigrigna language.

Chapter 3

3. Tigrigna POS and Tag sets

3.1. Introduction

Ethiopia is a multi-lingual country which includes more than 85 nations and nationalities with speaking different languages which have different in morphological, semantically, syntactically and lexically. Tigrigna is one of the Ethiopian and Eritrean languages, which is grouped under Semitic language family. Tigrigna is an official language of Eritrean people and Tigray region's people. All subjects that are taught in Tigray region and speakers of Tigrigna language are given in Tigrigna in the elementary schools and it is given as a degree program in Mekelle University. It uses a special character representation called Ge'ez. The Tigrigna language has its own distinct way of grammar construction, character representation called **ፊደል** (Fidel) and sentence formation as revealed by [1] [9] [18].

Words are traditionally grouped into equivalence classes called parts of speech (word classes, morphological classes, lexical tags. In traditional grammars there were generally only a few parts of speech (noun, verb, adjective, preposition, adverb, conjunction, etc.) [15].

3.2. Tigrigna Sentences Structure

Tigrigna sentences, as any other languages can be divided into two largest immediate constituents called subject and predicate [40]. Understanding subject and predicate is the key to good sentences writing. The following examples are taken from [1], each sentences have the subject and predicate structure.

A. **ሓውካ መጺኡ** ፡፡ hawka metsiu / your brother came (has come).

B. **ሓውካ ትማሊ ካብ ሽረ መጺኡ** ። hawka tmali kab shire metsiu / your brother came (has come) yesterday from shire.

C. **እቲ መጽሓፍ ኣብ ልዕሊ ዓራት ግበሮ** ፡፡ eti mtshaf ab leli arat gbero / put the book on the bed

In the first two sentences the word **ሓውካ/** hawka indicates the subject of the sentences while the rest **መጺኡ**/ metsiu and **ትማሊ ካብ ሽረ መጺኡ/** tmali kab shire metsiu indicates the predicate sentence. The subject of a complete sentence is who or what the sentence is about and the predicate tells about that subject. These two sentences elements contains the noun phrase that made up of a noun and all its modifiers that functions in a sentences as a subject, object and complement and verb phrase that consists of a main verb plus one or more helping verbs, its complements, objects, or other modifiers and functions as a verb which are headed under the noun and verb respectively. In the third sentence the subject is ended with the word **ግበሮ**/ gbero which indicates you 3rd person singular person that acts as a noun phrase and **እቲ መጽሓፍ ኣብ ልዕሊ ዓራት ግበሮ**/ eti mtshaf ab leli arat gbero acts as a verb phrase.

Sentences in Tigrigna can be classified as simple and compound sentences [1, 40]. Simple sentences is a sentences that had one subject part and one predicate part where as a compound sentences is a sentences that contains two or more simple sentences joined by conjunctions or that contains more than one verb. The examples that have mentioned in the above are simple sentences. The following sentences are an example of compound sentences in Tigrigna that are taken from [1].

እቲ ጤል ዝሰረቀ ሰብኣይ ተኣሲሩ ፡፡ /eti thel zsereqe sebeay teasiru/ (the man who stole a goat is jailed)

In this sentence there are two verbs **ዝሰረቀ/**zsereqe (stole) and **ተኣሲሩ**/teasiru (jailed). In other words, it contains a subordinate and main clause which have one verb each. A subordinate clause is a clause which is dependent to another independent clause called the main clause [1, 40]. In this case the main clause is **እቲ ሰብኣይ ተኣሲሩ**/ eti sebeay teasiru (the man is jailed) and the subordinate clause is **ጤል ዝሰረቀ/**thel zsereqe (who stole the goat).

3.3. Word Classification of Tigrigna

3.3.1. The Need for Categorizing a Word

Sentences is a grammatical unit of one or more words that express an independent statement, question, request, command, exclamation, etc and that typically has a subject and predicate to express a thought. So to articulate a meaningful thought, classification of words is playing a great role in formation of sentences. The same words can be used in different sentences and belongs to a different word class in each sentence.

Three basic criteria are considered in order to categorize words in a language. They are: the meaning of the word, the form or shape of the word, and the position or the environment of the word in a sentence. These can be taken as the main criteria to determine the categories of a given word [1, 38, 42].

3.3.2. Tigrigna Word Classes

Words of Tigrigna in general can be classified into two: open and closed. Open classes and closed classes are so called because new members are always added to the former and are unlimited in number; whereas, members of the closed classes are relatively fixed and few in number [1, 40].

An attempt of classification of Tigrigna words are done by [40]. He has classified Tigrigna words into two broad categories: open and closed in general and into eight classes in particular. The three classes, Noun, Verbs and Adjectives, are in the category of open classes and the rest five namely Pronouns, Determiners, Adverbs, Prepositions and Conjunctions, are in the category of closed classes. Interjections are words without syntactic functions. In this categorization, interjections are not considered as part-of-speech/word class.

Tigrigna Noun Class

A noun is a word that functions as the name of some specific thing or set of things such as living creatures, objects, places, actions, qualities, state of existence or idea. A noun is a word which is used to label things, such as a real thing (for example, cat), an imaginary thing (for example, ghost), an idea (for example, love), person name (for example, 'Abebe'). Tigrigna nouns, like English, are words used to name or identify a class of things, people, places or ideas.

Tigrigna Verb Class

The verbs are words or compound of words that expresses action, a state of being and/ or relationship between two things. Verbs are morphologically the most complex POS in Tigrigna, with many inflectional forms; numerous words with other POS are derived primarily from verbs. There are two major criteria to identify verbs from other word categories. These are: Syntactically and morphologically. In the former case, verbs function as predicates in a simple sentence and they are found at the end of a sentence. In the latter case, they reflect grammatical categories such as aspect, mood and agreement.

Tigrigna Adjective Class

Tigrigna adjectives may have separate masculine singular, feminine singular and plural forms, and adjectives usually agree in gender and number with the nouns they modify. The plural forms follow the same patterns as noun plurals; that is they may be formed by suffixes or internal changes or a combination of the two [41]. For example **ፀሊም** / tselim (black for boy), **ፀላም** / tselam (black for girl), **ፀለምቲ** / tselemti (black for plural) are indicating adjectives that represents for masculine, feminine and plural forms respectively.

An adjective is a describing word, the main syntactic role of which is to qualify a noun or noun phrase, giving more information about the object signified. Adjectives in Tigrigna usually precede the nouns that they modify or describe. For example ፀላም ጓል ፈትየ፡፡ / tselam gual fetye/ I love black girl. In this example the adjective ፀላም / tselam (black) preceds the ጓል / gual (girl) which it modifies. But this does not mean that a word is an adjective just because it precedes a noun. For instance, in the sentences **እቲ ወዲ ነዊሕ እዩ ፡፡** / eti wedi newih eyu (the boy is tall) the word **እቲ** / eti (the) precedes the noun **ወዲ** / wedi (boy), although the word **እቲ** / eti (the) functionally shares the feature of an adjective, modifier, it is a demonstrative pronoun.

Tigrigna Pronoun Class

In most languages, there are a small number of basic distinctions of person, number and often gender that play a role within the grammar of the languages. It can be taken as noun because it can function as a noun and can take the position of a noun. Examples of Tigrigna pronouns are **ኣነ** / ane (I), **ንሱ** / nsu (he), **ንሳ** / nsa (she), **ነስኺ** / nski (you female singular), **ንስኻ** / nska (you masculine singular) **ንስኻትኩም** / nskatkum (you masculine plural) etc…

Tigrigna Preposition Class

Prepositions are small words which will have meanings only when they are attached or used together with other words such as nouns, verbs, pronouns and adjectives. They can express relationship between person, thing, or event etc and another. Tigrigna prepositions have the following central property [1]:

- They take nouns or adjectives as their complements (A simple preposition prefixed or attached with other words (e.g. nouns and verbs): e.g. **ብ-ሃንደበት/b**-handebet (Suddenly)
- They can stand alone as a separate word: e.g. **ናብ- ቤት ትምህርቲ /nab** bet tmhrti (To School)
- As compound prepositions having two parts, prepositional prefixes and post positions. The postpositions can either be single preposition that stand by their own or a preposition not separated from a noun.

ምስ እግዚኣብሔር

Preposition **ምስ** / ms (with) noun **እግዚኣብሔር** / egziabher (GOD)

Tigrigna Adverb Class: An adverb is another word class that acts as a modifier. Adverbs function to add more information to verbs, other adverbs, adjectives, and even to whole sentence. In many languages adverbs are classified as open classes; however, Tigrigna's adverbs are classified as closed classes [1] [38] [40]. Modifiers of verbs or verb phrases usually express time, place, manner etc. Modifiers of adjectives and adverbs commonly express degree while adverbs functioning as sentence modifiers usually express the speakers‟ attitude regarding the event spoken. Example:

I. **ሎሚ ቀልጢፉ መጺኡ ፡፡** / lomi qeltifu metsiu/ today he came quickly From the above sentence the word **ሎሚ** /lomi (today) represents adverb of time, **ቀልጢፉ** / qeltifu (quickly) indicates adverb of manner and **መጺኡ** / metsiu (came) is verb of the sentences. These adverbs are articulates the time and manner of the words.

II. **ንፀጋም ገፅ ተመለስ ፡፡** / ntsgam gets temeles (turn to the left side)

The word **ንፀጋም** / ntsgam (to the left) point to the adverb that shows place of adverb, **ገፅ** / gets (side) is noun and **ተመለስ**/temeles (turn) is the verb.

III. **ትማሊ ኣዝዩ ቀልጢፉ መፂኡ ፡፡** / tmali azyu qltifu metsiu (yesterday he came very quickly) Here the word **ትማሊ**/ tmali (yesterday) is adverb of time, **ቀልጢፉ** /qeltifu (quickly) is adverb of manner and the word **መጺኡ** / metsiu (came) shows the verb of the sentence. But there is one additional word called **ኣዝዩ** /azyu (very) is added before the adverb of manner **ቀልጢፉ** /qeltifu (quickly) to modify the adverb of manner [1, 40].

Tigrigna Conjunction Class: Conjunctions are words that join words, phrases or sentences and used to create larger grammatical units. Conjunctions can be distinguished as two categories: coordinating or subordinating. They coordinate words, phrases, clauses and sentences.

E.g. **ሰብኣይ-ን-ሰበይት-ን** / sebay-n-sebeyt-n (husband and wife)

Most of the items that function as prepositions in Tigrigna can also function as conjunctions.

Determiner in Tigrigna: Tigrigna demonstrative adjectives divided into expression for near ('this, these') and far ('that', 'those') referents, with separate forms for the four combination of singular and plural number and masculine and feminine gender [41].

Tigrigna Demonstrative Adjective			
Number	Gender	Near	Far
Singular	Masculine	እዚ/ezi	እቲ/eti
	Feminine	እዚኣ/ezia	እቲኣ/etia
Plural	Masculine	እዚኦም/eziom	እቲኦም/etiom
	Feminine	እዚአን/ezien	እቲኤን/etien

Table 3.1 Tigrigna Demonstrative Adjective

Interjection in Tigrigna

Introjections in the language are words that have unique functions to express emotion, sudden surprise, pleasure, sadness, and so on. Tigrigna has many words to articulate emotions, sudden, surprise pleasure and unexpected happens. For example the word **'ዋ!'** is used to give warning in Tigrigna when somebody is at fault. These Tigrigna interjections can stand-alone by themselves outside a sentence or can appear anywhere in a sentence.

e.g. **ዓሸ**! / ash! (WOW!) **ዓሸ**! **ንፍዕቲ ዛጓለይ** / ash! nfeti zagualey (WOW! You (my daughter) excellent)

Numerals in Tigrigna

Tigrigna has singular and plural number, but nouns that refer to multiple entities are not obligatorily plural. That is, if the context is clear, a formally singular noun may, refer to multiple entities: ሓሙሽተ/ hamushte (five), ሰብኣይ / sebeay (man), ሓሙሽተ ሰብኣይ / hamushte sebeay (five men). It is possible for a formally singular noun to appear together with plural agreement on adjectives or verbs: ብዙሓት / bzuhat (many plural), ዓዲ / adi (village), ብዙሓት ዓዲ/ bzhat adi (many villages) [41].

There are also special numerals in Tigrigna that correspond to the English half, quarter etc. Examples of these include ፍርቂ/frqi (half), ርብዒ/rbei (quarter) and ሲሶ /siso (one third).

As in Arabic, Tigre, and Ge'ez, noun plurals are formed both through the addition of suffixes to the singular form and through the modification of the pattern of vowels within (sometimes outside) the consonants that make up the noun root. In some cases suffixes may also be added to an internal plural. For example the word ዓራት / arat (bed) the plural form is ዓራታት፣ዓራውቲ / aratat or arawti (beds), ሓረስታይ / harestay (farmer), ሓረስቶት /harestot (farmers) [41].

