JIMMA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

DEPARTMENT OF INFORMATION SCIENCE

CONTEXT BASED SPELLCHECKER FOR AFAN OROMO WRITING

BY:

TIRATE KUMERA

November, 2018

Jimma, Ethiopia

A Thesis Submitted to the Department of Information Science in Meeting the Partial Fulfillment for the Award of the Degree of Masters in Information Science (Information and Knowledge Management)

Advisors

Principal Advisor: Dr.Million Meshesha(PhD)

Co-Advisor: Workineh Tesema (MSc)

November, 2018

Jimma, Ethiopia

JIMMA UNIVERSITY

COLLEGE OF NATURAL SCIENCE

DEPARTMENT OF INFORMATION SCIENCE

CONTEXT BASED SPELL CHECKER FOR AFAN OROMO WRITING

By:

TIRATE KUMERA

As members of the board of examining of the MSc thesis open defense examination of the above title, we members of the board (listed below), read and evaluated the thesis and examined the candidate.

Name	**Title**	**Signature**	**Date**
____________________	Chairperson	________________	_________
Dr.Million Meshesha(PhD)	Principal Advisor	________________	__________
Workineh Tesema (MSc)	Co-Advisor	________________	_________
______________________	External Examiner	________________	_________
______________________	Internal Examiner	________________	_________

DECLARATION

I declare that this thesis is my original work and it has not been presented for a Degree in any other universities. All the material sources used in this work are duly acknowledged.

Tirate Kumera

November, 2018

This thesis has been submitted to the department for examination with our

Approval as university advisors:

Principal Advisor: Dr.Million Meshesha (PhD) …………………….

Co-Advisor: Workineh Tesema (MSc) …………………….

DEDICATION

This work is dedicated to my father Kumera Biyana, my mother Wayitu Bulcha, all my brothers and my sister with their family who were able to reap the fruit of their own.   I love you all.

ACKNOWLEDGEMENTS

First and foremost, I am extremely grateful to heavenly father who helped me to succeed in all my life long learns. Next, I would like to thank to my Advisor Dr. Million Meshesha(PhD) for his valuable assistance in providing his genuine, professional advice and encouragement goes even beyond the accomplishment of this study. And also I would like to acknowledge my co-advisor Workineh Tesema (MSc) for all his guidance at every step of the thesis work, for patiently listening to me even during uncountable hours, for advising me to do my study for imparting so much valuable knowledge and for all his encouragement and words of kindness. Finally, I would like to express my appreciation to Mettu University for sponsoring my Msc. study at the Department of Information Science, specialization in Information and Knowledge Management in Jimma University.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CBSCAO:	Context Based Spelling Checker for Afan Oromo
HMM:	Hidden Markov Model
IBM:	International Business Machine Corporation
IR:	Information Retrieval
MS:	Micro Soft
MT:	Machine Translation
NLP:	Natural language processing
OBN:	Oromia Broadcasting Network
OCR:	Optical Character Recognition
OCT:	Oromia culture truism
POS:	Part Of Speech
SOUNDEX:	SOUND index
SR:	Speech recognition
VOA:	Voice of America

ABSTRACT

Spellchecking is the process of detecting and providing spelling suggestions for incorrectly spelled words in a text. It is directly interposed with several applications like post handwritten text digital correction and user word correction in the retrieval process. This thesis describes the design architecture, implementation and testing of a model that have been developed to detect and correct both non-word and real word. The main focus of this study is to design Context based spell checker for Afan Oromo writing depends on the spelling error patterns of language based on the sequence of words in the input sentences contextually. The technique used for this spelling correction is unsupervised statistical approach. Unsupervised statistical approach helps to prepare manually tagged data sets to help under resource like Afan Oromo language from collected corpus. The Process of spelling correction is undertaken through the following major phases: error detection, candidate suggestion and ranking candidate suggestion. Error detection is based on the dictionary look up method and bigram analysis. The researcher collected the data from the different sources and prepare the dictionary and bigram model for error detection and correction. The non-word error candidate generation is based on calculating the similarity between the misspelled word and list of token in the dictionary, similarity is measured using the Levenshtein to the dictionary token and ranking accordingly and for real word error, bigram frequency was used to detect the error and bigram probability was computed for the correction of misspelled. To conduct experiment 14,896 and 3231 words were used to learn and test the model respectively. Experiment result shows that, the spell checker score recall of 93.7% and accuracy of 93.9% for both non-word and real word spelling errors. According to gated result the accuracy of the system is 93.9%, this shows that the model is optimistic in order to correct misspelling Afan Oromo words. We advise to improve and complete the quality of the designed model through mixed approach (rule based approach and N-gram).

Key words: Context based Spellchecker, Real-word Error, Non-word Error, N-gram, and Levenshtein.

CHAPTER ONE

INTRODUCTION

1.1 Background of the study

Natural language processing (NLP) is a computational linguistics field of computer science that deals with human-computer interaction where formalization is applied on the element of human language to be performed by a computer system (Achenkunju and Bhuma, 2014). NLP has the ability to design and build software for analyzing, understanding and generating spoken and written words in natural language like English, Amharic, Arabic, Chinese, Afan Oromo (Manning and Schutze, 2008). Moreover, NLP has capabilities to understand and process natural language, translate languages accurately in real time, or extract and summarize information from a variety of data sources, depending on the users words (Zue et al., 2000). This help NLP effectively communicate with human since it focuses on designing and implementing tools, techniques, frameworks of natural languages.

NLP began in the late 1940s, it was focused on machine translations; in 1958, NLP was linked to the information retrieval by the Washington International Conference of Scientific information (Hindle et al., 2012). Primary ideas for developing applications for detecting and correcting text errors started at that period of time. Until today NLP has a great interest because it plays an important role in the interaction between human and computers. It represents the intersection of linguistics and artificial intelligence, where machine can be programmed to manipulate natural language(Nadkarni et al., 2011).

NLP has many applications such as automatic summarization, Machine Translation (MT), Part-Of-Speech (POS) Tagging, Speech Recognition (SR), and Optical Character Recognition (OCR). Also, Spell checker is another application of natural language process and started early mid of 20th century by Lee Earnest at Stanford University, USA; but the first application was created by Ralph Gorin, who is Lee's student. He use a dictionary of 10,000 English words and design application of spell checker(Al-bakry, 2014).

Spell checking typically has the major steps to correct misspelled words (Xie et al., 2015). The first step is to detect misspelled words in the document or from the user inserted words. The most popular way to identify misspelled word is to check in a dictionary lookup or in the list of correct words. The dictionary look up techniques helps to detect non-word error, a word which does not exist in the list of the dictionary. In natural languages a large number of words are available; as a result, huge dictionary needed to have all words; therefore the task of looking every word in the dictionary consumes a long time to identify incorrect word from dictionary (Achenkunju and Bhuma, 2014).

Candidate generation to correct the misspelled words is another steps in spellchecker. For non-word errors candidate words come from the dictionary list by calculating similarity between the misspelled words with the dictionary words. While, for real word error candidates computed from the bigram model depending on the probabilistic information. Finally, generated candidates was ranked based on their similarity score for non-word, whereas bigram probabilities information used to generate real word candidates. According to their probability results, the candidates was listed and displayed for the user to select the appropriate word(Mishra, at el., 2013).

Context based spell checking for misspelled word is new approach (Dickinson, 2013). Nowadays many applications were developed contextually to correct the spelling error. For instance, Microsoft word 2007 provided context based spellchecker to solve both non-word and real-word error during typing on the Microsoft word interface(Flor and Futagi, 2012). Besides, Google provides Google Suggester to solve the misspelled words due to the context based error from the user query (Inkpen, 2009).

There are different algorithm used to design context based spellchecker (which recognize both non-word error and real-word error correction) (Bassil and Semaan, 2012). Among them selecting appropriate algorithm is depending on different factors; such as corpus size, performance issues and runtime are recognized during non-word and real word error detection and corrections in order to meet the objective of context based spellchecker (Daiber et al., 2013).

