Improving Performance of Word Sense Disambiguation Model using Part-of-Speech Tagged Dataset for Tigrigna Text: Machine Learning Approach

Ngus Hailemariam
HiLCoE, Computer Science Programme, Ethiopia
Gambella ICT Agency, Gambella, Ethiopia
ngus.hmariam@yahoo.com

Solomon Teferra
HiLCoE, Ethiopia
School of Information Science, Addis Ababa University, Ethiopia
solomon_teferra_7@yahoo.com

Abstract

The need to use information technology grows rapidly in our day to day life. However, in Natural Language Processing (NLP), the existence of ambiguous words is a hindrance to effectively implement applications such as Machine Translation (MT), Information Retrieval (IR), etc. This is due to the inability of machines to discriminate the exact senses of ambiguous words. Word Sense Disambiguation (WSD) is the task of assigning the correct sense of ambiguous words based on the surrounding context. The objective of this paper is to design WSD prototype model for Tigrigna text with the intention of investigating the effect of part-of-speech tagged corpus and improve the performance of the model.

The study employed an experimental research on a corpus of 1250 sentences for only five selected ambiguous words namely *kefele* (ክፈለ), *OareQe* (ዓረቐ), *seOare* (ሰዓረ), *medeb* (መደብ), and *Halefe* (ሓለፈ) using part-of-speech tagged corpus to develop the model. We applied preprocessing techniques using Python 3.6.1 programming language to make the dataset clean and transliterate from Tigrigna into Latin characters to be compatible with the machine learning tool, Weka, used to develop the WSD model. Meanwhile, 10-fold cross validation was used to evaluate the classification algorithms which uses each data point once for testing and 9 times for training. The finding using semi-supervised learning on part-of-speech tagged corpus shows performance improvement using ADTree, AdaBoostM1, Bagging, and Naïve Bayes, respectively as compared to the baseline result. We concluded that we have got an improved WSD model for Tigrigna using semi-supervised learning method on the part-of-speech tagged corpus.

Keywords: Ambiguity; Machine Learning; NLP; Part-of-Speech Tagged Corpus; WSD

1. Introduction

These days the availability and use of information technology has been increasing from time to time. In order for the available information to be useable, a need has emerged on how to use technology aiming at NLP, has come up with an indispensable focus of natural language computations in response of such a need. NLP facilitates human to machine interaction. However, ambiguity is challenging in natural language in which words denote different meanings in different contexts. Natural language is highly ambiguous and computers do not have the common sense that enables real sense understanding [1]. Even if a human has an inborn capability to distinguish multiple senses of an ambiguous word in a particular context, it is very difficult for machines. As many other languages, Tigrigna is influenced by ambiguity.

Tigrigna is a Semitic language widely spoken in Tigray region of Ethiopia and Eritrea. It is the official language of Tigray region in Ethiopia and is the language most widely spoken in this region. It has more than six million speakers worldwide [2, 3]. The number of speakers of the language is increasing from time to time and Tigrigna electronic documents are also increasing. Though, to effectively utilize NLP applications such as MT, IR, etc., ambiguity is a hindrance. There are many ambiguous words in Tigrigna. For instance, the Tigrigna word "ቀረበ" can be translated into English as "he brought forth" or "he came closer" [4]. In Tigrigna, ambiguity can

emerge as Phonological, Referential, Structural, Semantic, Orthographic, and Lexical ambiguities.

Computationally, the correct sense of an ambiguous word can be distinguished based on the context in which the word is used which is known as disambiguation. As the process of disambiguation relies on the context of the target word, WSD is an NLP task concerned with computationally determining which sense of the ambiguous word has been used in a given context. Historically, WSD is one of the oldest problems in computational linguistics [5]. It was considered as a fundamental task of MT already in the late 1940s [6].