In Tigrigna there are also that are completely irregular plurals.

e.g. ሰበይቲ / sebeyti (woman) is singular form, ኣንስቲ / anesti (women) is plural form.

ጓል / gual (girl, daughter) is singular form when we change to plural form it looks like this, ኣዋልድ / awald (girls), daughters (alongside ኣጓላት / agualat).

3.4. Tigrigna Tags and Tag-sets

The broad Tigrigna categories are explained from different works for different languages because most of the word classes used for these languages is used for Tigrigna language also [1] [20] [38]. In this section we can try to discuss the actual tags used in this thesis. A tag is a word or phrase that describes the information of lexical category of a word that are found in a sentence and tag-sets are the collection of tags that are used to label the words using the Tigrigna part of speech tagger.

According to [1] there are no any readymade tag-sets for Tigrigna unlike that of Amharic language that can be used for language processing applications. So Teklay Gebregziabher tries to prepare by asking different experts of Tigrigna language and by customizing the tag-sets prepared by Amharic language, since both languages are almost they have the same property. In addition the researcher of [1, 38] are used the tag-sets prepared by [43] due to the similarities of the languages because both are Semitic derived from the same ancestor (Geez). This work [43] was done for the annotation of Amharic text news collected form Walta Information Center. The researchers [43] have identified 30 Tag-sets for the annotation of the text news that Solomon Asres has used for his thesis work [38]. So for this thesis we used the tag-sets that are compiled by Teklay Gebregziabher that are 36 tag-sets was done to annotation of Tigrigna texts collected from different sources such as websites of Dimtsi Woyane Tigray, Mekalh Tigray, Woyen Newspaper, FM Mekelle, Tigray Development Association and Ethiopian Radio and Television Agency Tigrigna Department as the news collected from these sources can be representative of the language from the perspective of the news domain. According [1] there are around one thousand sentences are collected with 26,000 of words containing around 8000 distinct words were collected. The tag-sets that are discussed below are classified as a basic class and subclasses of the basic class where noun, pronoun, verb, adjective, preposition, conjunction, adverb, interjection are considered to be the basic classes. In addition, numeral and punctuation are also included as basic classes in the process of identifying the tag-sets. The concept hierarchy of the total tag-set that are identified is represented in the following figure in terms of the basic classes and their respective subclasses. The figure is taken from the work of [1]

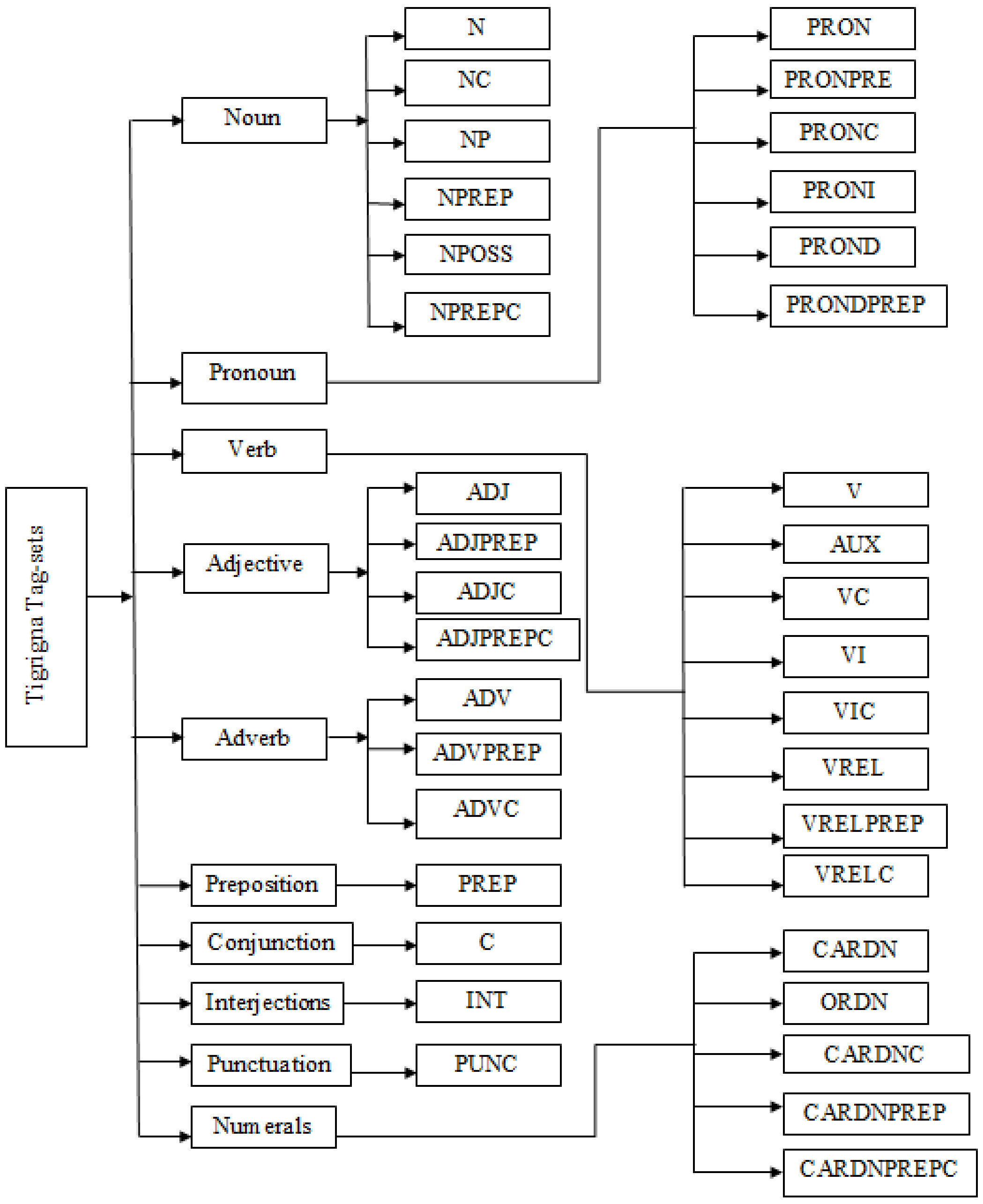

Figure 3.1 Tag-set concept hierarchies [1]

Noun

A noun is a word that functions as the name of some specific thing or set of things, such as living creatures, objects, places, actions, qualities, states of existence, or ideas. They have attributes such as gender, case, number and definiteness. But for this research we are considering the only tag-sets done by [1] because applying all the attributes of the noun makes the tagsets detail and complex. According to the above figure the noun is classified into verbal noun, verbal noun with conjunction, verbal noun with preposition, proper noun, verbal noun with possession and verbal noun with both preposition and conjunction.

- Nouns in Tigrigna that describes some specific name of person, place, thing, institutions, organizations, religions etc are represented as a subclass of noun and are represented or tagged as NP. Example **ትግራይ** / Tigray, **ኣልማዝ** / Almaz, **ኣብያታ** / Abayata, etc
- Noun prefixed with preposition and when the preposition cannot be separated from the noun is considered to be the noun preposition subclass and tagged as NPREP. Example **ብ-ባንኪ** / b-banki (by bank), **ብ-ትካል** / b-tkal (by organization)
- Nouns affixed with conjunction and when the conjunction cannot be separated from the noun is considered to be the noun conjunction subclass and tagged as NC. Example **ፓርትታት-ን ውልቀ-ን** / parttat-n wlqe-n (parties and individuals)
- Nouns that are attached to possession and cannot be separated from the noun and are classified as noun possession subclass and tagged as NPOSS. Example **ሃገር-ና** / hager-na (our country)
- Sometimes nouns can be attached with preposition and conjunction and cannot be separated from the noun, and classified as noun with preposition and conjunction and tagged as NPREPC. Example **ብ-ምውላዕ-ን** / b-mwla-n (by lighting and)
- Other forms of nouns such as common nouns, concrete nouns, abstract nouns etc that can be classified under the above subclasses are tagged as N. examples are: **ምግቢ** / mgbi (food common noun), **ኣራዊት** / arawit (undomesticated animal concrete noun), **ቁልዕነት** (childhood Abstract noun)

Pronoun: Pronouns are used instead of noun or noun phrases or they are special types of nouns. But they are treated as independently in this thesis work than taken as a subclass of noun. As stated in [43], pronouns are taken as independent of noun in Amharic work so concerning this we take the pronoun for Tigrigna as independent also. There are five sub categories and one general tags.

- Pronouns which come with preposition and cannot separate are called pronoun with preposition and are tagged as PRONPRE. Example **ን_ባዕልና** / n-baelna (for us)

- Pronouns which come with conjunction and cannot separate are called pronoun with conjunction and are tagged as PRONC. Example **ኣነ_ን_ንስኻ_ን** / ane-n-nska-n (me and you)
- Pronouns that can be used to ask questions are classified as interrogative Pronouns and are tagged with PRONI. Examples: **እንታይ** / entay (what), **ንምንታይ** / nmntay (why)
- Pronouns that can identify a noun or pronoun are called demonstrative pronoun and are tagged as PROND. Example: **እቲ** / eti (the)
- The demonstrative pronoun can be attached with prepositions and classified as demonstrative pronoun with prepositions and it is tagged as PRONDPREP. Example **ኣብቲ** / abti (on the)
- Pronouns that cannot classified as the one of the above classification are tagged as PRON. Example: **ሕድሕደ** / hdhd (for each)

Verbs

The verbs are words or compound of words that expresses action, a state of being and/ or relationship between two things [15]. A sentence without a verb cannot give a complete meaning. According [1] verbs are classified into seven subclasses and one general verb categories, so that we are using these tags to this research.

- Auxiliary verbs are one sub categories which are tagged by **AUX**. AUX is used to show such verbs in all forms without any distinction like number, gender, tense etc. example: **እዮም** / eyom (masculine and feminine plural), **እዩ** / eyu
- Verbs which are attached with conjunction are tagged as **VC** for any type of verb, auxiliary or relative. Example **በሊዑ_ን_ሰቲዩ_ን** / beliu-n-setiu-n (ate and drink)
- Verbs that show an infinitive are tagged by VI. Example: **ንምሻጥ** / nmshat (to sold), **ንምርግጋፅ** / nmrggats (to assure)
- An infinitive verb attached with conjunction is tagged as VIC. Example: **ንምውሳኽ_ን** / nmwsakn (to add and)
- Relative verbs of Tigrigna are verbs that are prefixed most of the time with ዘ or ዝ which indicate the subject of the sentence and are tagged with VREL. Example: **ዝጠመተ** / zTemete (focused)
- Relative verbs which are attached with prefix are tagged with VRELPREP and are called relative verbs with preposition. Example **ንዘይኣምኑ** / nzeyamnu (who didn't agree upon)

- Relative verbs which are attached with conjunctions are tagged with VRELC. Example: **ዝረኸቡ_ን** / zrekebu-n (who gained something)
- All other verbs which cannot be classified in the above subclasses are tagged by the general tag for verbs V. Example: **በቂዑ** / bekiu (enough)

Adjective

Adjectives in a sentence modify nouns to denote quality of a thing; that is, it specifies to what extent a thing is as distinct from something else. In the identification of tagsets for Tigrigna part of speech tagger, one general tag and the following three sub categories are stipulated out.

- Adjectives attached with preposition are called adjective with preposition and are tagged with ADJPREP. Example: **ን_ቀፃሊ** / nqtsali (for future)
- Adjectives followed by conjunctions are referred to the adjective with conjunction and tagged with ADJC. Example: **ኢኮኖሚያው_ን** / ekonimiyawi-n (economically)
- Sometimes adjectives are attached with preposition affixes and conjunction suffixes that cannot separate from the adjective and tagged by ADJPREPC. Example: **ን_ማሕበራው_ን** / n-mahbraw-n (for social and)
- Any other adjective which does not belong to these subcategories is tagged as a general tag ADJ. Example: **ኣይተ** / ayte (Mr.)

Adverb: Tigrigna adverbs are words that modify a verb, adjective, clause or other adverbs. Though adverbs can be classified as adverbs of time, manner, place etc, these all are considered as a general adverb that can be tagged as the general adverb tag (ADV) and the following two categories.

- Adverbs attached with preposition that cannot separate from the adverb are called adverb with preposition and tagged with ADVPREP. Example: **ብ-እዋኑ** / b-ewanu (by that time)
- Adverbs attached with conjunction that cannot detach from the adverb are called adverb with conjunction and tagged with ADVC. Example: **እዋናዊ_ን** / ewanawi-n (situational)
- Adverbs that are other than these two adverbs are tagged by ADV which is a general tag. Example: **ብሓፈሻ** / bhafesha (in general)

Preposition

Prepositions alone will not convey any meaning unless they are attached or used with nouns and other basic classes. Some of the prepositions are attached to the word, may be noun or something else, in such a way that they cannot be separated as this work focuses on the surface form of the words, and some of them can be found alone used with other words separately. If the preposition is separated from the word being used it is tagged as PREP. Example: **ኣብ** / eab (on), **ልዕሊ** / leili (over).