Context based error detection and correction is help to improves the performance of spell checker, since it focuses not only on non-word error, but also on the real word error (Cucerzan and Brill, 2002). As a result, this study used context based ideas to correct the misspelled words that result of intended words error and non-existence in the dictionary. Generally, the aim of this study is to design context based spell checker which is unsupervised statistically capture the contexts to fix misspelled words.

Contextual information for individual word was captured from collected corpus to train and taste the model, to capture the information different algorithm was used. The most know algorithm to identify misspelling are bigram with dictionary look up method to detect misspelled words from user words(Ahmed et al., 2009).

On another hand, to correct the misspelled words, Levenshtein edit distance has been used to calculate similarity between misspelt words with all words so as to identify the correct word by generating candidate list for non-word error(Nejja and Yousfi, 2015). Moreover, N-gram works with probabilistic and frequency information, where *N* is either a unigram, bigram, trigrams, etc. model. The value of n-gram probability used to suggest possible corrections for real world errors(Kim and Rush, 2016).

The model provide a context-sensitive spell checking method that able to detect and correct kinds of human-generated real word errors and non-word errors. As suggested by Bassil and Semaan (2012) the starting point for the context sensitive spell checker is a dictionary look up spelling error detection and correction application that detects and corrects non-word errors. The context-based spell checker for Afan Oromo combine dictionary look up method and bigram to detect and correct the both non-word and real word error. Bigram is preferable when the corpus size is small, since Afan Oromo didn't have standard collected corpus the researcher collected sample corpus to test the model. In fact, as former study shows bigram method score good performance than trigram for small size of corpus (Ahmed et al., 2009). As a result bigram model was selected to detect and correct real word   by focusing on the bigram probability and frequency they have in the collected corpus. As well as, Dictionary look up method used for non-word detection and correction, according to the value of listed corrected words.

As per the researchers knowledge the spell checker and corrector techniques and many NLP tools have been developed for English language to more degree of acceptance, efficiency and correctness than that of Afan Oromo language. Regarding Afan Oromo language, there are numerous numbers of researches being undergoing and done to improve the gap and alleviate the problem in different areas of NLP.

Spelling checker of various kinds (e.g. probabilistic, rule based) have been developed for different foreign languages text, which have relatively wider use nationally and/or internationally (for example; English, German, Chinese and Arabic) (Cucerzan and Brill, 2002). Now, this study focus to designing a model of interactive context-sensitive spell checking for Afan Oromo language using bigram probability information to provide a valid solution to the problem of real-word errors and provide solution for non-word by using Levenshtein edit distance.

Afan Oromo language is one of the Cushitic languages it is the working language of Oromia regional state and medium of instruction for primary and secondary schools of the region. Most of the official documents in governmental and private sectors in Oromia are written in Afan Oromo and most private sectors in Oromia write in Afan Oromo. Afan Oromo language users can write Afan Oromo texts in word processing applications, mobile applications, search engine and others application. However, the users cannot be sure whether the written texts are spelling correct or not. Therefore, this research contributes a lot in identifying spelling errors from written texts and handling the misspelling words by providing corrected words in order to clearly transfer information.

Today knowledge is shared through the writing system. As a result, writing system must clearly transmit the knowledge without misspelling between the readers and authors. In a language if there is poor spelling, information gap, loss of information and misunderstanding of information can happen; this is dodgy for the advancement of the language.

1.2 Statement of the Problem

Writing system has a great role to share information between the authors and reader’s world. Poor spelling (misspelled) can hinder the communication between authors and readers(Treiman and Kessler, 2005). As a result, authors did not transfer his/her knowledge clearly for the readers and also poor spelling also has influence on the development of languages. This problem is happened in Afan Oromo writing system, due to misspelling words Oromia government take a correction measurement over government institution, non-government and business center misspelled words written on the banner and posted publically.

Unfortunately, existing word processors do not come with built-in spell checkers for Afan Oromo. Even if it supports many languages, it does not provide spelling check for all languages over the word. Currently, the problem of Afan Oromo language has no spell checker that correct spelling error contextually in Afan Oromo writing system. Obviously, due to lack of spelling checker in Afan Oromo, a lot of misspelled words appear in a piece of Afan Oromo written texts. This problem was observed on printing press, regional government secretary offices, advertise notices, board notices. This limitation need to be solved in order to support the authors and readers for best communication and reliable information/knowledge sharing.

Up till now, in Afan Oromo writing system there is absence of research conducted to design a context based spelling checker for correcting misspelling in the writing system. However, rule-based Afan Oromo Grammar Checker was designed in order to form well organized arrangement of words in Afan Oromo sentence (Tesfaye, 2011). But, this cannot hinder the problem of misspelling in Afan Oromo, since grammar focus on the construction of the sentence, while this spelling checker focuses on the contexts to check spelling error at the words level. Therefore, the aim of this research is to design a prototype context based spellchecking in order to solve the problem of misspelling in the Afan Oromo writing system.

1.3 Research Questions

This study answers the following research questions to come up with the solution for the misspelling problems for the success of context based spell checker for Afan Oromo:

- How the context based spell checker was designed for Afan Oromo words writing?
- Which algorithm is best to design context based spelling checker?
- To what extent the context based spell checker improve the Afan Oromo misspelled words?

1.4 Objective of the Study

1.4.1 General Objective

The general objective of this study is to develop a prototype of Context based spell checker for Afan Oromo writing system so as to improve knowledge sharing through good communication between the readers and authors world.

1.4.2 Specific Objectives

To achieve the general objective, the study attempts to address the following specific objectives:

- To identify suitable methods for implementing spell checker;
- To collect and prepare Afan Oromo corpus for training and testing the proposed model;
- To design context based spell checker model for Afan Oromo writing;
- To evaluate the performance of prototype spell checker;

1.5 Scope and Limitation of the Study

This study was bounded to develop a context based spell checking model for Afan Oromo language based on statistical frequency of words by using bigram probability of word occurrence preceding to another one and dictionary look up method. The model focuses on the spelling errors as a result of a typographical errors which may happen due to deletion, insertion, substitution, or translation of character as well as, real word error. Obviously, there are two types of error in spell checking(Ahmed et al., 2009). The first one is non-word error, an error that happen when the word didn't found in the list of the dictionary.

On other hand, other type of error is real-word error, an error happens when the word is correctly spelled since it matches with the list of bigram dictionary but it appears in wrong position in the sentence. Generally, in this study both non-word and real-word error are considered. However, this study did not cover all Afan Oromo words for corpus preparation. As a result, the model didn’t provide the candidate for all Afan Oromo misspelled words. As well as, the model didn’t automatically correct the misspelled words even when one candidate list is provided, it need the user interaction to fill the candidate for the misspelled word. Also compounds and abbreviation didn’t covered by the model, the system accept compound word as different words. For instance, ‘Mana citaa’ which means thatched house, the model accepted as ‘mana’ and ‘citaa’.

1.6 Methodology

1.6.1 Research Design

This study follows experimental research. According to (Gersten et al., 2005), experimental research is an approach that uses empirical evidence. It is a way of gaining knowledge by means of direct and indirect participation in the experience. As a result, in this study the researcher was used experimental method for model building and prototype development and performance evaluation. Generally for prototype system development the following procedures were applied in the study. These include literature review, corpus collection and preparation, system design and development, and finally evaluation of the system performance.

1.6.2 Literature Review

Related works was reviewed to get a deeper understanding about Afan Oromo language spelling structure, preprocesses of corpus and user words as well as, spelling check development and fundamental concepts of related to this work. A review on different approaches of spell checking systems also made to identify the best approach.

1.6.3 Data Processing

Construction of the text corpus is very helpful for the detection and correction of misspelled words in the spell checking(Han and Baldwin, 2011). In this work, Afan Oromo text corpus is created manually to apply in Afan Oromo Spell Checker system. Because, in Afan Oromo there is no standard corpus developed, therefore we collected free text corpus from Oromia Broadcasting Network (OBN), Oromia Culture and Tourism Office, different website like Voice of America (VOA) Afan Oromo section. This corpus contains different contents of disciplines, such as cultural, social, political, sports, and economic in order to avoid data scarcity and to prepare rich dictionary as well as test the model. Dictionary prepared from the collected corpus was preprocessed by applying tokenization and normalization of the collected data. Stored words are saved in the forms of text and help us for cross check the user inserted words.