Research works for WSD have been conducted in other languages. For Tigrigna, there are only inceptions of WSD researches that attempt using unsupervised and semi-supervised learning for the language. Although these research works had laid a ground to WSD prototype model development for Tigrigna, the performance achieved is below the tolerance of utilizing WSD systems. This paper is a corpus based approach attempting to explore supervised, unsupervised and semi-supervised learning which is intended to investigate the effect of part-of-speech tagged corpus and develop WSD model using the selected method that best improves the performance of WSD for Tigrigna text, in this case, semi-supervised learning. The study differs from other Tigrigna works since it uses linguistic feature of part-of-speech tag information to improve WSD performance and add value to future real world applicable WSD systems of the language.

1.1 Approaches (Methods) To WSD

Most current WSD systems are developed using either knowledge based or corpus-based approaches. Corpus-based (machine-learning) approach uses a corpus and train the system to perform the task of WSD [7, 8]. After the model is trained on numerous examples, it is tested on unseen examples to determine the effectiveness of the trained classifier. On the other hand, knowledge based approach uses external lexical resources like dictionary and Thesauri [9]. WSD is an “intermediate task” which is not an end in itself, but rather is necessary at one level for other NLP tasks like MT, IR and hypertext navigation, grammatical analysis, speech processing and text processing [10].

a. *Supervised Machine Learning:* In this approach, a training corpus is required, where the system learns to classify and a test corpus which the system must annotate [7]. The training set typically contains a set of examples in which a given target word is manually tagged with a sense from the sense inventory of a reference dictionary. So, Supervised Machine Learning suffers from knowledge acquisition bottleneck. In this paper, we manually labelled the experimental corpus to conduct the experiment of supervised learning.

   Naïve Bayes classifier is a probabilistic classifier and one of the supervised learning algorithms that classifies text documents using two parameters: the conditional probability of each sense of a word and the features in the context.

b. *Unsupervised Machine Learning*: This approach is based on the idea that the same sense of a word will have similar neighboring words. So, clustering technique is used in this method. Clustering is the process of grouping the data into classes or clusters, so that objects within a cluster have high similarity [11]. Unsupervised WSD methods are based on unlabelled corpora, and do not exploit any manually sense-tagged corpus to provide a sense choice for a word in context [12]. So, they have the potential to overcome the knowledge acquisition bottleneck [13]. Even if this method does not require human effort for corpus annotation, it provides worse performance than that of the supervised ones [14].

c. *Semi-supervised Learning*: Is a method that can learn from fewer labelled data points with the help of a large number of unlabelled data points [15]. It requires only a small amount of labelled training data and is sometimes able to improve performance using unlabelled data [15]. Supervised and unsupervised machine learning are combined to balance their drawbacks [7, 14]. The unsupervised part is usually applied first to the data in order to make some assumptions about the distribution of the data, and then these assumptions are reinforced using a supervised one [14]. In this paper, we incorporated unlabelled

data by combining some seed instances directly into the data representation (features) so that unsupervised algorithms (in this case, Expectation Maximization Clustering Algorithm) are directly applied for clustering and supervised algorithms are applied for classification after the unlabelled data are given their labels using the cluster labels found during clustering.

As it has been explained in [16], *bootstrapping* looks like supervised approaches, but it needs only a few seeds instead of a large number of training examples. The charm of this approach lies in its continuous optimization of the trained model until it reaches convergence. As it can be seen in Figure 1 [16], it recursively uses the trained model to predict the sense of new cases and in return optimizes the model by new predicted cases.

The following are bootstrapping algorithms used in this paper.

i. *AdaBoost (AB)*: is a general method for obtaining a highly accurate classification rule by combining many weak classifiers, each of which may be only moderately accurate [17]. In AdaBoost, the aim is to combine different weak learners or classifiers together to improve the classification performance.