Conjunction

Conjunctions are words that join words, phrases or sentences. If the conjunctions are used with words such as nouns, adjectives as a separate word they are tagged as C. Example: **ብሓፈሻ** / bhafesha (in general).

Numerals

Numerals include all elements which refer to quantity or amount. In Tigrigna like the other languages numerals can be divided into cardinal and ordinal numbers which are tagged as CARDN and ORDN respectively. But in this research there are also other representations of tag according to [1]:

- If the cardinal number is attached with a conjunction and is tagged with CARDNC and is called cardinal number with conjunction. Example: **ክልቲአን** / kltian (two of them), **ን** 7 (to 7)
- if the cardinal number is attached with a preposition is called cardinal with preposition and is tagged by CARDNPREP
- sometimes cardinal numbers come with affixes called preposition and with suffixes called conjunction and is called cardinal with conjunction and preposition and is tagged CARDNPREPC

Interjection

An interjection or exclamation is a word used to express a particular emotion or sentiment on the part of the speaker. An interjection sometimes expressed as a single word or non-sentence phrase, followed by a punctuation mark. Interjections, words or phrases used to express emotions, can stand by themselves or can appear anywhere in a sentence. And is tagged by INT. Example: **ዋ! / Wa!, ዓሽ!! / ash!!**

Punctuations

In Tigrigna there are different punctuations like ፡, ፡ ፡, ፤, ፡-?! , tagged by PUNC.

Unknown word (Unclassified word)

For words not found in the lexicon of the tagger, **UNC** is assigned because there might be unrecognized words.

3.5. Summary

Tigrigna parts of speech are classified in different categories according different experts for different usage. According to Teklay [1], he tries to classify the idea of different professionals of used for Tigrigna parts of speech and he merged into ten major classes of the parts of speech with their different subclasses. The classification type can be varying according to the objective set by the researchers. For this thesis work, all essential word categories are tried to consider for the tagging process. These are very important to set the tags for Tigrigna language.

NO	Basic category or tag	Derived category or tag	Description	Example
1	Noun	N	Noun	ቤትፅሕፈት/biettshfet (office)
2		NP	Proper noun	ትግራይ/tgray, ዳዊት/ Dawit
3		NC	Noun + conjunction	ከተማታትን/ktematatn (towns’s)
4		NPREP	Noun + preposition	ንዕውትነት/newtnet (for success)
5		NPREPC	Noun + preposition + conjunction	ብምሕጋዝን/bmhgazn (by supporting)
6		NPOSS	Noun + possession	ሃገሮም/hagerom (their country)
7	Pronoun	PRON	Pronoun	ኣነ/ane, ንስኻ/nska
8		PRONPREP	Pronoun + preposition	ንባዕልና/nbaelna
9		PRONC	Pronoun + conjunction	ኣነንንስኻን/anennskan
10		PRONI	Infinitive pronoun	እንታይ/enaty
11		PROND	Demonstrative pronoun	እቲ/eti
12		PRONDPREP	Pronoun + demonstrative + preposition	ኣብቲ/abti
13	verb	V	Verb	ተገሊፁ/tegelitsu
14		AUX	Auxiliary	እዩ/eyu
15		VC	Verb + conjunction	በሊዑንሰትዩን/beliunsetiyun
16		VI	Infinitive verb	ንምርግጋፅ/nmrggats
17		VIC	Infinitive verb + conjunction	ንምቅላልን/nmqlaln
18		VREL	Relative verb	ዝጠመተ/ztxemete
19		VRELPREP	Preposition + relative verb	ንዝተሓቱ/nztehatu
20		VRELC	Conjunction + relative verb	ዝተሳተፍሉን/ztesatefulun
21	Adjective	ADJ	Adjective	ዶክተር/dokter
22		ADJC	Adjective + conjunction	ሰላማውን/selamawn
23		ADJPREP	Adjective + preposition	ንመሰረታዊ/nmesertawi

24		ADJPREPC	Adjective + preposition + conjunction	ንማሕበራውን/nmahberawn
25	adverb	ADV	Adverb	ብልቃሕ/blqah
26		ADVPREP	Adverb + preposition	ብኣማኢት/bamaeit
27		ADVC	Adverb + conjunction	እዋናዊን/ewanawi
28	Preposition	PREP	Preposition	ኣብ/ab
29	Conjunction	C	Conjunction	ድሕሪ/dhri
30	Numerals	CARDN	Cardinal number	100
31		ORDN	Ordinal number	ቀዳማይ/qedamay
32		CARDNC	CARDN + conjunction	ሓደንክልተን/hadenklten
33		CARDNPREP	CARDN + preposition	ብ100, ንክልተ/nklte
34		CARDNPREPC	CARDN+ preposition + conjunction	ንሰለስተን/nselesten
35	Punctuation	PUNC	Punctuation	፡, ።, ፤, ፣, ፡-
36	Interjection	INT	Interjection	ዓሸ!/ash!
37	Unknown word	UNC	Unclassified word	

Table 3.2 Tigrigna Tag-sets

Chapter 4

4. Design of Tigrigna POS Tagger

Assigning grammatical categories to words in a text is an important component of a natural language processing (NLP) system. Text collection tagged with Part of speech (POS) information are often used as a prerequisite for more complex NLP applications such as information extraction, syntactic parsing, machine translation or semantic field annotation etc. A POS tagger is a program that assigns part of speech to Tigrigna words based on the context in which they appear in the sentences. In this chapter, we describe about the design of the artificial neural network tagger as well as approaches and design goals of Tigrigna POS tagger.

4.1. Design Approaches and Techniques

Part of speech tagging is the process of marking up the words in a text as corresponding to a particular part of speech, based on its definition as well as its context i.e. relationship with adjacent and related words in a phrase, sentence, or paragraph. Part-of-speech tagging is not a mere list of words and their parts of speech, since some words can represent more than one part of speech at different times. As we described in chapter 2 there are many approaches and techniques to tackle this problem such as rule-based, statistical, and neural networks. These approaches help us to device a POS tagger for a specific language like Tigrigna. Our research concerns only artificial neural networks.

In the field of computational linguistics, the problem of labeling words in a given sentence with the correct parts of speech can be tackled using different techniques. Some of the techniques are artificial neural network is one of the most currently popular approach.

Artificial Neural Networks, use learning algorithm that acquires a language model from training corpus. It is the recent approach that attracts attention in solving many computational linguistics problems [44]. As we discussed in chapter 2 it has been applied successfully for the assignment of POS tags due to its advantages over others.

Part-of-speech (POS-tagging) is considered as a process for automatically assigning the proper grammatical tag to each word of a written text according to its appearance on the text. Thus, the task of POS-tagging is attaching appropriate grammatical or morpho-syntactical category labels to every token, and even to punctuation marks, symbols, abbreviations,. . . etc. in a corpus [17].

It is often assumed that for each input lexicon there is a set of priori possible POS tag categories, or a probability function over them, and the tagger has to choose from this limited set of candidate categories.

According to [45, 46] POS tagging techniques are classified into supervised and unsupervised approaches. Supervised technique usually relies on pre-tagged corpora to automate the tagging process. From these pre-tagged corpora, it requires linguistic information in the form of probabilities so as to perform the training. Such type of tagging technique requires probabilities acquired in the training phase to assign tags to the words. Training and tests are performed based on tagged corpus.

Unsupervised techniques do not require pre-tagged corpus but use sophisticated computational techniques. Induction of dictionary and tag-set are possible in this technique without having tagged training data. As a result, tagging of test data can be performed using induced dictionaries. It is characterized by randomly generating tags and usually has low accuracy [47].

These approaches and techniques have direct implication and reflection on the design process in particular and the development of the tagger system in general [48]. The design of this thesis is dependent on a single approach which is call artificial neural network which uses multi-layer perceptron with back propagation techniques. Artificial Neural Network approach enables the system to learn the pattern through the training period. Based on pattern information, the system can classify words to their appropriate category unlike the statistical approach. However using this technique is not 100% efficient to classify word to their appropriate POS tag accurately but it has higher accuracy when they compare to other POS techniques by different experts. So why we choose for this thesis artificial neural network is it has more advantageous than the previous technique which is done for Tigrigna language that is HMM and rule based separately.

4.2.Design Goals

The main goal of designing artificial neural network POS tagger is to achieve better performance in tagging Tigrigna texts. In addition to this, the tagger is expected to be easy to implement and increase speed in responding a tagged text, and easy in obtaining the required knowledge that can be estimated using large tagged corpora when we compare to the other approaches of POS taggers like HMM and rule based basically done for Tigrigna text on previous [50, 51].

4.3.Designing Neural Network Model

As we explained in the above part of speech tagging for Tigrigna is done using neural network model. The process of designing Neural Network model is an iterative course of action that achieves an optimal result by adjusting the parameters of the network. In general, designing an ANN model consists of the following decisions and activities:

- Deciding network topology, which involves deciding the number of layers in the network, the number of neurons in each layer, and the type of connection and communication among neurons for different layers, as well as among the neurons within the same layer.
- Deciding the learning algorithm, this is a prescribed set of well-defined rules for the solution of learning problem
- Encoding schema, which refers to deciding the way a neuron receives input and produces output: how the input and output data are represented or encoded, is a major component to successfully instructing a network.

4.3.1.Feature Extraction

Feature extraction is the process of mapping the original features into fewer features that include the main information of a given word. With the advent of neural networks, more and more problems are solved by simply feeding large amounts of 'raw data' to a neural network [38].

During training the network learns what value to place on what feature. The features are extracted from the tagged corpus and pre-defined tagsets. Mainly, the features are extracted from the following information.

- **POS information**: the candidate POS tags to the current word occurrence in the corpus this helps us to determine what word is mostly occurred and then to determine the target output.
- **POS and word information**: The occurrence order indicates the frequency order of the POS in the training data when it is used for the current word.

Words, tags, and associated tags are very essential information in POS tagging process. As it is mentioned in chapter three, 36 tags are identified for this thesis work. So, for each word, its occurrence is calculated to be one specific tag. This is calculated using lexical probability concept, i.e., word frequency approach. It is based on the probability that a word occurs with a particular tag. Frequency measurement, i.e. (p (word/tag)), is the probabilities of a word given a category. The lexical generation probability is the probability that a given category is realized by a specific word and is estimated simply by counting the number of occurrence of each word by a category: $$P(W_i/T_i) = \frac{\text{Count of } (W_i, T_i)}{\text{Count of } (T_i)}$$

For each word there are 36 values, as it is shown in the table below, indicates the presence of 36 input vectors in each word and with their target outputs. The target output is taken if a word has higher frequency of a word tag occurrence.

Word/tag	T_1	T_2	…	T_{36}	Target output
W_1					T_{28}
W_2					T_1
W_3					T_1
…					…
W_n					T_m

Table 4.1 Probability of words given a tag

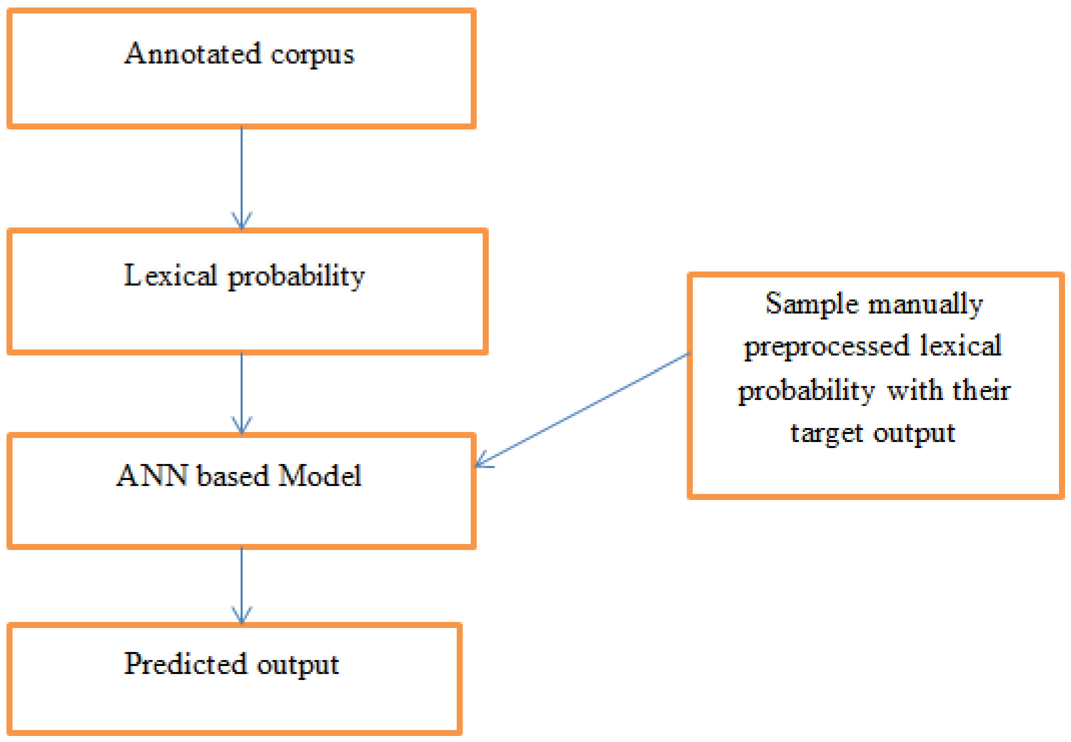
Annotated corpus
Lexical probability
Sample manually preprocessed lexical probability with their target output
ANN based Model
Predicted output

Figure 4.1 Architecture of ANN based Parts of Speech Tagger

4.3.2.Representation of Input for MLP _tagger

The tagging of POS can be tackled in two levels: at sentence level and at word level. In the case of word level tagging, the problem can be posed as a classification problem. Whereas, in the case of sentence level tagging, a series of tags corresponding to the sequence of words in the sentence need to be found. In this thesis work, the MLP-Tagger with back propagation algorithm addresses the classification problem posed by the word level POS tagging. The input to this network is the set of words that fall into a window of pre-specified size centered on the target word to be tagged. The output of the network is the corresponding tag for the target word.