1.6.4 Development of System Procedure

The model footpaths unsupervised statistical approach. Since unsupervised approach allow manual preparation of tagged and annotated text which is ideal for under resourced languages like Afan Oromo. Current study use collected corpus to detect and correct both real-word and non-word error. N-gram statistical methods help to detect and correct the spelling errors contextually depending on the neighboring words. In n-gram model the probability of occurrence of a word in a sentence is approximated by its probability of occurrence within a neighbor words. Typically sequences of two or three words (bigrams or trigrams) are used, and their probabilities are estimated from a large corpus. These probabilities are combined to estimate the a priori probability of alternative acoustic interpretations of the utterance in order to select the most probable interpretation for real-word error.

Additionally, the model uses dictionary lookup method to detect non-word error form the given text and flagged error full word by highlighting and display candidate suggestion by computing distance measurements between misspelled and dictionary words. As well as, the model uses N-gram analysis method to detect and correct the real-word error from the given text and flagged error by highlighting.

To develop the context based spell checker for Afan Oromo Java NetBeans 8.2 version programing environment was used to implementing the algorithm and prototype of spell checker. Java is used as a programming language in this study since it is a general purpose programming language, run in any platform and simple to design interface.

1.6.5 Evaluation Procedure

The designed prototype is evaluated in order to measure the accuracy of the model. Evaluation metric was one measurement to test the accuracy of context spell checker. Recall and precision evaluation metrics help to measure that actual performance of spell checkers for both non-word and real word spelling errors(Momtazi, 2002). Afan Oromo Spell checker also used Recall and Precision evaluation metrics to measure the performance of the model.

1.7 Significance of the Study

Spelling writing is a major activities that everybody performs from day to day and become a part of everyday life for today generation. It is the best way to transfer information from one source to another. Misspelling words can hinder information transfer between the author world and reader world. Today, spelling check is inevitable part of the process such as text editing tools in various areas like word prediction, machine translation, information retrieval, word processor, search engines and mobile applications. As a result designing spellchecker for Afan Oromo has a vital value to fostering the language.

The results of this study produce experimental evidences to detect and correct misspelling from the user words. It provides a model that check and corrects the misspelled words and facilitates good communication between the author and readers. As a result, the users of the language are the first beneficiary to minimize information gap and seamlessly transfer their knowledge/information from one to another. Also such NLP related research fosters the advancement of the Afan Oromo language.

1.8 Thesis Organization

This thesis is organized into six chapters: the first Chapter briefly discusses about the introduction part of the study, background of the problem area, the general and specific objectives of the study, the research methodology, the scope and limitation, programming tools used to develop the prototype, evaluation procedure and significance. Chapter two gives overview of related research work .It discusses the concepts in spell checking and n-grams techniques and concepts that motivate the experiment in this research are discussed in adequate detail. And also n-gram approaches for context sensitive spell checking and a review of works on related spell checking is presented. In chapter three, the characteristics of the Afan Oromo writing system that are related to the research area are discussed. The modeling and text preprocessing procedures for spell checker including described in chapter four. Experimentation and evaluation of the model by domain area is discussed in the fifth chapter and finally chapter six gives final conclusions of the research done and forward recommendations for future studies.

CHAPTER TWO

LITERATURE REVIEW AND RELATED WORKS

This chapter deals with the state of the art relating to context based spell checking and correcting with its spelling error types in the written texts and techniques to identify the misspelled words and correcting mechanism.

2.1 History of Spellchecker

Spellchecker is an application program that identify the misspelled words from a given text (Ratnasari et al., 2017). It may be stand-alone, or integrated with the other application, such as a, search engine, email client, electronic dictionary, or word processor. Application of spellchecking is not new, it start 20th century, the first spell checker application was developed for mainframe computers in 1970s. It support group six languages for the IBM (International Business Machine Corporation) Company. The main function of this application is that it only shows the error instead of the correct word for replacing the misspelled one. Moreover, for personal computer spell checkers first introduced on 1981 from the IBM Company. By the mid-1980s many popular word processing applications (such as WordStar and WordPerfect) had integrated spell checkers(Sidoti et al., 2008).

The main tasks of a spellcheckers are preprocessor (tokenization), error detection and correction, and ranking the suggestions. Error detection step focus on to identify and display misspelled words from the text. While, error correction module provide candidate suggestion for misspelled words to represent error words with correct words and ranking candidate suggestion (Tijhuis, 2014).

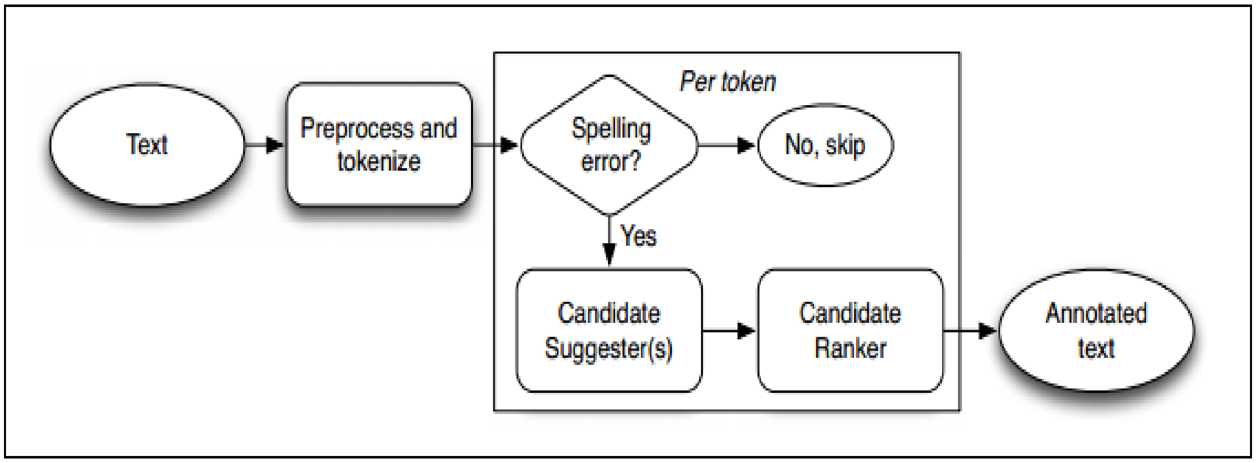


Figure 2. 1: Spelling check procedure (Source: Tijhuis, 2014).

2.2 Approach of Spellchecker

So far it has been discussed about spell checking and related concepts. This section focuses on brief descriptions of different approaches that used in spell checking. Various algorithms and methods were devised to tackle misspelling problem. There are several approaches that can be used in designing spell check model. Most common approaches are rule based approach and statistical approach (Ahmed et al., 2009).

2.2.1 Rule Based Approach

Rule-based approaches are based on a theory of morphology laid down by experts. These group of methods enable to incorporate sophisticated linguistic theory. These technique applies a set of rules on the misspelled word depending on common mistakes patterns to transform words in to valid one. After applying all the applicable rules, the set of generated words that are valid in the dictionary suggested as candidates and help to correct misspelled words (Liang, 2008).

As discussed by Jurafsky and Martin (2009), this algorithm uses two stages architecture. The first phase is concerned with getting all possible tags of each word of the sentence and the second phase is concerned with identification of the correct tag by using some hand written rules. It is the core component of the rule-based tagger. All words are given unique codes based on their context word categories.

The rules are called tag changing rules that provide information about appropriateness of a given words based on the context and transform erroneous words into correct words. These rules could specify a word following with others. Rule can be contextual rules, which are predefined rules based on context, or it can be lexical rules that help the tagger to make reasonable guess. Contextual rules modify the tag of the word based on the surrounding words whereas lexical uses morphological behavior of the word itself (Xie et al., 2015)

2.2.2 Statistical Based Approach (Corpus Based)

The statistical approach as its name implies extracts the statistical properties of words in the training phase to label words with their correct information includes frequency, probability or statistics (Koehn et al., 2007). Statistical approaches do not strictly follow explicit theory of linguistics like rule based. The approaches are completely based on test corpora, which constitute the input data. Approaches in this category use some algorithms to learn, say about the word formation process of a language from a given corpus and perform the analysis based on this knowledge. Moreover, the employed algorithms are subject to modification and further fine tuning during the operation(Markou and Singh, 2003).