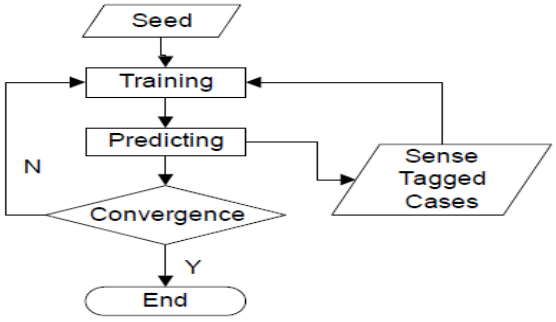

Figure 1: Flow of Recursive Optimization Algorithm

ii. *Bagging*: or bootstrap aggregation is a specific type of machine learning that uses ensemble learning to evolve machine learning models [18]. The idea of bagging has been used extensively in machine learning to create better fitting for models. The idea is that if one takes several independent machine learning units, they can function collectively better than one unit that would have more resources.

iii. *ADTree*: An Alternating Decision Tree (ADTree) is a generalization of Decision Tree and vote-stumps [19]. In addition to Decision Tree, ADTree consists of alternating binary Decision nodes and prediction leaves. Each leaf is a logical formula of the pathway node decisions with weights. An instance is classified by an ADTree by following all paths for which all decision nodes are true and summing any prediction nodes that are traversed. An instance is scored by summing the scores of all decisions. Since ADtree has a reduced training cost and a reduced computation cost with respect to Adaboost, it is comparably applied for real-time applications as Adaboost [14].

1.2 Evaluation Techniques to WSD

In this paper, classification performance measure is used to evaluate classification algorithms whereas clustering performance measure is used to evaluate clustering algorithms. Among the available classification evaluation modes in Weka, 10-fold cross validation has been used. The 10-fold cross validation is the default evaluation mode in Weka which iteratively experiments after dividing the dataset into ten mutually exclusive folds. Then it averages the results obtained in each trial. In 10-fold cross validation, each data point will be used once for testing and nine times for training.

1.3 Overview of Tigrigna Language

Tigrigna is a Semitic language that has its own writing system, Ethiopic writing system, and its own characters (alphabets), punctuation marks, and number systems [20]. It has 35 base symbols with seven orders. Each Tigrigna language script occurs in one basic form and six other forms that may be described as orders. The Tigrigna writing system can be transliterated into corresponding Latin characters using system for Ethiopic representation in ASCII (SERA) [21]. For instance, the Tigrigna letter ‘ሀ‘ and its seven orders can be represented using Latin characters as shown in Table 1.

Table 1: Sample example of Tigrigna letters and their corresponding Latin letters

Order	*1st*	*2nd*	*3rd*	*4th*	*5th*	*6th*	*7th*
Letter in Tigrigna	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
Corresponding Latin letter	Be	Bu	Bi	Ba	Bie	B	Bo

Tigrigna is morphologically rich which makes use of prefixing, suffixing and internal changes to form inflectional and derivational word forms. Hence, a given root word can result in a very large number of variant forms [22]. Besides, Tigrigna has its own syntactic structure which exhibits a unique syntax; which is SOV (Subject-Object-Verb) [22].

2. Related Work

A number of research works on WSD have been done in other languages, including Ethiopian languages. However, in this section, we focused only on WSD researches directly related to Tigrigna.

The work in [23] uses unsupervised machine learning of corpus-based approach which requires information that can be automatically extracted from untagged text. A total of 631 sense examples for four of the ambiguous words namely ሓለፈ (*Halefe*), መደብ (*medeb*), ሃደመ (*hademe*), and ከበረ (*kebere*) have been used. The author explored if stemming can enhance the performance of Tigrigna WSD using unsupervised learning. In determining the standard window size using the selected clustering algorithms, window size of 2-2 was found to be the optimal window size.

The other work in [24] also attempted WSD research for Tigrigna using semi-supervised learning method that uses few labelled and more unlabelled datasets. In the study, 1250 sentences for five ambiguous words: *kefele* (ከፈለ), *OareQe* (ዓረቐ), *seOare* (ሰዓረ), *Halefe* (ሓለፈ), and *Medeb* (መደብ) have been used. The author compared three machine learning methods and found out that semi-supervised machine learning achieved the best result. Hence, to develop the model, ADTree, AdaBoostM1, Bagging, SMO, and Naïve Bayes were used. The author also reported that window size of 1-1 is the optimal window size using semi-supervised learning.