As input to the Neural network takes numerical values encoding of input word into a suitable form, which the network can identify and use is essential. Each word, W, from the corpus is encoded as n-element vector $W = (P_{1\,,}P_{2,}P_3 \ldots P_m \ldots P_n)$, where n corresponds to the total number of tags and P_m is the prior probabilities (lexical probabilities) that the word W corresponds to the tagT_m. P_m is estimated from $P(Word/tag)$. Each word in a text document is processed and represented in a form of vector with 36 elements, since 36 tag categories are considered in this study [12, 38]. In order to understand and process by the neural network all words are transformed in to numeric value by calculating the lexical probability (frequency occurrence) of each word with corresponding tag.

4.3.3.Representation of Output for MLP_Tagger

For the coding of the target vector, the binary coded method is used. The idea behind this method is for each class in the tag sets there is a vector consists of binary numbers. The expected output is from the defined 36 tag sets. These tags are given numeric code in decimal number. The decimal number is converted into binary number, which has 6 bits because the total tag set is 36. The output is expected by these 6-bits binary number. Each tag has its own unique 6 bit representation.

4.3.4.Back Propagation Algorithm

Multi-layer feed forward networks, also known as multi-layer perceptron (MLPs) consist of a certain number of neurons which are split into several disjoint sets, so called layers. Every MLP contains an input layer, to which the input vector is fed, and an output layer, which produces the vector of output values. In addition, the network may contain one or more hidden layers, located between the input and the output layer [23].

Forward propagation network is a supervised learning algorithm and describes the flow of information through a neural network from its input layer to its output layer. The algorithm of forward propagation network is described as following [24]:

1. Set all weights to random values ranging from -1.0 to +1.0
2. Set an input pattern (decimal number ranging from 0 to 1) to the neurons of the network's input layer
3. Activate each neuron of the following layer:
   - ✓ Multiply the weight values of the connections leading to this neuron with the output values of the preceding neurons
   - ✓ Add up these values
   - ✓ Pass the result to an activation function, which computes the output value of this neuron
4. Repeat this until the output layer is reached
5. Compare the calculated output pattern to the desired target pattern and compute an error value
6. Change all weights by adding the error value to the old weight values
7. Go to step 2
8. The algorithm ends, if all output patterns match their target patterns

By performing this procedure repeatedly, the error value gets smaller and smaller. The algorithm is successfully finished, if the net error is zero (perfect) or approximately zero.

Back-propagation is a supervised learning algorithm and is mainly used by Multi-Layer-Perceptron to change the weights connected to the network's hidden neuron layers. The back-propagation uses a computed output error to change the weight values in backward direction.

Back-propagation is the most commonly used method for training multilayer feed forward neural network. This training scheme is used for adjusting the connection weight of each unit in such a way that the error between the desired output and the actual output is reduced. More clearly, the neural networks adjust the connection weights to each unit, beginning with the connection between the last hidden layer and the output layer [26].

To get this network error, a forward propagation phase must have been done before. While propagating in forward direction, the neurons are being activated using the sigmoid activation function.

The formula of sigmoid activation function is

$$f(x) = \frac{1}{1 \;+\; e^{-input}}$$

In BP algorithm, the network is first initialized by setting up all its weights to be small numbers. Next, the input pattern is applied and the output calculated. The calculation gives an output which is completely different from what is expected, since all the weights are random.

The algorithm works as follows:

1. Perform the forward propagation phase for an input pattern and calculate the output error
2. Change all weight values of each weight matrix using the formula

$$Weight(old) + learning\ rate \times output\ error \times output(neurons\ i) \times output(neurons\ i+1) \times \big(1 - output(neurons\ i+1)\big)$$

3. Go to step 1
4. The algorithm ends, if all output patterns match their target patterns

By performing this procedure repeatedly, the error value gets smaller and smaller. The algorithm is successfully finished, if the net error is zero (perfect) or approximately zero.

If all values of an input pattern are zero, the weights in weight matrix would never be changed for this pattern and the network could not learn it. Due to the fact, a ‘pseudo input’ is created, called bias that has a constant output value of 1.

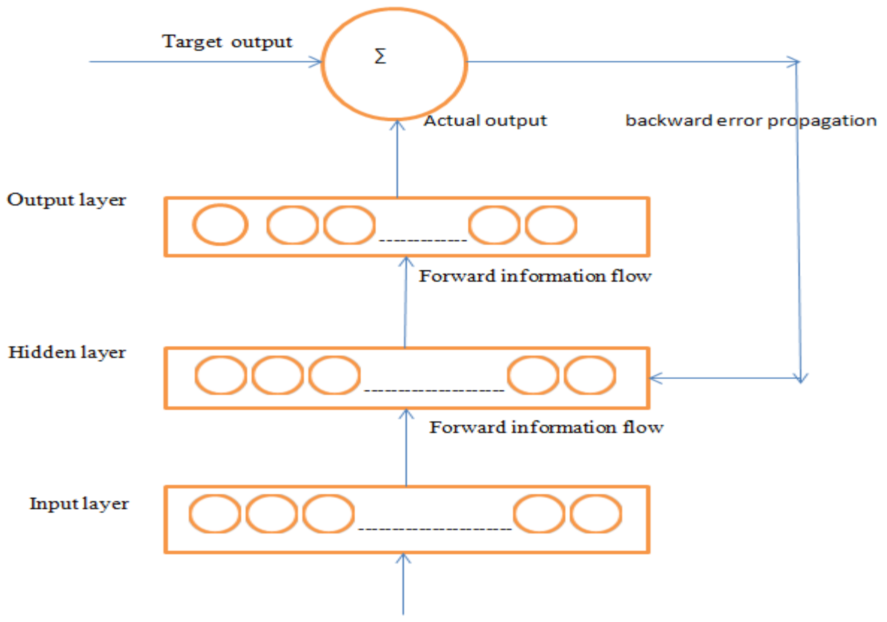

Figure 4.2 Back propagation network architecture

4.4.Summary

MLP-Network learns by adapting the weights of the connections between units, until the correct output is produced. One widely used method is the back propagation algorithm which performs a gradient descent search on the error surface. The BP algorithm uses the gradient descent to change link-weights in order to reduce the difference between the network output vectors and the desired output vectors. In general, MLP network with hidden layers are more powerful than networks without one. In this network types, the tagging of a single word is performed by copying the tag probabilities of the current word and its neighbors in to the input units, propagating the activations through the network to the output units and determining the output unit which has the highest activation value.

Chapter 5

5. Experimentation and Discussion

5.1. Data Preparation

Natural Language Processing is a data-intensive field. The success or failures of most NLP applications depend on the quality and availability of appropriate data. The data used in computational linguistic tasks generally takes the form of corpora. Corpora can be divided into two categories: annotated corpora and un-annotated corpora. Un-annotated corpora are simply large collection of raw text, whereas annotated corpora add additional information to the text, such as phonetic transcription, part-of-speech tags, parses trees, etc. Annotated corpora with appropriate part of speech are useful to train part of speech taggers.

5.1.1. Source of Sample Data

Tigrigna news texts were collected from different sources both in softcopy and in hardcopy. Such sources include websites of Dimtsi Woyane Tigray, Mekalh Tigray, Woyen Newspaper, FM Mekelle, Tigray Development Association and Ethiopian Radio and Television Agency Tigrigna Department as the news collected from these sources can be representative of the language from the perspective of the news domain [1].

Neural network requires large amount of data in order to do successful training process. Thus, it is important to have large size data which is annotated. But there is no well-prepared corpus for Tigrigna so it is difficult to prepare a large amount of data for this thesis work. But we take a sample of data around 16,000 words and it has around 290 sentences with above 2000 distinct words.

5.1.2. Lexicon Preparation

The lexical probabilities have been estimated by computing the relative frequencies of every word per category from the annotated corpus. All statistical information, that enables to develop probabilities, are derived automatically from a hand annotated corpus (the lexicon).

In the lexicon probabilities, each word W_i occurrences tagged with tag T_i is counted and divided by the counted number of occurrence of the tag (T_i Word with given lexical category).

$$P(W_i/T_i) = \frac{Count\ of\ (W_i, T_i)}{Count\ of\ (T_i)}$$

In the Table 5.1 below, for instance, the lexical probability of the word ኣብ tagged with PREP is calculated as:

Count of (ኣብ, PREP) = 277

Count of (PREP) = 508

So $P\left(\frac{ኣብ}{PREP}\right) = \frac{Count\ of\ (ኣብ, PREP)}{Count\ of\ (PREP)}$ $\quad where\ p\ is\ probability\ and$

$count\ of\ is\ number\ of\ occurences\ of\ words\ and\ tags$

$= \frac{277}{508}$ = 0.545276

Word with their appropriate category	Lexical probability
P (ኣብ/PREP)	0.545275590551
P(።/PUNC)	0.767624020888
P(ክምዝተፈጠሩን/VRELC)	0.0555555555556

Table 5 Sample Lexical probabilities

5.1.3. Representation of Input for MLP Tagger

As we have discussed in chapter four a neural network is working in numeric format. So each word W from the corpus is encoded as n-element vector: $W = (P_1, P_{2,} P_3 \dots P_k \dots P_n)$, where n corresponds to the total number of tags and P_k is the lexical probability that the word W corresponds to the tag T_k. Once we have calculated the lexical probability of each word we can get the target output by observing and comparing the occurrence of the word with a tag. For example table 5.2 shows a sample representation of the input dataset or corpus with their corresponding tags and lexical probability and by comparing the lexical probability we can decide the target output of each word.

Word/tag	T_1	T_2	…	T_{36}	Target output
መምህራን	0.00350877	…	…	0.0004205	T_{36}
ብመሰረት	0.0186335	0.0052634	…	…	T_1
ብሓፈሻ	0.006211	0.01578947	…	…	T_2
…	…	…	…	…	…
W_n	…	…	…	…	T_m

Table 5.1 Sample representation of the input dataset with their lexical probability and target output

The parameters for the network are taken by trial and error to get a better performance. To have a better performance we made n times to train to achieve a good result of performance. The parameters are the learning rate, number of folds, number of epochs and number of hidden neurons. The input could be a row from our training dataset, as in the case of the hidden layer. It may also be the outputs from each neuron in the hidden layer, in the case of the output layer.

5.1.4. Representation of Output for MLP Tagger

The output of each word is expected from the defined 36 tagsets. Each tagsets is represented by 6-digit binary number. The decimal and binary representation of each tag is presented in table 5.3. The output for MLP_Tagger can be represented as follows: OUT= (O1, O2, O3, O4, O5, O6). For example: F (target word) = N if OUT = (100101) =37.