Statistical approaches are further divided into supervised and unsupervised based on the type of test corpora they use. Supervised approach, requires annotated text corpora. In this case a trainer input is provided. On the other hand, unsupervised approaches use heuristics or probability information generated from the test corpora to generate the morphological analysis system (Hilas and Mastorocostas, 2008).

2.2.2.1 Supervised Statistical Approach

Supervised approach uses semantically annotated corpora to train machine learning (ML) algorithms to decide which word sense to choose in which contexts. The words in such annotated corpora are tagged manually using semantic classes taken from a particular lexical semantic resource (most commonly Word-Net). Supervised approached are known as Corpus-based methods when they learn from previously sense annotated data, and therefore they usually require a large amount of human intervention to annotate the training data (Navigli, 2009).

2.2.2.2 Unsupervised Statistical Approach

Unsupervised leaning approaches determine the class membership of each object to be classified in a sample without using sense tagged training examples. This involves clustering the contexts of an error word into a number of groups and discriminates among them without labeling them (Navigli, 2009). Misspelling as sense discrimination can be achieved through unsupervised clustering. As the result, unsupervised approach was implemented for current study.

2.3 Text Error

A word can be mistaken in two ways: the first is by incorrect spelling a word due to lack of enough information about the word spell or intentionally mistaking symbol(s) within the word, this type of error is known as non-word errors where the word cannot found in the language lexicon. Whereas, the second is using correctly spelled word in wrong position in the sentence or unsuitable context. These errors are known as real-word errors where the incorrect word is accepted in the language lexicon (Amber et al., 2008).

Non-word errors are easier to detect, unlike real word errors; the later needs more information about the language syntax and semantic nature. Accordingly, the correction techniques are divide into isolated words error detections that is concerned with non-word error; and context based error correction which deal with real word error(Hirst and Budanitsky, 2003)

According to Christen (2012) Spelling errors are divided into two different categories: Typographical errors and Cognitive errors. The first category, called typographical error occurred when the correct spelling of the word is known, but the word is mistyped by mistake. These errors are mostly related to the keyboard and therefore do not follow any linguistic criteria. Whereas, cognitive errors were produced when the correct spellings of the words are not known and lack of knowledge about correct spelling of the target language. In these types of errors, the pronunciation of the misspelled word is the intended as correct word. The user simply does not know how to spell or misconception on how to spell the word and writes the word in an erroneous form, for instance, when user writing “computer flies” instead of “computer files”. In the spelling error cognitive error happened due to pronunciation similarities between the erroneous word and correct word(Christen, 2012). For instance, the user write “ingenious” instead of “ingenuous” the sound they formed are similar.

2.3.1 Non-Word Errors

Non-words errors are spelling errors that not found in the list of words in the dictionary (Tavast et al., 2012). In non-word error, a word may incorrectly type because there is extra space, extra character, misspelled word, or other possibilities. These errors are easier to detect, because just comparing the words in a text with the entries in a dictionary will filter out the erroneous words. According to Liang (2008) 80% of misspelled words that are non-word errors are the result of a single insertion, deletion, substitution or transposition of letters:

- **Insertion.** Adding an extra letter, e.g., '*laekki'* instead of '*lakki'* which means no. An important special case is a repeated letter, e.g. '*deemmi'* instead of '*deemi'* relate with go in English context.
- **Deletion.** Missing a letter, e.g., *'tle' instead of 'tole'*. An important special case is missing a repeated letter, e.g., *'eyee instead of 'eyyee''*.
- **Substitution.** Substituting one letter for another, e.g., *'ejjenni' instead of 'ejjenno'*. The most common substitutions are incorrect vowels.
- **Transposition.** Swapping consecutive letters, e.g. '*jiaarchuu'* instead of '*jiraachuu'*.

Non-word spell checker relies on the prepared dictionary to detect the error as well as, to correct the misspelled words by providing the candidates(Ratnasari et al., 2017). According to (Misha and Navjot, 2013), non-word error detection is used by the dictionary lookup methods. Dictionary is list of the word prepared from the collected corpus and accepted as the corrected or free non-error words. To correct the misspelled words most models used Levenshtein distance to generate candidate by measuring the similarity between misspelled and corrected word(Ratnasari et al., 2017).

2.3.2 Real Word Error

These errors occur through mistaking an intended word by another on that is dictionary accepted. It is correctly spelled word in wrong position in the sentence or unsuitable context. This error is error where the incorrect word is accepted in the language lexicon and can be resulted from Cognitive errors (Amber et al., 2008).

Correcting real-word error is the most challenging task for a spell checking system. However different scholars try to tackle the problem of real word error. Brill and Moore (2000) suggested the use of a noisy channel to predict the actual correction for a real-word error. They try to prepare around 100 million words corpuses and use n-gram statistics to correct real-word error. (Sundby, 2009) proposed an approach using the *n*-gram model to predict the actual correction for a real-word error.

The idea centers on generating candidate spellings for every misspelled word by applying simple edit operations such as insertion, deletion, and substitution, and then using *n*-gram statistics derived from a corpus by computing the probability of word. The spell checker of the recently released Microsoft Word 2007 is able to detect and correct some real-word mistakes (Jurafsky, and Martin, 2009). Concerning the correction techniques, context sensitive error correction deals with real-word error and mostly solved by using the n-gram analysis. Generally, correcting real word errors is context based in that it needs to check the surrounding words and sentences before suggesting candidates.

2.4 Spelling Error Detection Techniques

Indicating whether a word is correct or not is based on the type of correction procedure; non word detection is usually checking the acceptance of a word in the language dictionary (lexicon) and marks any mismatch words as incorrect. Whereas, real word error is more complex task, it requires analysis large parts from the text to provide candidate for misspelled (Sundby, 2009).

According to Ratnasari et al., (2017) there are two main techniques to detect the non-word error detection: character n-gram analysis and Dictionary lookup. Character n-gram techniques work by examining each character n-gram in an input string and looking it up in a precompiled table of n-gram frequencies. Strings that didn’t found in the precompiled are accepted as misspelled words. N-gram analysis techniques require a large lexicon or corpus in order to compile the table of n-gram frequencies because of this limitation dictionary lookup techniques are more used than character n-gram techniques for detecting human generated errors (Cohen, 2001).

2.4.1 Dictionary Look Up Method

Dictionary lookup is a method straightforward methods, directly it checks the presence of every input text in the dictionary. However, the response time becomes a problem when dictionary size exceeds a few hundred words. In document processing and information retrieval, the number of dictionary entries can range from 25,000 to more than 250,000 words. However, to gaining fast access to a dictionary hash table techniques preferable(Pagh and Friche, 2004).To look up an input string, one simply computes its hash address and retrieves the word stored at the address in the structure hash table. Additionally, dictionary-lookup techniques are more accurate than character n-gram techniques (Han et al., 2012) to correct the non-word error. Therefore, most techniques for detecting and correcting human-generated errors are dictionary based.

Dictionary lookup methods performed through hashing, binary search trees and finite state automata. Among the techniques of dictionary lookup Hashing is the most significant and efficient lookup strategy to minimize the time computational relies on input string to detect where a matching pattern found (Liang, 2008).