Both of the WSD works, the unsupervised learning used in [23] and semi-supervised learning done in [24] used corpus based approach and achieved very encouraging result. However, due to bootstrapping nature of semi-supervised learning, better performance accuracy was achieved by the semi-supervised approach [24]. But, still the result obtained is below the tolerance of utilizing the WSD systems. Since different studies recommended [25, 24] that adding part-of-speech (POS) tag information into linguistic corpus improves performance of WSD, this paper employs part-of-speech tagged corpus to develop WSD model for Tigrigna which is different from the works in [23, 24].

3. The Proposed Solution

System architecture is the conceptual model that defines the structure, behavior, and views of a system. Figure 2 presents the system architecture for the three machine learning paradigms.

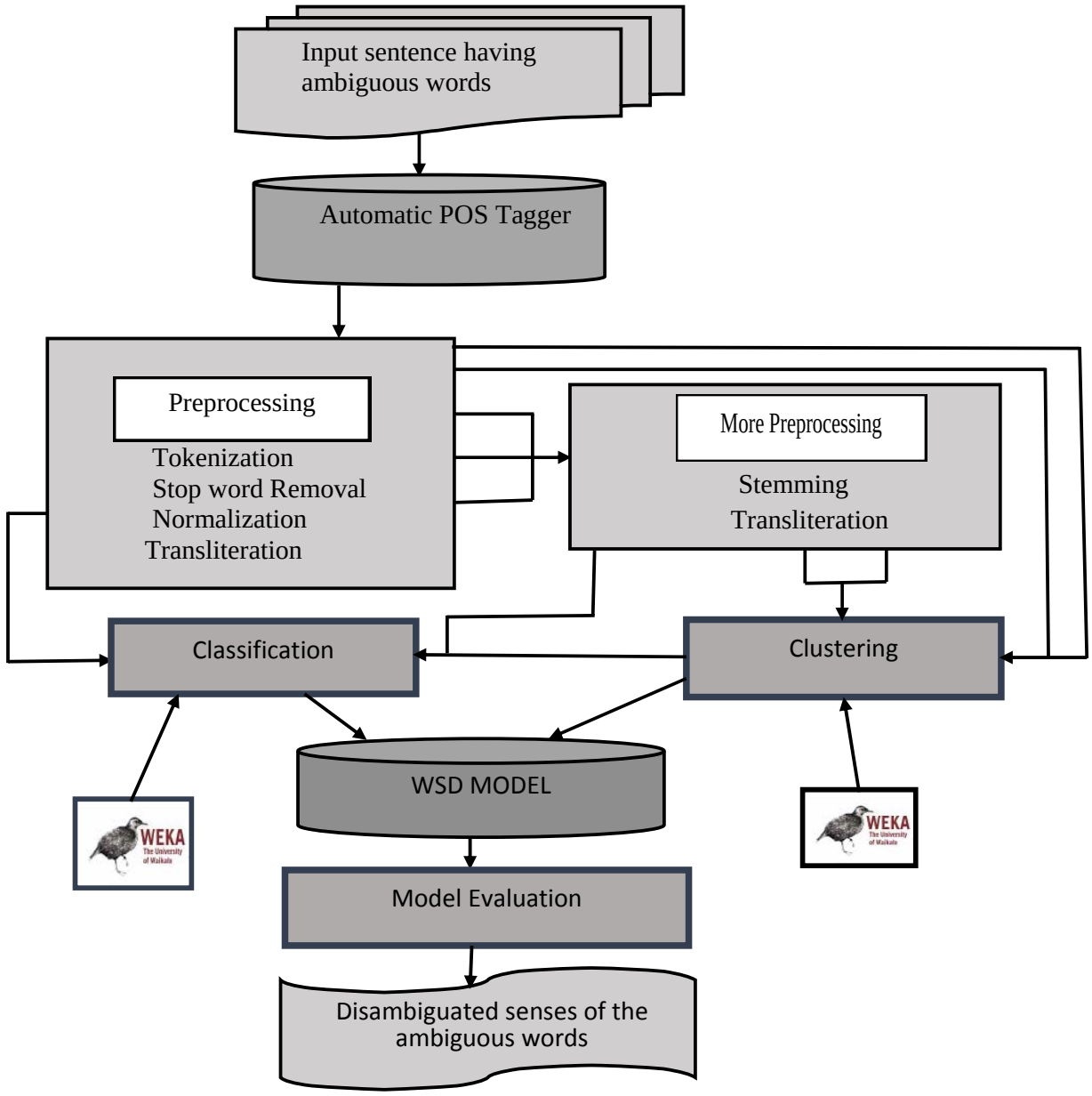

Figure 2: System Architecture of Tigrigna WSD using the three machine learning methods when part-of-speech tag information is incorporated with stemmed and unstemmed corpus

3.1 Corpus Preparation (acquiring)

Corpus based WSD requires a collection of text data called corpus used for the purpose of linguistic processing and analyzing. In this paper, we used the corpus in [24]. Only part-of-speech tag information was attached to this linguistic corpus. Table 2 presents selected ambiguous words with their sense examples [24].

Table 2: Selected ambiguous words with their corresponding number of sentences

Ambiguous word	*Senses*	*No. of Sense examples*	*Total Sentences*
ሰዓረ	Winning Not Fasting	103 101	204
ከፈለ	Pay Divide	102 102	204

Ambiguous word	*Senses*	*No. of Sense examples*	*Total Sentences*
ዓረቐ	Negotiate	82	164
	Poverty	82	
መደበ	Plan	102	274
	Grouping or Assigning	102	
	Traditional Bed	70	
ሓለፈ	Pass	101	404
	Pass Away (Die)	101	
	Promote	101	
	Lead	101	
Total			*1250*

3.2 Using part-of-speech tagger