NO.	Type of tag	Binary representation	Decimal representation
1	N	100101	37
2	NP	100100	36
3	NC	100011	35
4	NPREP	100010	34
5	NPREPC	100001	33
6	NPOSS	100000	32
7	PRON	11111	31
8	PRONPREP	11110	30
9	PRONC	11101	29
10	PRONI	11100	28
11	PROND	11011	27
12	PRONDPREP	11010	26
13	V	11001	25
14	AUX	11000	24
15	VC	10111	23
16	VI	10110	22
17	VIC	10101	21
18	VREL	10100	20
19	VRELPREP	10011	19
20	VRELC	10010	18
21	ADJ	10001	17
22	ADJC	10000	16
23	ADJPREP	1111	15
24	ADPREPC	1110	14
25	ADV	1101	13
26	ADVPREP	1100	12
27	ADVC	1011	11
28	PREP	1010	10
29	C	1001	9
30	CARDN	1000	8
31	ORDN	111	7
32	CARDNC	110	6
33	CARNPREP	101	5
34	CARDNPREPC	100	4
35	PUNC	11	3
36	INT	10	2
37	UNK		

Table 5.2 Binary and decimal number representation of tags

5.2. Implementation

Here we explain the detail implementation of the lexical probability and back propagation neural network. In this paper we have used python 2.7 and 3.5 software's are used to develop the implementation part on the Pycharm environment. First the lexical probability is implemented using python language on the python 2.7. And the output of the lexical probability is recorded to an excel sheet in the format of .CSV. In the row we put the tag sets and in the column we put the words with their target output on the last column of the excel sheet. Then we can implement the back propagation algorithm using the steps described in chapter 4. The back propagation neural network is implemented by using a series of steps which are first initialize the network, forward propagate, and back propagate error, train network and predicting using the given corpus.

Each neuron has a set of weights that need to be maintained. One weight for each input connection and an additional weight for the bias. We will need to store additional properties for a neuron during training, therefore we will use a dictionary to represent each neuron and store properties by names such as "**weights**" for the weights. It is good practice to initialize the network weights to small random numbers. In this case, we use random numbers in the range of 0 to 1.

We can calculate an output from a neural network by propagating an input signal through each layer until the output layer outputs its values this is called forward propagate. Forward propagate is divided in to neuron activation, neuron transfer and forward propagate. The input to the algorithm is the row from our datasets. Neuron activation is calculated as the weighted sum of the inputs.

$$activation = \sum (W_i \, In_i) + b$$

Where W_i $is\ the\ network\ weight\ at\ i\ and$

$In_i\ is\ an\ input\ at\ i\ and\ b\ is\ the\ bias$

Once the neuron is activated we need to transfer the activation to see the output. In this paper we used sigmoid activation function as a transfer function that can take any input value and produce a number between 0 and 1. The sigmoid activation function formula is:

$$output = \frac{1}{(1 + e^{(-activation)})}$$

Where e is base of the natural logarithms (Euler's number)

Error is calculated between the expected outputs and the outputs forward propagated from the network. These errors are then propagated backward through the network from the output layer to the hidden layer, assigning blame for the error and updating weights as they go. This back propagation method has two steps the transfer derivative which is the difference from the given output of neuron which can be calculated as follows [52]:

$$derivative = output \times (1 - output)$$

And the error back propagation the error for each output neuron; this give us our error signal (input) to propagate backwards through the network [52].

$$error = (expected - output) \times derivative(output)$$

Where **expected** is the expected output value for the neuron, **output** is the output value for the neuron and **derivative ()** calculates the slope of the neuron's output value.

The back-propagated error signal is accumulated and then used to determine the error for the neuron in the hidden layer, as follows [52]:

$$error = \left(W_k \times e_j\right) \times derivative(output)$$

Where

$W_k\ is\ the\ weight\ that\ connects\ to\ the\ k^{th}\ neuron\ to\ the\ current\ neuron\ and\ e_j\ is\ the\ error$

$signal\ from\ the\ j_{th}\ neuron\ in\ the\ output\ layer\ and\ output\ is\ the\ output\ to\ the\ current\ neuron$

The network is trained using stochastic gradient decent this part is divided into update weights and train network. Once errors are calculated for each neuron in the network via the back propagation method above, they can be used to update weights. Network weights are updated as follows [52]:

$$W = W + \delta \times e \times In$$

Where $\boldsymbol{W}$ is a given weight, $\boldsymbol{\delta}$ is a learning rate parameter that you must specify, $\boldsymbol{e}$ is the error calculated by the back propagation procedure for the neuron and $\boldsymbol{In}$ is the input value that caused the error.

5.3. Experimentation

In this study the problem of part of speech tagging is carried out through Artificial Neural Network using Back-propagation algorithm which is implemented by python programming language. We have used K-fold cross validation technique to make validation on the given datasets. This technique is works by splitting the data sets in to K-folds (K different subsets). We use K-1 subsets (folds) to train our data and leave the last subset (last fold) as test data. We then average the model against each of the folds and then finalize our model. The value of the K is it is advisable to take K=10 by most experts. We make experimentation by dividing the given dataset in to 75%, 50%, 25% and 100% to show the effect of K-fold validation technique. The experimentation is done by try and error method to get a better mean accuracy performance on the parameters of the neural network. The parameters are number of folds this can be used to divide the given datasets in to K-folds and then making training and testing based on the theory of K-fold cross validation. The second parameter is learning rate is a common parameter in many of the learning algorithms, and affects the speed at which the ANN arrives at the minimum solution or converges. If the learning rate is too high the system will makes the weights and objective function diverge so there is no learning at all. In standard back propagation too low learning rate makes the network learn very slowly or to converge on the final solution. Trying to train NN using constant learning rate is usually a tedious process requiring much trial and error. The epoch parameter is defined as an epoch is one forward pass and one backward pass of all training examples. There are above 2000 records and 36 numerical input variables which can be used in this thesis. Since an ANN is dealing with a classification or prediction problems so that there are 36 output classes which can be classified according to the input variables. The accuracy of the prediction of in each fold is calculated by:

$$Accuracy = \frac{number\ of\ correct\ predicted}{length\ of\ actual\ output} \times 100$$

Mean squared error measures the average of the squares of the errors that is the difference between the estimator and what is estimated. Mean accuracy of the NN is calculated using the given formula:

$$Mean\ Accuracy = \frac{\sum(Scores)}{number\ of\ Scores}$$

Training and testing a neural network model for part of speech tagger can be done using the following procedure [38]:

1. Determine neural network size (number of input, hidden and output nodes).
2. Decide on the node activation function to be used
3. Initialize the network weights
4. Training the neural network using the back propagation algorithm and the selected training data set.
5. If training time, error goal or epochs are satisfactory, save weights.
6. If not, vary number of hidden nodes and/or re-initialize weights and go to step 4.
7. Test network
8. If the network is at expected performance level, stop it
9. If not, go to step 6.

Using this procedure the number of input layers is really just a row from our training dataset. And the numbers of input nodes are 36 because we have 36 input variables. The numbers of output nodes are 36. The network model is trained for numerous experiments by changing the parameters to get good performance.

NN parameters	n-folds = 10 l-rate = 0.2 n-epoch = 0 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1000 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 2000 n-hidden = 8
Mean Accuracy on 75% (%)	47.177%	87.752%	88.256%	89.667%	90.726%
Training Time	0:00:00.194	0:14:54.064	0:30:36.542	0:45:46.525	1:01:05.868

Table 5.3 Performances of Network Models for different values of parameters 75%

From the plot of accuracy we can see that the model could probably be trained a little more as the trend for accuracy on both datasets is still rising for the last few epochs. We can also see that the model has not yet over-learned the training dataset, showing comparable skill on both datasets.

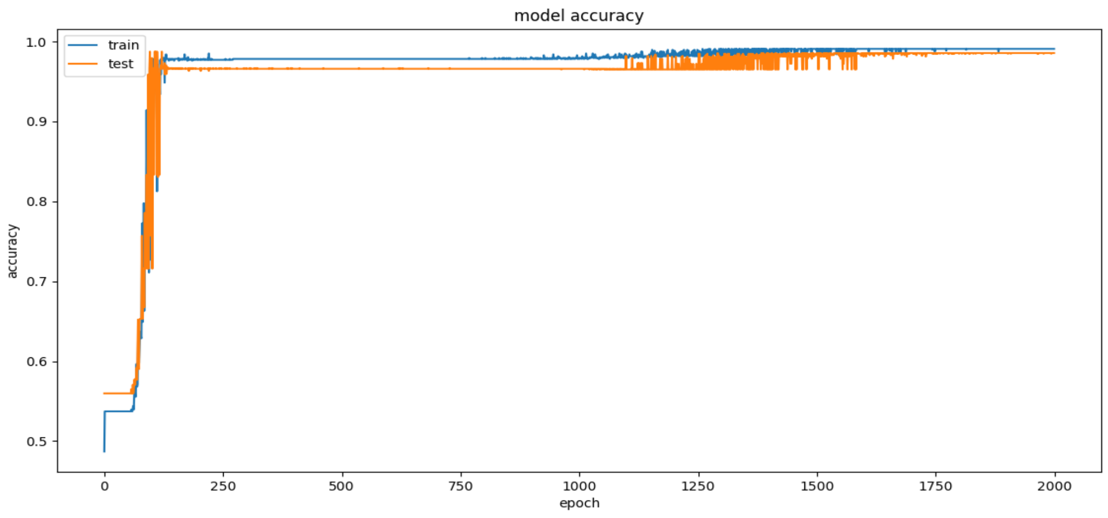

Figure 5.1 Plot of Model Accuracy on Train and Validation Datasets 75%

From the plot of loss, we can see that the model has comparable performance on both train and validation datasets (labeled test). If these parallel plots start to depart consistently, it might be a sign to stop training at an earlier epoch.

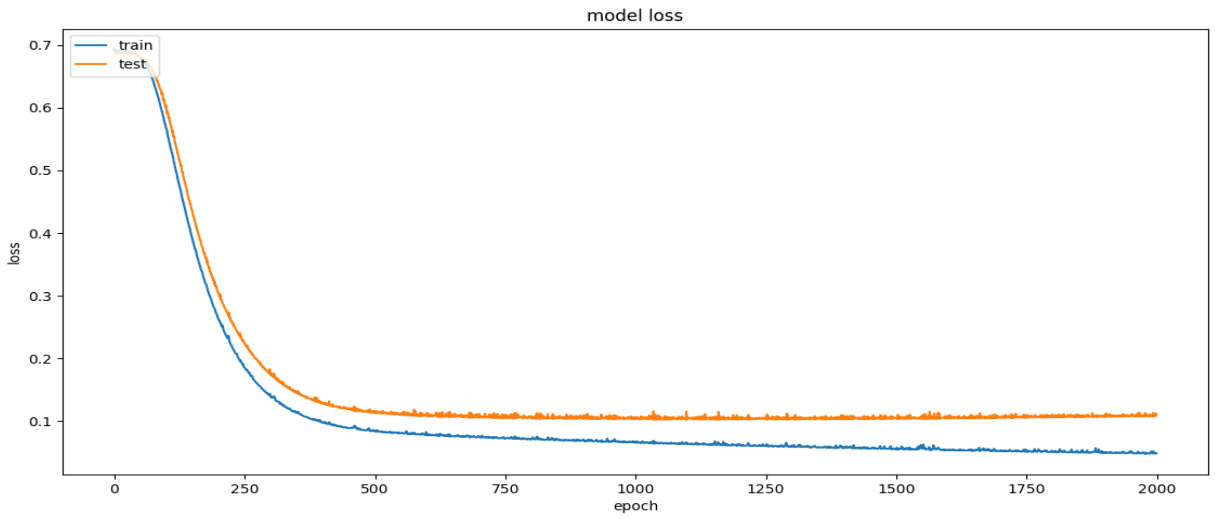

Figure 5.2 Plot of Model Loss on Training and Validation Datasets 75%

In the below table we evaluate the system using 50% of the datasets using 10-fold validation technique by changing the parameters of the neural network.