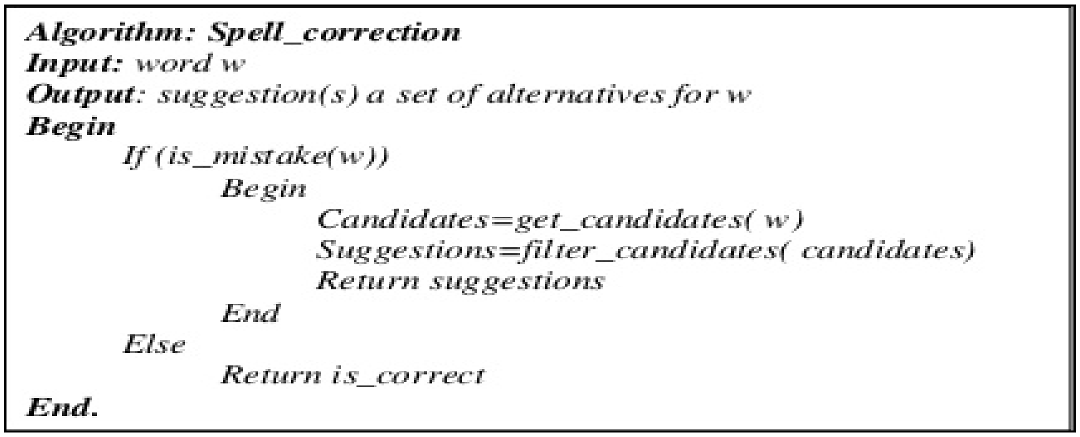
2.4.2 N-Gram Analysis

N-gram is defined to be n subsequences of words or string where n is variable, often takes values: one to produce unigram, two to produce bigram, three to produce trigrams or rarely takes large values (Sundby, 2009). This techniques detects errors by examining each n-gram from the given string and looking it with a precompiled n-gram statistics table. The decision depends on the existence of such n-gram or the frequency of it occurrence, if the n-gram is not then the words or strings are misspelled (Misha and Navjot, 2013). In this work bigrams were chosen and extracted from the input sentences at word level rather than character level during error detection. The existence of bigram word accepted as the corrected words, whereas the nonexistence of bigram words is accepted as misspelled words.

2.5 Spelling Error Correction Techniques

Error correction is the method which the spell checker find out the candidate list for the misspelled words. Numerous approaches are proposed to correct spelling errors, such as Levenshtein edit distance, noisy channel, Similarity Keys, Probabilistic techniques(Ahmed et al., 2009).

Boswell(2005), describes algorithm for spelling correction as shown in figure 2.2.



```
Algorithm: Spell_correction
Input: word w
Output: suggestion(s) a set of alternatives for w
Begin
        If (is_mistake(w))
                Begin
                        Candidates=get_candidates( w )
                        Suggestions=filter_candidates( candidates)
                        Return suggestions
                End
        Else
                Return is_correct
End.
```

Figure 2. 2: Outline of spell correction Algorithm.

2.5.1 The Noisy Channel Model

The concept behind the noisy channel model is to consider a spelling error as a noisy signal that has been distorted somehow during transmission. The essence of this approach is that if one could know how the original word was distorted, it is then easy to find the actual correction (Jurafsky and Martin, 2009). The noisy channel model is a special case of Bayesian inference (Wilde, 2017) which is principally a classification type model that inspects some observations and ranks them into appropriate classes and categories.

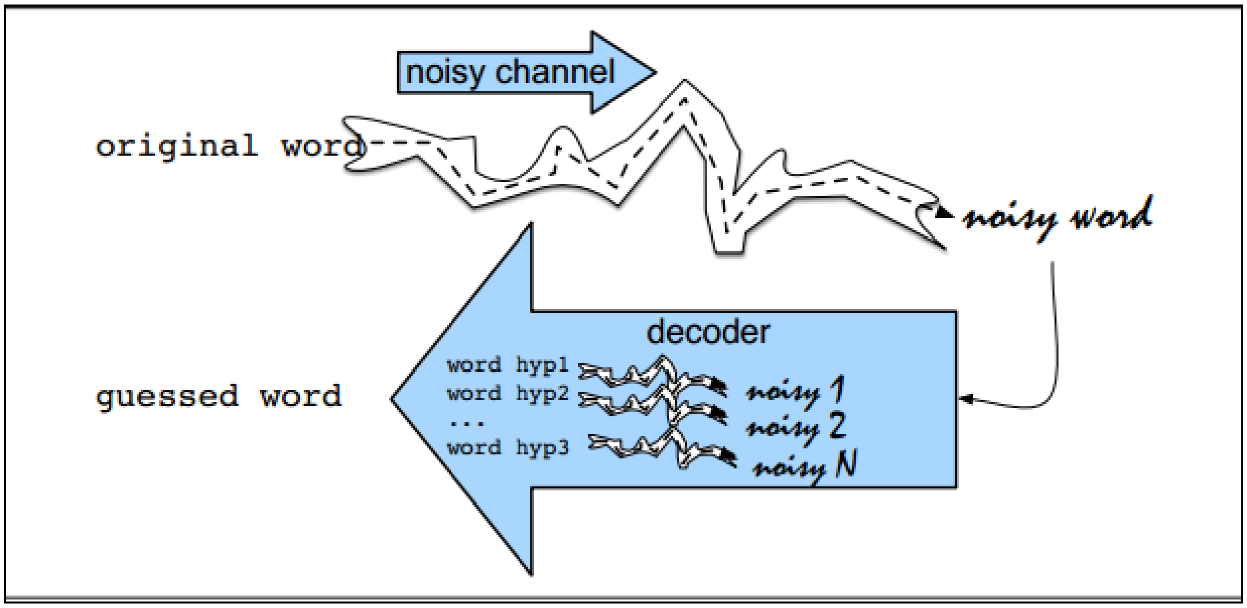


Figure 2. 3: Noisy channel structure (Source: Wilde, 2017).

Mathematically, the Bayesian model is a probabilistic model based on statistical assumptions and probability theory, namely the prior probability P (w) and the likelihood probability P(O|w)which are usually calculated by the following equation(Bassil and Alwani, 2012):

$$W= \underset{W\varepsilon V}{\operatorname{argmax}} \frac{P(O\backslash W)P(W)}{P(O)} = \underset{W\varepsilon V}{\operatorname{argmax}} P(O\backslash W)P(W) \qquad (2.1)$$

P (w) is called the prior probability and indicates the probability of w to occur in a specific corpus. P (O|w) is called the likelihood probability and denotes the probability of observing a misspelling O given that the correct word is w. O is the actual misspelled word and w is a potential spelling candidate. For every candidate, the product of P (O|w)*P (w) is to be calculated; the candidate having the greatest product is to be selected as a correction for O and is denoted by w'. The prior probability P(w)is straightforward as it is simply computed as P(w)= C(w)+ 0.5 / N+ 0.5, where C(w) is the frequency or the number of occurrence of the word w in the corpus, and Nis the total number of words in the corpus. In order to avoid zero counts for C (w), the value of 0.5 is added to the equation. On the other hand, the likelihood P (O|w) is harder to calculate than P (w) as it is imprecise to find the probability of a word to be misspelled, however, it can be estimated by calculating the probability of possible wrongful insertion, deletion, substitution, and transposition in general.

2.5.2 Minimum Edit Distance Techniques

This technique stand on counting the minimum number of primal operation required to convert the source string into the target one. The minimum edit distance has been defined as the minimum number of editing operations (i.e. insertions, deletions and substitutions) that is required for transforming one string into another (Misha and Navjot , 2013). For the first time, minimum edit distance spelling correction algorithm idea was implemented by Damerau (Deorowicz and Ciura, 2005) based on the insertions, deletions and substitutions character transformation. The major advantage of using a minimum edit distance measure is the fact that ranking can be performed easily. Minimum edit distance has different algorithms like Levenshtein algorithm, Hamming, Longest Common Subsequence (Mishra and Kaur, 2013).

2.5.2.1 Levenshtein Edit Distance

Levenshtein distance is a string matrix for measuring the difference between two sequences. It is minimum number of single character edits required to change one word into the other (insertion, deletion, substitution) each costs 1 edit. It is used in Spell checking, Speech recognition (SR), DNA analysis, and Plagiarism detection.

In spellchecking it calculates the similarity between misspelled and every words in the dictionary and take the lowest score as the best match by counting one distance from every primal operation, Hamming Algorithm works like Levenshtein but limited with only string of equal length and Longest common substring find the mutual substring between two words. Levenshtein algorithm is preferred because it has no limitation on the type of symbols, or on their lengths (Haldar and Debajyoti, 2011). Mathematically it is defined as follows (Su, 2008):

$$f(i, j)=\min[(f(i-1,j)+1,f(I,j-1)+1, f(i-1, j-1)+d(n_i\, m_j))] \qquad (2.1)$$

where $d(n_{i,}\, m_i)=0$ if $q_i= l_i$

$d(n_{i,}\, m_i)=1$ else return Null.