Part-of-speech tagging refers to associating word class markers to sentence members in particular and attaching additional information about corpora contents in the forms of tag label [26]. It provides significant information about a word and its neighbor. It is used as a preprocessing for other NLP applications, and in our case we used it for WSD. Word category of one language may be different from that of another language. In this paper, all Tigrigna words in the experimental datasets, except the target word, were automatically tagged using the Tigrigna Automatic part-of-speech Tagger developed in [26].

3.3 Preprocessing

Preprocessing is the primary step to transform the data into a format that will be more easily and effectively processed and make the experimental data sets compatible with the machine learning tool, Weka. In this paper, tokenization, stop word removal, normalization, and stemming were applied. Finally, to make the experimental datasets compatible with Weka, transliteration was done. For all algorithms Python programming language was used.

3.4 Preparing datasets for experiment

We prepared our datasets in the form of CSV and ARFF file formats to represent in Weka. Besides, to make the datasets capable of determining optimal window size, the target word is identified and written at the middle of the sentences. So, 10-context word to the left and 10-context word to the right including part-of-speech tag information is attached to each context word or 40 attributes (20 to the left and 20 to the right) of the target word were prepared.

4. Results and Discussions

In this paper, a number of experiments were conducted. The first experiment was to determine baseline result. The result was similar as reported by a previous work [24]. In the second experiment, we used two experimental datasets: stemmed and unstemmed. In both datasets, part-of-speech tag information was added. According to the experimental results of the three machine learning paradigms namely: supervised, unsupervised, and semi-supervised, semi-supervised learning performed best. The supervised learning achieved performance improvement only on the unstemmed version next to semi-supervised learning. But, the unsupervised learning method indicated almost no improvement in our experiment for both experimental datasets. Table 3 shows the result using semi-supervised learning on stemmed and part-of-speech tagged corpus of window 9-9 and 1-1.