NN parameters	n-folds = 10 l-rate = 0.2 n-epoch = 0 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1000 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 2000 n-hidden = 8
Mean Accuracy on 50% (%)	54.837%	87.386%	88.301%	89.150%	90.131%
Training Time	0:00:00.187569	0:14:28.224659	0:28:59.667027	0:40:59.637500	0:54:35.085403

Table 5.4 Performances of Network Models for different values of parameters 50% of the dataset

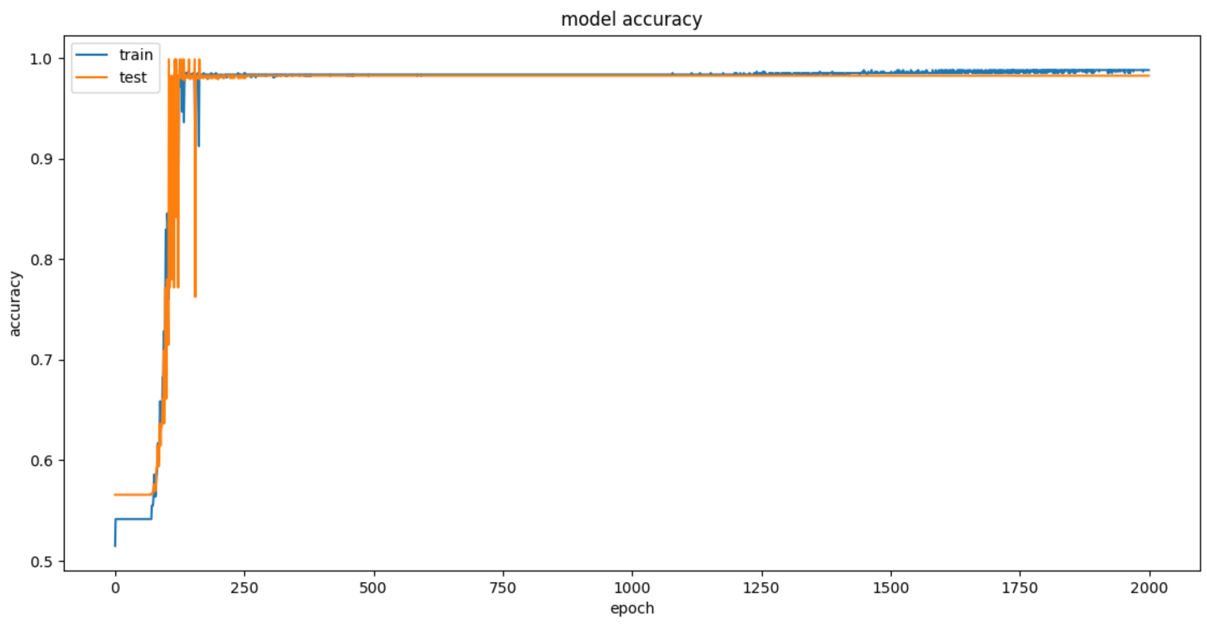

Figure 5.3 Plot of Model Accuracy on Train and Validation Datasets for 50%

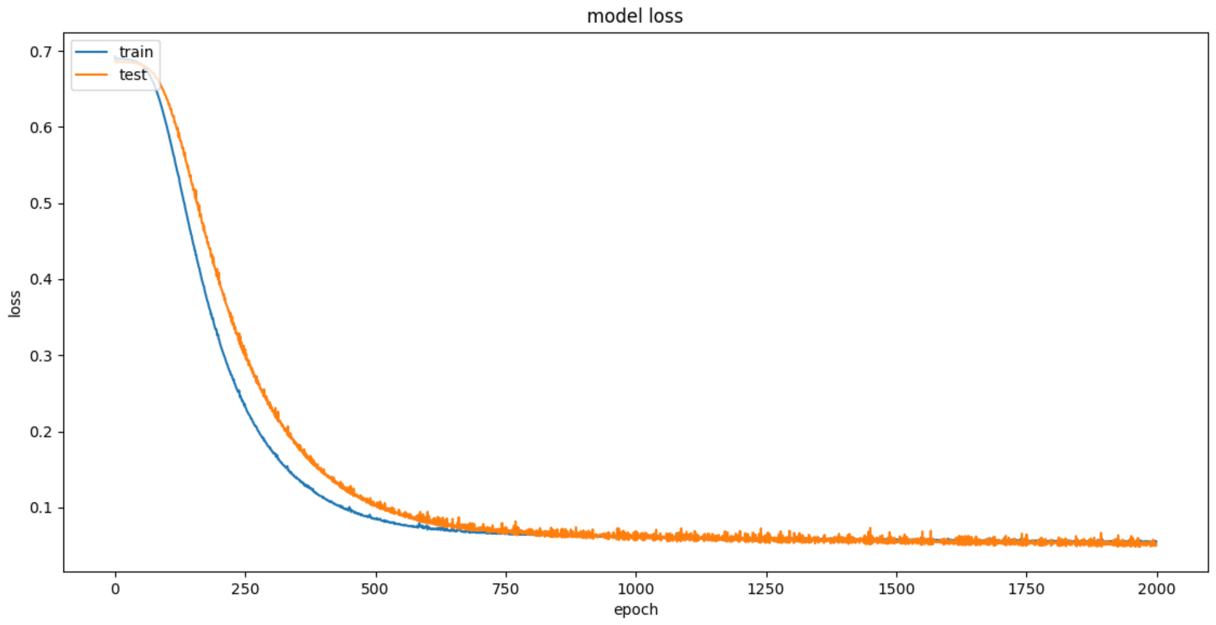

Figure 5.4 Plot of Model Loss on Training and Validation Datasets for 50%

In the below table we have tried to evaluate the system with a constant of number of folds and learning rate but by varying the number of hidden layer and number of epochs.

NN parameters	n-folds = 10 l-rate = 0.2 n-epoch = 0 n-hidden = 5	n-folds = 10 l-rate = 0.2 n-epoch = 500 n-hidden = 3	n-folds = 10 l-rate = 0.2 n-epoch = 500 n-hidden = 20	n-folds = 10 l-rate = 0.2 n-epoch = 1500 n-hidden = 16	n-folds = 10 l-rate = 0.2 n-epoch = 2000 n-hidden = 5
Mean Accuracy on 25% (%)	50.850%	86.275%	87.386%	72.941%	89.869%
Training Time	0:00:00.156264	0:06:33.883392	0:33:51.547122	1:32:36.152249	0:42:10.719122

Table 5.5 Performances of Network Models for different values of parameters 50% of the dataset

In the below table we have tried to evaluate the system using 10-fold validation technique by changing the parameters for a number of experimentation on 25% of the given dataset.

NN parameters	n-folds = 10 l-rate = 0.2 n-epoch = 0 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1000 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 2000 n-hidden = 8
Mean Accuracy on 25% (%)	53.464%	86.747%	84.789%	83.434%	85.693%
Training Time	0:00:00.010	0:04:58.050	0:09:56.549	0:14:00.741	0:20:23.287

Table 5.6 Performances of Network Models for different values of parameters 25% of the dataset

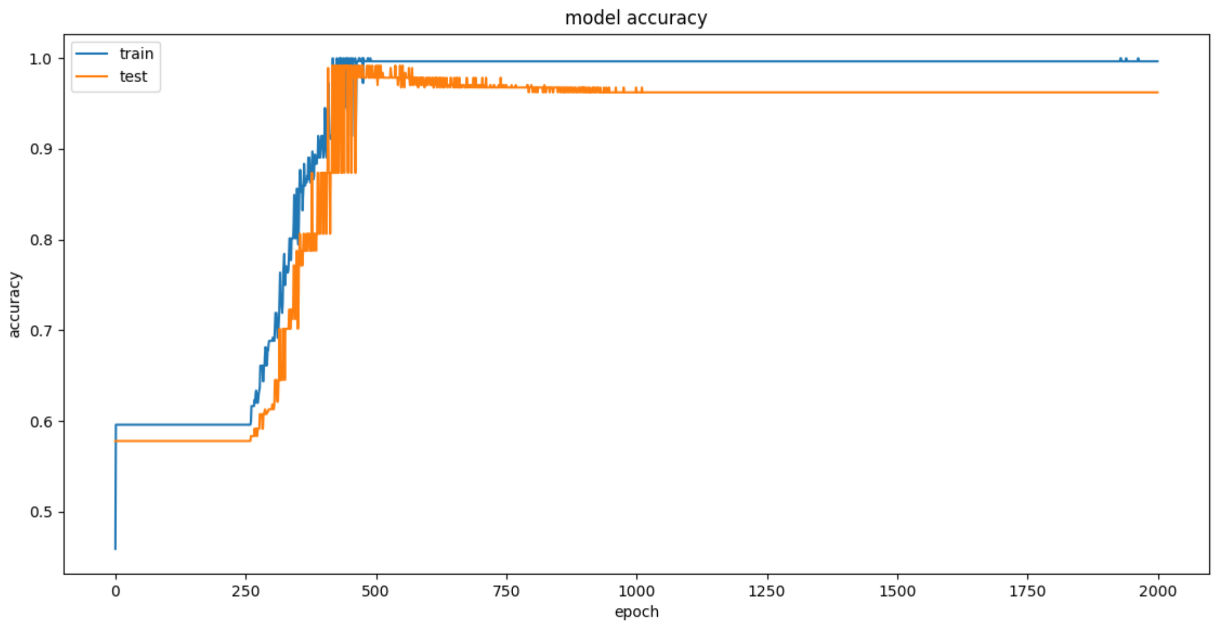

Figure 5.5 Plot of Model Accuracy on Training and Validation Datasets for 25%

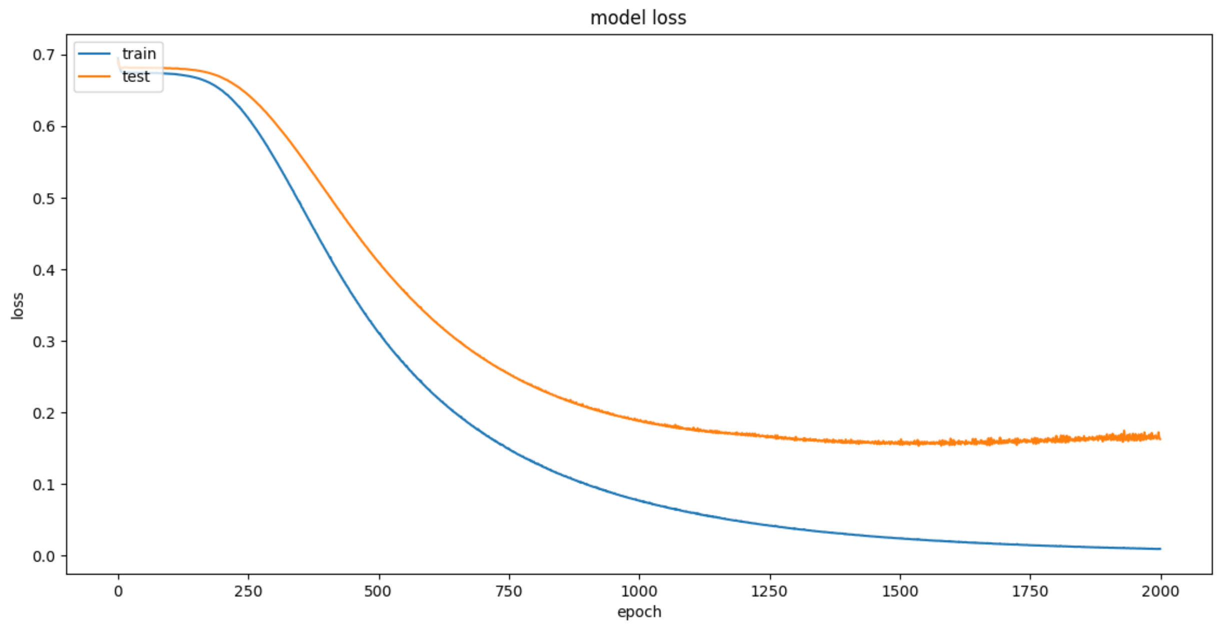

Figure 5.6 Plot of Model Loss on Training and Validation Datasets for 25%

In the below table we have tried to evaluate the system using 10-fold validation technique by changing the parameters for a number of experimentation

NN parameters	n-folds = 10 l-rate = 0.2 n-epoch = 0 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1000 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 1500 n-hidden = 8	n-folds = 10 l-rate = 0.2 n-epoch = 2000 n-hidden = 8
Mean Accuracy (%)	49.057	87.434	91.019	91.208	93.849
Training Time	0:00:00.2296	0:24:20.2344	0:49:04.4114	1:28:22.1439	1:57:19.2876

Table 5.7 Performances of Network Models for different values of parameters 100% of the dataset