A function f(i, j) is calculated for all error word letters and all dictionary words, iteratively counting the string difference between the misspelled word n1, n2, …, ni and the dictionary word m1, m2,…, m j and each insertion, deletion, or substitution is gated score of 1.

2.5.3 Similarity Keys

Another method to correct the detected error is similarity key measurements techniques. This technique finds a unique key to assembly similarity spelled word together. The similarity key is computed for the misspelled word and mapped to a pointer refers to the group of words that are similar in the corrected words. Soundex algorithm helps us to find the similarity key to correct the misspelled word. It finds the similarity key depending on the pronunciation of the words. And also SPEEDCOP system finds the similarity key by rearranging of the letter of the words (Misha and Navjot, 2013).

2.5.4 Probabilistic Techniques

Probabilistic techniques based on the statistical features of the language. This employed by two common methods are transition probabilities and confusion probabilities (Ahmad and Kondrak, 2005). Transition a probability is depends on the probabilities of a given letter to be followed by another one. The probability is estimated according to n-gram statistics from corpus. Confusion probability is the probability of a given letter to be confused or mistaken by another one. This type of probability is depending on the source.

- **Transition or Markov probabilities**: it determines the likelihood that a certain given letter will be followed by another given letter in a given language. These probabilities can be determined by collecting n-gram frequency information from a large corpus (Baik et al., 2006).
- **Confusion or error probabilities**: determine the probabilities that a certain letter substitutes another given letter. It can be determined by feeding a sample text into the Optical Charter Recognition (OCR) device and tabulating error statistics (Evermann and Woodland, 2000).

2.5.4.1 Hidden Markova model

Hidden Markov Model (HMM) is one of the most commonly used probabilistic models which depend on Markov Models. Markov models are state-space models that can be used to model a sequence of random variables that are not necessarily independent. A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules (Blunsom, 2014). The defining characteristic of a Markov chain not focus on how the process arrived at its present state, the possible future states are fixed. In other words, the probability of transitioning to any particular state is dependent solely on the current state and time elapsed. The state space, or set of all possible states, can be anything: letters, numbers, weather conditions, baseball scores, or stock performances. The correct implementation of Markov models requires reliable and robust estimation of transition probability matrices (TPMs).

In general, Hidden Markov Model is defined by specifying the set of {S, V, A, B, π}, which are:

S = the set of states = {S1, S2... Sn}

- V = the output alphabet = {v1, v2... Vm}
- π(i) = probability of being in state qi at time t = 0 (i.e., in initial states)
- A = transition probabilities = {a_{ij}} where a_{ij} = Pr[entering state qj at time t+ 1 j in state qi at time t]: Note that the probability of going from state i to state j does not depend on the previous states at earlier times; this is the Markov property.
- B = output probabilities =$b_{i(k)}$ where b= Pr[producing vk at time t j in state qj at time t].

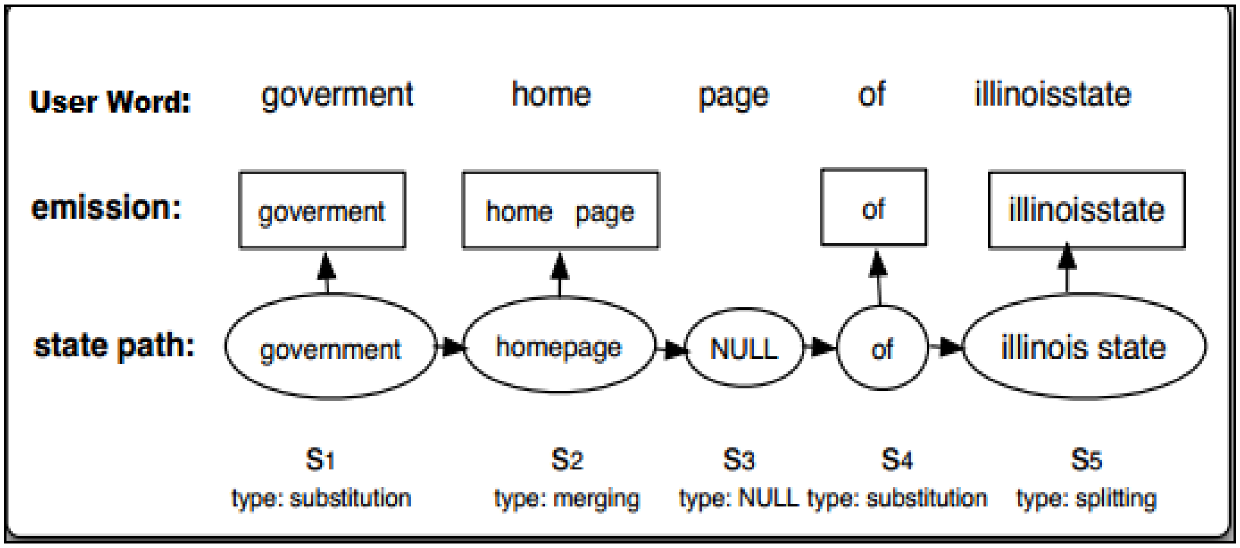


Figure 2. 4: Markova Model steps (Source: Li, 2012).

This sequence of states is also called Markov Chain. Suppose that X = (X1... XT) is a sequence of random variables taking values in some finite set S = {S1… SN} the state spaces, then the Markov properties are:

- **Limited Horizon:** -That is, future elements of the sequence are conditionally independent of past elements, given the present element.

$$P(X_{t+1}= S_k/X_{1\ldots}X_t) = P (X_{t+1}= S_k/X_{1\ldots}X_t) \qquad (2.2)$$

- **Time invariant (stationary):-** The above (limited horizon) dependency doesn't change over time.

$$P(X_{t+1}= S_k/X_{1\ldots}X_t) = P (X_2/X_1) \qquad (2.3)$$

Where, X is represent Markova Chain.

2.6 Context based Spelling Checking Techniques

Context based spelling error correction is the task of detecting and correcting errors that result in user type a correct spelled word when another intended(Nejja and Yousfi, 2015). Because of the existence of real-word errors, we need to use a different approach in order to identify errors. When using the context for error detection, we do not only look at each individual word, but take the words surrounding it into account when deciding if some word is incorrect or not. We now take the two words, and try to produce similar words. For instance, in the sentence "you should constantly backup your computer flies", the word "flies" is a real-word error mostly caused by a typographical mistake. Obviously, the writer did not intend to mean that computer flies like planes, but he most probably meant "computer files". This slight confusion produced a real-word error that is actually valid in the English dictionary, however invalid with respect to the sentence in which it has occurred.

Context-sensitive spelling error correction tries to detect and correct such real-word errors by inspecting their structural contexts. For those words we calculate a probability that indicates how likely the words is occur with each other. The words with the highest probability of occurring is then selected as the proper word and can be suggested to the user. According to Inkpen(2009), three methodologies that can be used to work on context-sensitive spelling detection and correction, they include: method based on semantic information, method based on machine learning and method based on probability information (Inkpen, 2009). The method used in this study was probability information.

2.6.1 Methods Based on Semantic Information

The semantic information approach was first proposed by Hirst and St-Onge in 1998(Budanitsky and Hirst, 2006). Later developed by Hirst and Budanitsky (Hirst and Budanitsky, 2003). This approach was based on the observation that the words that a writer intends to use are semantically related to their surrounding words whereas some types of real-word errors are not, such as the example given in Hirst and Budanitsky's paper, "It is my sincere (hope) that you will recover swiftly." Such errors will result in a worry of the consistency and coherence of the text. Hirst and Budanitsky use semantic distance measures in WordNet to detect words that are potentially anomalous in context that is, semantically distant from nearby words; if a variation in spelling results in a word that is semantically closer to the context, it is hypothesized that the original word is an error and the closer word is its correction(Inkpen, 2009).