From previous WSD research works for Tigrigna, the optimal window size of ambiguous words using unsupervised learning was 2-2 [23] and 1-1 for semi-supervised learning [24]. The window 1-1 is similar to our baseline result using semi-supervised learning. However, while using the same experimental datasets and the same algorithms as in the baseline except that part-of-speech tag information was added in our experimental datasets, we found out that optimal window size of 9-9 or 11 is enough using the stemmed and part-of-speech tagged corpus. The reason for the increase in window size might be due to the increase in the number of attributes as part-of-speech tag information was attached to each word. Comparing the result with the baseline, we found performance improvement of 1.85% for ADTree,

8.19% for AdaboostM1, 9.99% for Bagging, and 11.76% for Naïve Bayes except SMO which performs less as compared to the baseline result. As it can be seen in Table 3, the highest performance achievement is using ADTree algorithm. The reason could be because ADtree has a reduced training cost and a very much reduced computation cost [16]. The second best performance is achieved using AdaboostM1. It is a general method for obtaining a highly accurate classification rule by combining many weak classifiers, each of which may be only moderately accurate [17]. In AdaboostM1, different weak learners or classifiers are combined together to improve the classification performance. On the other hand, bagging trains multiple instances of a classifier on different subsamples of the training data [27]. Using Bagging, we achieved comparatively lower performance out of the five selected algorithms; but it is better than a previous work [24] using semi-supervised learning. Similarly, SMO and Naïve Bayes perform almost comparable.

Table 3: Results Using Semi-supervised learning (9-9 or 1-1 window)

Ambiguous word	*Optimal Window size and Algorithms*				
	Bootstrapping (9-9)			*SVM (9-9)*	*Bayes (1-1)*
	ADTree	*AdaBoostM1*	*Bagging*	*SMO*	*Naïve Bayes*
Kefele	97.449%	96.9388%	88.7755%	90.8163%	91.8367%
OareQe	98.125%	98.75%	91.875%	93.125%	92.5%
SeOare	91.9753%	89.5062%	79.6296%	87.6543%	91.9753%
Halefe	95.5941%	88.9182%	88.9182%	88.6544%	85.974%
Medeb	94.4342%	96.7213%	96.7213%	93.4426%	88.5246%
Average	*95.5155%*	*94.1669%*	*89.1839%*	*90.7385%*	*90.162%*

5. Conclusions and Future Works

The study was aimed to investigate and improve the performance of WSD prototype model for Tigrigna using part-of-speech tagged corpus. To do so, a number of experiments were conducted. We achieved the highest result using semi-supervised learning. Hence, we concluded that semi-supervised learning is the best machine learning method for Tigrigna WSD while using part-of-speech tagged corpus compared to supervised and unsupervised learning. We determined optimal window size of 9-9 for the bootstrapping algorithms (ADTree, AdaboostM1, and Bagging) and SVM (SMO). Similarly, for Naïve Bayes, window size of 1-1 is enough. In general, we achieved an improved WSD model for Tigrigna. Therefore, we concluded that inclusion of part-of-speech tag information into each word of the corpus used in [21] improved performance of WSD for Tigrigna using semi-supervised learning. From this, it can also be inferred that integration of WSD that incorporates part-of-speech tag information can improve the performance of Tigrigna NLP applications. In this study, we achieved better results compared to previously conducted WSD researches for Tigrigna.

We will further investigate using semi-supervised learning by improving the corpora and adding set of features in addition to those used in this study.

References

[1] Maurice van Keulen and Mena B. Habib, “Handling uncertainty in information extraction”, PhD Thesis, University of Twente, Enschede, The Netherlands, 2011.

[2] Leslau, W., Documents Tigrigna, Paris: Libraire Cklincksieck, 1998.

[3] Gebrehiwot Assefa, "A Two Step Approach For Tigrigna Text Categorization," Addis Ababa, 2011.