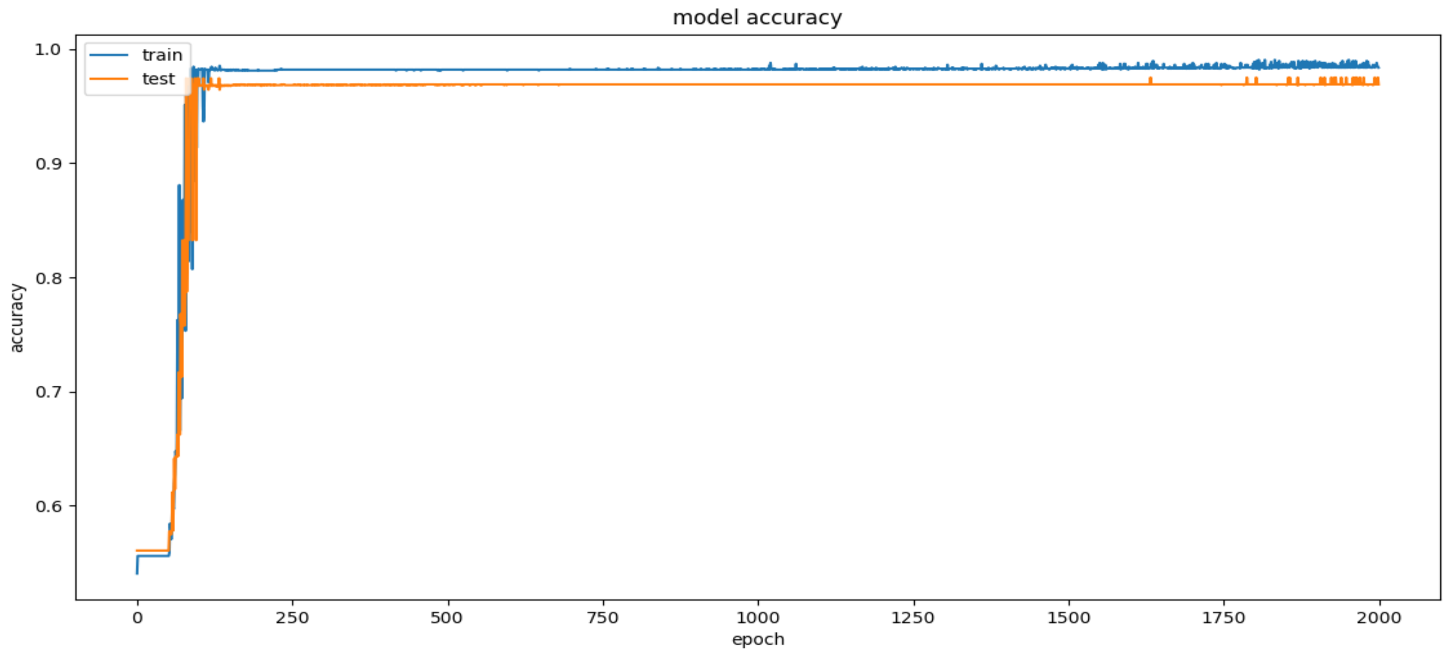

Figure 5.7 Plot of Model Accuracy on Train and Validation Datasets

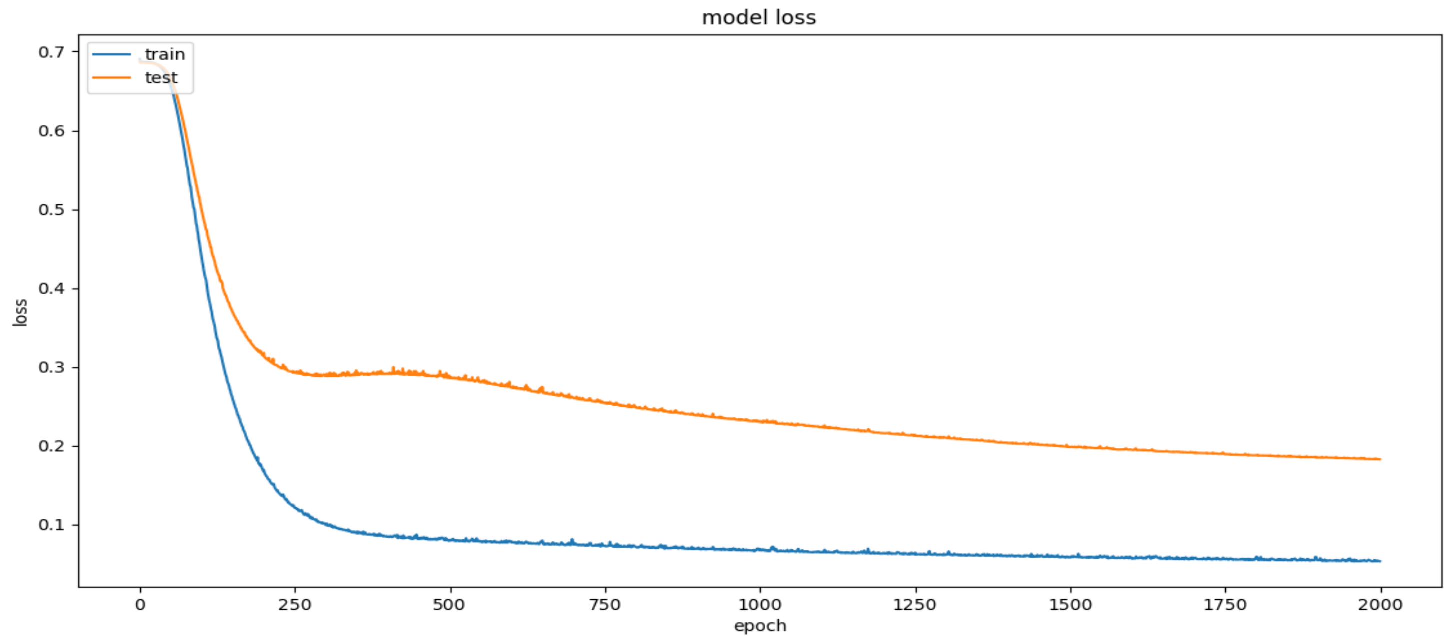

Figure 5.8 Plot of Model Loss on Training and Validation Datasets

5.4. Summary

In this chapter we make experimentation on the MLP tagger by adjusting the different parameters used in the algorithm. When we are making experiment the parameters are selected by try and error method until we get better performance. From the experimentation we take a learning rate to small value which is 0.2 and the number of neurons in the hidden layer is 8 and the number of folds is 10 in validation technique. The performance of the MPL tagger is become well when we increase the number of epochs.

The performance of the Tigrigna MLP tagger in the 75%, 50% and 25% of the given dataset is 90.726%, 90.131% and 85.693% respectively while the performance of the MLP tagger on the whole datasets which can be used 10 fold technique is 93.849% which takes a total time about 2 hour to train. In this experimentation the time elapsed to train and test the MLP is increase as the number of epochs increase.

Chapter 6

6. Conclusion and Future Works

6.1. Conclusion

Part-of-speech (POS-tagging) is considered as a process for automatically assigning the proper grammatical tag to each word of a written text according to its appearance on the text. Currently it is the hot research area because it is the first step in linguistic analysis. Part of speech tagging is an input to different NLP processes such as machine translation, spell checking and correcting, speech recognition, information retrieval information extraction, corpus analysis, syntactic parsing and text to speech synthesis. In this work we have presented artificial neural network approach with back propagation algorithm for the problem of POS tagging for Tigrigna and achieved practical accuracy of 90.726%, 90.131% and 85.693% on the 75%, 50% datasets and 25% datasets respectively using 10-fold evaluation method. While the performance of the MLP tagger on the whole datasets which can be used K fold technique is 93.849%.

In NLP standardized data corpus is an important which plays an important role in the development of POS. Because to increase the performance and accuracy of the POS tagger we need large size of corpus and with standardized tagged tag sets. In this paper we have used around 16,000 words is collected from different sources which has 36 part of speech tags are identified to annotate the corpus.

We have used python 2.7 and python 3.5 are used to implement the lexical probability and back propagation neural network. Pycharm is used as development environment for the back propagation neural network and to implement the graphical figure using the anaconda3 which is integrated with the python 3.5.

In conclusion, this research work has got encouraging results on the experimented neural network models though a large size and balanced corpus is not developed even if the time elapsed is high.

6.2. Future Works

There are many research areas on computational linguistic concerning languages spoken in Ethiopia. To avoid obstacles on this aspect, some points are recommended.

- ✓ Language resource center has to be set up by concerned bodies that will be easily accessible to researchers.

Finally this research work suggests the following suggestions as a future work:

- ✓ Extending this part of speech tagging using artificial neural network by using bi-gram, tri-gram and up to n-gram
- ✓ Extend this work by acquiring well-annotated and large size data so as to increase the efficiency.
- ✓ Comparison of two hybrid approaches: the hybrid of rule based and HMM tagger and the hybrid of rule based and ANN for Tigrigna language
- ✓ Comparison study on the three approaches (statistical, rule-based, and ANN) for other Ethiopic languages based on their language family should be done.

Reference

[1]. Teklay Gebregzabiher Abreha: **PART OF SPEECH TAGGER FOR TIGRIGNA LANGUAGE** *a thesis submitted to the school of graduate studies of Addis Ababa university in partial fulfilment of the requirement for the degree of master of science in computer science November, 2010*

[2]. Martha Yifiru Tachbelie and Wolfgang Menzel: **Amharic Part-of-Speech Tagger for Factored Language Modeling** *International Conference RANLP 2009 - Borovets, Bulgaria, pages 428–433*

[3]. Sisay Fissaha Adafre: **Part of Speech tagging for Amharic using Conditional Random Fields** *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages, pages 47–54, Ann Arbor, June 2005.*

[4]. Bjorn Gambackand, Fredrik Olsson, Atelach Alemu Argaw, and Lars Asker: **Methods for Amharic Part-of-Speech Tagging**

[5]. Martha Yifiru Tachbelie and Solomon Teferra Abate and Laurent Besacier: **Part-of-Speech Tagging for Under-Resourced and Morphologically Rich Languages – The Case of Amharic** *Conference on Human Language Technology for Development, Alexandria, Egypt 2-5 May 2011*

[6]. Bjorn Gamback: **Tagging and Verifying an Amharic News Corpus** *Workshop on Language Technology for Normalization of Less-Resourced Languages (SALTMIL8/AfLaT2012)*

[7]. Michael Gasser: **HornMorpho: a system for morphological processing of Amharic, Oromo, and Tigrinya**

[8]. Abraham Gizaw Ayana: **To Wards Improving Brill's Tagger Lexical and Transformation Rule for Afaan Oromo Language** *Open Access rec: 10 Jul 2015, publ: 10 Jul 2015*

[9]. Omer Osman and Ibrahim Yoshiki Mikami: **Stemming Tigrinya Words for Information Retrieval** *Proceedings of COLING 2012: Demonstration Papers, pages 345–352, COLING 2012, Mumbai, December 2012.*

[10]. Binyam Gebrekidan Geber: **PART OF SPEECH TAGGING FOR AMHARIC**

[11]. Schmid H. “**Part-of-Speech Tagging With Neural Networks**”, Proc. COLING’94, Kyoto, Japan, pp. 172-176, 1994.

[12]. Ankur Parikh: **Part-Of-Speech Tagging using neural network** *International Conference on Natural Language Processing (ICON-2009)*

[13]. Salvador Tortajada Velert, Marıa Jose Castro Bleda, and FerranPla Santamarıa: **Part-of-Speech tagging based on artificial neural networks**

[14]. Nuno C. Marques and Gabriel Pereira Lopes: **Neural Networks, Part-of-Speech Tagging and Lexicons** International Conference on Natural Language Processing (ICON-2009)

[15]. Daniel Jurafsky and James H. Martin: **Speech and Language Processing**, **an Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition** *Draft of September 28, 1999.*

[16]. Anne Kao and Steve Poteet: **Text Mining and Natural Language Processing – Introduction for the Special Issue** *Volume 7, Issue 1*

[17]. HASAN MUAIDI: **Levenberg-Marquardt Learning Neural Network For Part-of-Speech Tagging of Arabic Sentences** *Volume 13, 2014*

[18]. http://www.omniglot.com/writing/tigrinya.htm visited on November 3, 2015

[19]. Elizabeth D. Liddy: **Natural Language Processing** – 2001

[20] Getachew Mamo: **Part Of Speech Tagger for Afaan Oromo Language** *a thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement for the degree of Master of Science in information science January, 2009*

[21] Anish A: **Part of Speech Tagger for Malayalam** *in partial fulfillment of the requirement for the award degree of Master of technology in computational engineering and networking amrita school of engineering, COIMBATORE, July 2008*

[22] https://user.phil-fak.uni-duesseldorf.de/~bontcheva/SS10CTCL/CTCL-IntroNotes.pdf visited on May 27, 2016

[23] Martin Frodl: **Part-of-Speech Tagging Using Neural Networks** *a thesis submitted to the Masaryk University Faculty of Informatics Master's thesis Brno, autumn 2013*

[24] Jochen Frohlich, Neural Net Components in an Object Oriented Class Structure

[25] Teferi Kebebew: **Speech Recognition for Afan Oromo Using Hybrid Hidden Markov Model and Artificial Neural Network** *a thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement for the degree of Master of Science in information science June, 2010*

[26] Eskinder Mesfin Cherinet: **application of multilayer feed forward artificial neural network perceptron in prediction of court case's time span: the case of federal supreme courts'** *a thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement for the degree of Master of Science in information science January, 2009*

[27] Gebrehiwot Assefa Berhe: **A Two Step Approach for Tigrigna Text Categorization** *a thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement for the degree of Master of Science in information science June, 2011*

[28] Richard P. Lippmann, (1988). **An Introduction to Computing with Neural Nets**, *IEEE ASSP Magazine, Vol. 4 (1987), pp: 4-22, at pp: 15-18; DARPA, Darpa Neural Network Study (Fairfax, VA: Armed Forces Communications and Electronics Association [AFCEA] International Press), pp: 78-80*

[29] Simon Haykin, (1994). **Neural Networks: A Comprehensive Foundation**, *2nd ed. Prentice Hall PTR, Upper Saddle River, NJ.*

[30] Ravi Narayan, S. Chakraverty, and V. P. Singh. **Neural Network based Parts of Speech Tagger for Hindi,** *Third International Conference on Advances in Control and Optimization of Dynamical Systems March 13-15, 2014. Kanpur, India*

[31] Anderson D. and McNeil, G. (1992). **Artificial Neural Network Technologies: data analysis for software**. *August 20 1992 Rome, NY: Rome Laboratory*.