2.6.2 Methods Based on Machine Learning

The machine learning method is regarded as a lexical disambiguation task and confusion set are used to model the ambiguity between words. The machine learning and statistical approaches are often based on pre-defined confusion sets which are sets of commonly confounded words, such as (their, there) and (principle, principal). These methods learn the characteristics of a typical context for each member of the set and detect situations in which one member occurs in context that is more typical of another. Such methods are limited to a set of common and predefined errors, but such errors can include both content and meaning words. Given an occurrence of one of its confusion set members, the spellchecker's job is to predict which member of that confusion set is the most appropriate in the context(Inkpen, 2009).

2.6.3 Method Based on Probability Information

Moreover, according to Inkpen(2009), Mays proposed a statistical method using trigram (a sequence of three words) probabilities for detecting and correcting real-word errors without the need of requiring predefined confusion sets. In this method, if the trigram deduced that the probability of an observed sentence is lower than the sentence obtained by replacing one of the words with a spelling variation, then the original is a real-word error and the variation is what the user intended to use(Inkpen, 2009).

For current study, the probability information approach has been chosen, because as research has shown the accuracy of this method approach is relatively high(Inkpen, 2009). As mentioned in the previous chapter, this study will focus Context-sensitive spell checking using bigrams probability. N-grams probabilistic was the appropriate algorithm to detect and correct real word error in spelling checker application (Bassil, 2012). This technique detects errors by examining each n-gram from the give string looking it with a pre compiled n-gram statics table. N-gram techniques usually require either dictionary look up techniques or a large corpus of text in order to pre-compile an n-gram table. The major advantage of n-grams techniques are language independent which means didn't require knowing language knowledge to develop the spellchecking application (Neha, 2006). Additionally, n-gram help by providing probability information that estimate a given letter followed by another one to find a valid solution for real-word errors (Deksne and Skadins, 2011).

Knowledge about words in a context are stored in a language model. The probabilities are trained by taking large amounts of texts and then count how often every n-gram occurs. Given the number of token and the value of N, the maximum number of sequences for the tokens can be calculated by the formula (Jurafsky and Martin, 2009):

$$\text{Ns} = \text{Nt} - (\text{N} - 1) \tag{2.4}$$

Where, Ns is the number of sequences, Nt is the number of tokens and N is the N value in the

N-gram.

Suppose Ns is list of sequences for a given sentence. One of the sequences in Ns has W1 and

W2 in it (this means the value of N is 2 this time). The probability of the sequence i.e. the Probability of W2 given W1 is calculated as follows (Jurafsky and Martin, 2009):

$$\text{P (W2/W1)} = \frac{Count(W1W2)}{Count(W1)} \tag{2.5}$$

Where, W2 is the second word and W1 is the first word in the sequence.

The probability of sequences for N = 3, trigram, can be calculated by counting the number of the three words occurring together over the number of the first and second words together. If W1, W2 and W3 are the first, second and third word in a given sequence, the probability of the sequence (which is the probability of W3 given W1 and W2 together) can be calculated as follows(Grinstead and Snel, 2012):

$$P\ (W3/W1W2) = \frac{Count(W1W2W3)}{Count(W1W2)} \qquad (2.6)$$

The generalized formula of the probability of the sequence for N = n is given by:

$$P\ (Wn/W1W2 \ldots Wn-1) = \frac{Count(W1W2 \ldots Wn-1\ Wn)}{Count(W1W2 \ldots Wn-1)} \qquad (2.7)$$

2.7 Performance of spellchecker Evaluation method.

The designed prototype must be evaluated in order to measure the effectiveness of the model. Evaluation metric help us to measure that actual performance of spell checkers for both non-word and real word spelling errors (Borah, 2017). Evaluation of the prototype system is made with the evaluation parameter that matches the number of user-words, which are categorized correct and misspelled word. We use the metrics such as precision, recall and accuracyfor effectiveness measures to evaluate context based spell checking for Afan Oromo Writing.

2.7.1 Recall

Recall is a measure of the completeness of the spell checker (Borah, 2017). It focuses on measuring to what extent the model identifies valid word from the corpus.

Lexical Recall (LR): it is the number of valid words in the text that are recognized by the spelling checker (true positive (TP)), in relation to the total number of correct words in the text (true positive and false negative (FP) (Olson and Delen, 2008), (Borah, 2017).

$$\textbf{Lexical Recall (LR)} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \qquad (\mathbf{2.8})$$

Error Recall (ER)*:* which is the number of invalid words in the text that are flagged by the spelling checker (true negative (TN)), in relation to the total number of incorrect words in the text (true negative and false positive (FP)) (Olson and Delen, 2008). .

$$\textbf{Error Recall (ER)} = \frac{\boldsymbol{TN}}{\boldsymbol{TN+FP}} \qquad (\mathbf{2.9})$$

2.7.2 Precision

Precision is a measure of the correctness of the spell checker’s responses(Olson and Delen, 2008), (Borah, 2017).

Lexical Precision (LP): is computed by dividing all correct non-flags (true Negative) by the total number of non-flags (i.e. true positives plus false positives)

$$\textbf{Lexical Precision (LP)} = \frac{\boldsymbol{TN}}{\boldsymbol{TN+FP}} \qquad (\mathbf{2.10})$$

Error Precision (EP): is the number of correct flags (true negatives) in relation to the total number of flags assigned by the spelling checker (i.e. true negatives plus false negatives) gives an indication of the spelling checkers(Olson and Delen, 2008).

$$\textbf{Error Precision (EP)} = \frac{\boldsymbol{TN}}{\boldsymbol{TN+FN}} \qquad (\mathbf{2.11})$$

F-measure is one of a performance measure that combines Recall and precision into a single measure of performance, this is just by taking into account the product of Precision and Recall divided by their sum. Therefore, F-measure is expressed by formula as follows (Borah, 2017).

$$\boldsymbol{F\ measure} = \frac{\mathbf{2} * \boldsymbol{precision} * \boldsymbol{recall}}{\boldsymbol{precision} + \boldsymbol{recall}} \qquad (\mathbf{2.12})$$

2.7.3 Accuracy

Accuracy calculated from both precision and recall which measures the general quality of the spell checker. Predictive Accuracy is the overall performance of the spell checker that have been computed and the likelihood of any given word correct or incorrect being handled accurately by the spelling checker. This metric measure the performance of spell checker for both spelling error detection and correction of a given words in the input texts(Borah, 2017).

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TN+TP+FN+FP}} \qquad \textbf{(2.13)}$$

2.8 Related Works

There are different works reviewed from various journals and web pages. Some of the spell checking methodologies, techniques, and what the scholars are developed and their achievement are reviewed as follows.

According to Liang (2008), techniques of spelling check are dividing into two: Dictionary Look up and N-gram analysis (Liang, 2008). Dictionary Look Up method is most popular method to identify the misspelled words, as a result most spell checking use dictionary look up method to identify or to detect the error from the text (Mihov and Svetla, 2004). This dictionary is assumed to contain a complete list of all correctly spelled words in the language. In dictionary look up method first the word inserted tokenized and the token is searched in the list of stored words.

Chakraborty (2010) showed that the amount of words in a dictionary determines the effectiveness of finding misspellings in a text. Having a small dictionary will lead to a lot of correct words being marked as incorrect. Using a larger dictionary will solve this, but increases the time it takes to look up words. According to gated result the enlarged access times can be solved by exploiting hash tables, tries, and other optimized storage mechanisms. Nowadays, with the increased computational power available, the time needed to look up words in a dictionary can be neglected.

Raaijmakers (2013) proposed a model for spelling correction based on treating words as "documents" and spells correction as a form of document retrieval in that the model retrieves the best matching correct spell for a given input. The words are transformed into tiny documents of bits and hamming distance is used to predict the closets string of bits from a dictionary holding the correctly spelled words as string of bits. The model is knowledge free and only contains a list of correct word.