[4] Wikipedia, the free encyclopedia, “Tigrigna language”, retrieved from https://en.wikipedia.org/wiki/, last accessed on February 12, 2018.

[5] Ravi Som Sinha, “Graph-Based Centrality Algorithms For unsupervised Word Sense

Disambiguation", Masters Thesis, University of North Texas, 2008.

[6] Weaver, W., "Machine Translation of Languages," MIT Press, Cambridge, MA, 1949.

[7] Solomon Mekonnen, "Word Sense Disambiguation to Amharic Text: a Machine Learning Approach". Master's Thesis, Addis Ababa University, 2010.

[8] D. Jurasky and J. H. Martin, "Speech and Language Understanding", Upper Saddle River: Prentice-Hall, Inc., 2000.

[9] Prity Bala, "Knowledge Based Approach for Word Sense Disambiguation using Hindi Wordnet", The International Journal of Engineering and Science (IJES), Vol. 5, 2013.

[10] Nancy, Ide and Jean, Veronis, "Word Sense Disambiguation", The State of the Art, Computational Linguistics, 1998.

[11] Yogita Rani and Harish Rohil, "A Study of Hierarchical Clustering Algorithm", International Journal of Information and Computation Technology, Vol. 3, No. 10, 2013, pp. 1115-1122.

[12] Alok Ranjan Pal and Diganta Saha, "Word Sense Disambiguation: A Survey," IJCTCM, Vol. 5, No. 3, July, 2015, pp.1-16.

[13] R, Navigli, "Word Sense Disambiguation: A Survey," in ACM Computing Surveys, Vol. 41, 2009.

[14] Getahun Wassie, Ramesh Babu P., Solomon Teferra, and Million Meshesha, "A Word Sense Disambiguation Model for Amharic Words using Semi-Supervised Learning Paradigm".

[15] Thanh Phong Pham, Hwee Tou Ng, and Wee Sun Lee, "Word Sense Disambiguation with Semi-Supervised Learning", Department of Computer Science, National University of Singapore, 2005.

[16] Xiaohua Zhou and Hyoil Han, "Survey of Word Sense Disambiguation Approaches", College of Information Science & Technology, Drexel University, Philadelphia, pp.2-7.

[17] Gerard Escudero Bakx, "Machine Learning Techniques for Word Sense Disambiguation",

[18] https://www.techopedia.com/definition/ 33202/bagging, Last accessed 17, 09, 2018.

[19] Thanh Nguyen, Andrei Doncescu, and Pierre Siegel, "Performance Comparison of ADTree and Naive Bayes Algorithms for Spam Filtering", International Journal of Mathematical and Computational Sciences, Vol. 10, No. 5, 2016, pp. 1-6.

[20] Okbeab Tewelde Yohannes, "Phonetic-Based Keyboard Mapping for Tigrigna Language," Nairobi, Kenya, 2014.

[21] Yitna Firdyiwek and Daniel Yaqob, "The System for Ethiopic Representation in ASCII," 1993-1997.

[22] Tesfaye Tewolde, "A modern grammar of Tigrigna", Dettivia G. Savonarola Roma., 2002.

[23] Meresa Mebrahtu, "Unsupervised Machine Learning Approach For Tigrigna Word Sense Disambiguation," Gondar, March, 2017.

[24] Teklay Muruts, "Word Sense Disambiguation For Tigrigna Language Using Semi-supervised Machine Learning Approach,'' Addis Ababa University, June, 2018.

[25] Biruk Retta, "Application of Part-of-Speech Tagged Corpus to improve the Performance of Word Sense Disambiguation": The case of Amharic, Addis Ababa University, June, 2015.

[26] Mulugeta Atsbaha, "Automatic Part-of-Speech Tagger for Tigrigna Language Using Hybrid Approach, Addis Abeba University, October, 2016.

[27] Pavan, Kumar, Mallapragada, "Some Contributions to Semi-Supervised Learning", Michigan State University, Computer Science, 2010.