[32] S.M Sapuan and I.M Mujtab 2010, **composite materials Technology: Neural Network Applications** *CRC Press, USA*

[33] Joe Tebleskis, May 1995, **Speech recognition using Neural Networks**, *PhD Dissertation, Carnegie Mellon University, Pennsylvania*

[34] A. Ukil, 2007, **Intelligent systems and signal processing in power engineering**, *Springer*

[35] Surinder Dhanjal, 2001, **Artificial Neural networks in speech processing: problems and challenges**, *IEEE, pp. 510-513*

[36] Mirza Cilimkovic: **Neural Networks and Back Propagation Algorithm** *Institute of Technology Blanchardstown Blanchardtown Road North Dublin 15 Ireland*

[37] Ajith Abraham: **Artificial Neural Networks** *Oklahoma State University, Stillwater, OK, USA*

[38] Solomon Asres Kidanu: **Amharic Part-of-Speech Tagging Using Hybrid Approach (Neural Network and Rule-Based)** *a thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement for the degree of Master of Science in Computer Science September, 2008*

[39] Ahmed. 2002. **Application of multilayer perceptron network for tagging parts-of-speech,** proceedings of the Language Engineering Conference, IEEE.

[40] Tesfaye Tewolde (PhD), (2002), **a modern grammar of Tigrigna,** Tipografia U. Detti – via G. Savonarola Roma.

[41] https://en.m.wikipedia.org/wiki/Tigrinya_grammar visited on July 22, 2016

[42] Hardie A., **Developing a Tag set for Automated Part-of-Speech Tagging in Urdu.** *Proceedings of the Corpus Linguistics 2003 conference. UCREL Technical Papers Volume 16. Department of Linguistics, Lancaster University, 2003.*

[43] Girma Awgichew & Mesfin Getachew, (2006). **Manual Annotation of Amharic News Items with Part-of-Speech Tags and its Challenges,** *ELRC Working Papers Vol. 2; number 1: AAU Printing Press.*

[44] Q. Ma and H. Isahara, "**A Multi-Neuro Tagger Using Variable Lengths of Contexts**", Proceedings of COLINGACL 98, Montreal, pp. 802-806, 1998.

[45] Karel Oliva and Pavel Kvìtoò, "**Linguistically Motivated Bigrams in Part-of-Speech Tagging of Language Corpora**", the Prague Bulletin of Mathematical Linguistics 78, 2002.

[46] Mao Yonghang. “**Natural Language Processing Model (part of Speech Tagging and Sentences parsing)”.** Laboratory manual, 1997.

[47] Guilder Linda V., **“Automated Part Of Speech Tagging: A Brief Overview”,** At: Vanguill @ gusun.Georgetown.edu, 1995, last visited April 2007.

[48] M. Civit and M.A. Martí, “**Design Principles for a Spanish Treebank**” In Proceedings of the Conference: Treebanks and Linguistic Theories. Sozopol, Bulgaria, September 2002.

[49] Qing Ma, Masaki Murato, Kiyotaka Uchimoto, and Hitoshi Isaharc, “**Hybrid Neuro and Rule-Based Part of Speech Taggers**”, Communication research, Laboratory, Japan, 2000.

[50] Johan Carlberger, Viggo Kann, “**Implementing an Efficient Part-of-Speech Tagger**”, John Wiley & Sons, Inc. New York, USA, Vol. 29, pp. 815-832, July 1999.

[51] Levent Altunyurt & Zihni Orhan, “**Part of Speech Tagger for Turkish**”, Bogaziçi University, June 2006.

[52] https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/ visited on August 10, 2016

[53] Mulgeta Atsbaha Siyum: **Automatic Part-of-Speech Tagger For Tigrigna Language Using Hybrid Approach (Neural Network and Rule-Based)** *a thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement for the degree of Master of Science in Information Science October, 2016*

Appendices

Apendex1

Sample Manually Annotated Corpus:

ኣብ/PREP ወረዳ/N ወልቃይት/NP ዕርየት/N ትምህርቲ/N ኣብ/PREP ምርግጋፅ/N ዘተባብዑ/VREL ለውጥታት/N ይምዝገቡ/V ምህላዎም/N ተገሊፁ/V ።/PUNC

ቤትፅሕፈት/N ርክብ/N ህዝብን/NC መንግስትን/NC እታ/PROND ወረዳ/N ከምዝገለፆ/VREL ድሕሪ/C እቲ/PROND ኣብ/PREP ፅርየት/N ትምህርቲ/N ኣድሂቡ/V ብደረጃ/NPREP ወረዳ/N ዝተሳለጠ/VREL ኮንፈረንስ/N ኣብ/PREP ምርግጋፅ/N ፅርየት/N ትምህርቲ/N እንታይ/PRONI ዓብዪ/ADJ ዘተታት/N ኣመዝጊብና/V ዝብል/VREL ሰፊሕ/ADJ ገምጋማዊ/ADJ መድረኽ/N ድማ/C ልዕሊ/PREP 100/CARDN ርእሳነመምህራንን/NC ሱፐርቫይዘራትን/NC ተሳቲፎም/V እዮም/AUX ።/PUNC

ብመሰረት/C እዚ/PROND ድሕሪ/C እቲ/PROND ዝተሳለጠ/VREL ኮንፈረንስ/N ሞዴል/N ክፍልታትን/NC መምህራን/NC ከምዝተፈጠሩን/VRELC ብዓቕሚ/NPREP ንዝተሓቱ/VRELPREP ተምሃሮ/N ፍሉይ/ADJ ሓገዝ/N ይዋሃቦም/V ከምዘሎን/VRELC ብምሕባር/ADV ብሓፈሻ/ADV ፅርየት/N ትምህርቲ/N ንምርግጋፅ/VI ዘተባብዑ/VREL ጅማሮታት/N ይምዝገቡ/V ኣለዉ/AUX ።/PUNC

Sample Word Dictionary:

ኣብ/PREP

ወረዳ/N

ወልቃይት/N

ፅርየት/N

ትምህርቲ/N

ኣብ/PREP

ምርግጋፅ/N

ዘተባብዑ/VREL

ለውጥታት/N

ይምዝገቡ/V

ምህላዎም/N

ተገሊፁ/V

።/PUNC

ቤትፅሕፈት/N

ብደረጃ/NPREP

ርክብ/N

ህዝብን/NC

መንግስትን/NC

እታ/PROND

ወረዳ/N

ከምዝገለፆ/VREL

ድሕሪ/C

እቲ/PROND

ኣብ/PREP

ፅርየት/N

ትምህርቲ/N

ኣድሂቡ/V

Sample Lexical Probability

ኣብ/PREP 0.00196850393701

ወረዳ/N 0.0159798149706

ወልቃይት/N 0.000841042893188

ፅርየት/N 0.00420521446594

ትምህርቲ/N 0.0126156433978

ኣብ/PREP 0.545275590551

ምርግጋፅ/N 0.00294365012616

ዘተባብዑ/VREL 0.00340715502555

ለውጥታት/N 0.000420521446594

ይምዝገቡ/V 0.00347222222222

ምህላዎም/VREL 0.00170357751278

ምህላዎም/N 0.000420521446594

ተገሊፁ/V 0.0121527777778

።/PUNC 0.767624020888

ቤትፅሕፈት/N 0.00798990748528

ርክብ/N 0.00378469301934

ህዝብን/NC 0.0350877192982

መንግስትን/NC 0.0315789473684

እታ/PROND 0.165680473373

ወረዳ/N 0.0159798149706

ከምዝገለፆ/VREL 0.00681431005111

ድሕሪ/C 0.0745341614907

እቲ/PROND 0.402366863905

እቲ/PREP 0.00984251968504

ኣብ/PREP 0.545275590551

ፅርየት/N 0.00420521446594

ትምህርቲ/N 0.0126156433978

ኣድሂቡ/V0.00347222222222

Sample Preprocessed Lexical Corpus Used for Artificial Neural Network

word/tag	N	NP	NC	NPREP	NPREPC	NPOSS	PRON	PRONPREP	PRONC
ኣብ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ወረዳ	0.015979	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ወልቃይት	0.000841	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ፅርየት	0.004205	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ትምህርቲ	0.012616	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ምርግጋፅ	0.002943	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ዘተባብዑ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ለውጥታት	0.000421	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ይምዝገቡ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ምህላዎም	0.000421	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ተገሊፁ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
::	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ቤትፅሕፈት	0.007989	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ርክብ	0.003785	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ህዝብን	0.000001	0.000001	0.035088	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
መንግስትን	0.000001	0.000001	0.031578	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
እታ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ከምዝገለፆ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ድሕሪ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
እቲ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ኣድሂቡ	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001	0.000001
ብደረጃ	0.000001	0.000001	0.000001	0.039215	0.000001	0.000001	0.000001	0.000001	0.000001

Appendex2

Pseudo Code for Sample Steps of Back Propagation Algorithm

1. **Initialize a Network**

```
def initialize_network(inputs, hidden, outputs):
    network = list()
    for i in range(inputs + 1):
        for i in range(hidden):
            wieghts = random()
            network.append(hidden)
                                end for
    end for
    for i in range(hidden + 1):
        for i in range(outputs):
            wights = random()
          network.append(outputs)
        end for
    end for
    return network
```

2. Forward Propagate

```
# calculate neuron activation for an input

def activate(weights, inputs):

    activation = weights[-1]
    for i in range(len(weights) - 1):
        activation += weights[i] *
inputs[i]
    end for
    return activation
# Transfer neuron activation
def transfer(activation):
    return 1.0 / (1.0 + exp(-
activation))
# Forward propagate input to a network
output
def forward_propagate(network, row):
```

```
inputs = row
    for layer in network:
        new_inputs = []
        for neuron in layer:
            activation =
activate(neuron['weights'], inputs)
            neuron['output'] =
transfer(activation)

new_inputs.append(neuron['output'])
        end for
        inputs = new_inputs
    end for
    return inputs
```

3. Back Propagate Error

```
# Calculate the derivative of an neuron
output
def  transfer_derivative(output):
    return output * (1.0 - output)
# Back propagate error and store in neurons
def backward_propagate_error(network,
expected):
 for i in
reversed(range(len(network))):
   layer = network[i]
    errors = list()
     if i != len(network) - 1:
      for j in range(len(layer)):
        error = 0.0
         for neuron in network[i + 1]:
          error += (neuron['weights'][j] *
neuron['delta'])
```

```
               errors.append(error)
             end for
          end if
         else:
           for j in range(len(layer)):
  neuron = layer[j]
            errors.append(expected[j] -
neuron['output'])
           end for
         for j in range(len(layer)):
           neuron = layer[j]
             neuron['delta'] = errors[j] *
transfer_derivative(neuron['output'])
         end for
      end for
```

4. Train Network

```
# Update network weights with error
def update_weights(network, row,
l_rate):
    for i in range(len(network)):
        inputs = row[:-1]
        if i != 0:
            inputs = [neuron['output']
for neuron in network[i - 1]]
        end if
        for neuron in network[i]:
            for j in
range(len(inputs)):
                neuron['weights'][j] +=
l_rate * neuron['delta'] * inputs[j]
            neuron['weights'][-1] +=
l_rate * neuron['delta']
            end for
        end for
    end for
# Train a network for a fixed number
of epochs
def train_network(network, train,
l_rate, n_epoch, n_outputs):
            for epoch in
range(n_epoch):
        sum_error = 0
    for row in train:
            outputs =
forward_propagate(network, row)
            expected = [0 for i in
range(n_outputs)]
            expected[row[-1]] = 1
            sum_error +=
sum([(expected[i] - outputs[i]) **
2 for i in range(len(expected))])

backward_propagate_error(network,
expected)

update_weights(network, row,
l_rate)
        end for
```

Declaration

This thesis is my original work and has not been submitted as a partial requirement for a degree in any university

---

DAWIT TEKIE

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

Mr. GETACHEW M. (Ass Prof)