Bassil and Semaan (2012) produced a parallel spell checking algorithm for spelling errors detection and correction. The algorithm is based on information from Yahoo! N-gram dataset 2.0; it is a shared memory model allowing concurrency among threads from both parallel multi-processor and multi-core machines. The three major components (error detection, candidate generation and error correction) are designed to run in parallel way. Error detector based on unigrams; detects non-word errors; candidate generator is based on bi-grams; the error corrector; based on the context sensitive, is based on 5-gram information.

Seo et al. (2012) presented a novel method for grammatical error correction by building a meta-classifier. The meta-classifier decides the final output depending on the internal results from several base classifiers; they used multiple grammatical errors tagged corpora with different properties in various aspects. The method focused on the articles and the correction arises only when a mismatching occur with the observed articles.

Kirthi et al. (2011) proposed a semantic information retrieval system performing automatic spell correction for user queries before applying the retrieval process. The correcting procedure depends on matching the misspelled word a correctly spelled words dictionary using Levenshtein algorithm. If an incorrect word is encountered then the system retrieves the most similar word depending on the Levenshtein measure and the occurrence frequency of the misspelled word.

Ahmed et al. (2009) developed a language-independent spell checker. It is based on the enhancement of N-gram model through creating a ranked list of correction candidates derived based on N-gram statistics and lexical resources then selecting the most promising candidates as correction suggestions. Their algorithm assigns weights to the possible suggestions to detect non-word errors. They depend on "multiWordNet" dictionary of about 80,000 entries.

Amber et al. (2008) designed a noisy-channel model of real-words spelling error correction. They assumed that the observed sentence is a signal passed through a noisy channel, where the channel reflects the typist and the distortion reflects errors committed by the typist. The probability of the sentence correctness, given by the channel (typist), is a parameter associated with that sentence. The probability of every word in the sentence to be the intended one is equivalent to the sentence correctness probability and the word is associated with a set of spell variants words excluding the word itself. Correction can be applied to one word in the sentence by replacing the incorrect one by another from the candidates (its real-word spelling variations) set so that it gives the maximum probability.

Mihov et al. (2005) described an approach for lexical-post correction of the output of optical character recognizer OCR as a two research project. They worked on multiple side; on the dictionary side; they enriched their large sizes dictionaries with specialist dictionaries; on the candidates selection, they used a very fast algorithm depends on Levenshtein automata for efficient selecting correction candidates with a bound not exceeding .They ranked candidates depending on a number of features such as frequency and edit distance.

Verberne, (2002) described a context sensitive spell checking algorithm based on the BESL spell checker lexicons and word trigrams for detecting and correcting real-word errors using probability information. The algorithm split up the input text into trigrams and every trigram is looked up in a precompiled database which contains a list of trigrams and their occurrence number in the trigram database, otherwise it is considered an erroneous trigram containing a real-word error. The correction algorithm uses BESL spell checker to find candidates but the most frequent in the trigrams database are suggested to the user.

Ringlstetter et al., (2007) use bigram model to correct misspelled words. The corpus for training and testing model derived from Web Crawled corpora to construct dictionaries and language models, which are used to correct the input text. For dictionary creation and bigram model 4,000 documents are used. Correction accuracy is improved when integrating word bigram frequency values from the crawls as a new score into a baseline correction strategy based on word similarity and word (unigram) frequencies. To guarantee efficiency, bigram counts for arbitrary bigrams over large dictionaries are kept in main memory, using special techniques. It was shown that bigram counts improve simpler scores based on word similarity and word frequency.

Yitayal (2016) attempted to design a context based spell checker for Amharic. Spell checking is based on the collected corpus from different source and the collected corpus store in the form of the dictionary. The non-word error corrected from the dictionary words by using the minimum edit distance. As well as, the real word error bigram profanities extracted from the collected corpus and evaluate each user words. Suggested candidates are ranked according to their score based on minimum distances.

Generally, the aim of this study was to design context based spellchecker for Afan Oromo writing system. In Afan Oromo writing system, there is no spellchecker used to correct the misspelled words. Obviously, misspelled word obstruct the idea from author’s world and reader’s world. To transfer the knowledge correct spelling has great role in order to facilitate knowledge sharing. In Afan Oromo the structure of the language and the letter arrangement in Afan Oromo is different from the other languages. Context based spell checker for Afan Oromo specifically needed to correct the misspelled words in the writing system.

CHAPTER THREE

AFAN OROMO WRITING SYSTEM

3.1 Overview

The Oromo people establish the single largest ethnic group in Ethiopia. Oromo People are the largest single ethno-nation in Eastern Africa, constituting at least 40% of the Ethiopian population (Girma, 2014).The Oromo people speak Afan Oromo (the language of Oromo), which belongs to the Eastern Cushitic family of Afro-Asiatic phylum. Outside Ethiopia, the language is spoken by thousands of other Oromo tribes in Kenya (Tesema et al., 2016). Besides being the widely used language in Africa, Afan Oromo has been included among the essential languages in the world. Justifying this, the report by the U.S Government and its Education Department (1985) has revealed that Afan Oromo has been considered as one of the 169 critical languages of the world. Currently, Afan Oromo is the working language of Oromia regional state (which is the largest one state among the current Federal states in Ethiopia). Being the official language, it has been used as a medium of instruction for primary and secondary schools of the region (Tesema and Tamirat, 2017)

3.2 Description of Afan Oromo Alphabets and Sound Systems

Different scholars identified that the writing system Afan Oromo relies on is Latin Script; the alphabets and sounds of the language are modifications of Latin writing system. Thus, Afan Oromo shares a lot of features with English writing system by the some modifications, and the writing alphabet of the language is known as '*Qubee Afaan Oromoo'* which is designed based on the Latin script. Thus, letters in English or Latin Alphabets are also found in Afan Oromo except the ways they are combined in phonetic alphabets and the styles in which they are uttered (Tune et at., 2008).

Since Afan Oromo writing system is a modification to Latin writing system, it shares a lot of features of English writing with some modification. Thus, letters in English language is also in Oromo, however, the language structure is completely different. In Afan Oromo the construction of sentences is Subject-Object-Verb. But in English Subject-Verb-Object is the arrangement of the sentences. For instance, "*Inni Amerika dhaa dhufe*" which means "He come from America", *Inni*, represent subject, *Amerika,* represent object and *dhufee,* represent verb.

As it has been mentioned before, Afan Oromo uses Latin character consonants and vowels (see table 3.1). It has 28 letters. The basic alphabet in Afan Oromo does not contain "p‟, ‟v‟ and "z‟. This is because there are no native words in Afan Oromo that are formed from these characters. However, later on a new letters was included in the alphabet as there are words which require the letter.

3.3 Vowels ('*Dubbachiiftuu*')

There are five vowels in Afan Oromo; these are 'a', 'e', 'o', 'u' and 'i'. They are similar to that of English, but they are uttered differently. Each vowel is pronounced in a similar way throughout its usage in every Afan Oromo literature(Tesema and Tamirat, 2017). In Afan Oromo words are constructed from the vowels and consonant. Vowels are sound makers and are sound by themselves. Vowels in Afan Oromo are characterized as short and long vowels ('aa','ee','oo','uu' and 'ii') to change the meaning of words. For example, while "*fayyaa*" means health, "*faaya*" is beauty. In Afan Oromo Vowel shortest and longest can make spelling error. Additionally, consonants characterized as double and single. Spelling error can occur by the adding or doubling and make single consonant. For example, "*gubbaa*" is represented as over and "*gubaa*" is represented as hot in English context. Hence misspelled word in Afan Oromo formed because of the shorten and longest of vowels, doubling and making single of consonant and interchange the position of character, removal of character, insertion of additional character.

Table 3. 1: List of Afan Oromo letters

Number	Capital	Small/Long	Type
1	A	a, aa	Vowel
2	B	B	Consonant
3	C	C	Consonant
4	D	D	Consonant
5	E	e,ee	Vowel
6	F	F	Consonant
7	G	G	Consonant
8	H	H	Consonant
9	I	i,ii	Vowel
10	J	J	Consonant
11	K	K	Consonant
12	L	L	Consonant
13	M	m	Consonant
14	N	n	Consonant
15	O	o,oo	Vowel
16	P	p	Consonant
17	Q	q	Consonant
18	R	r	Consonant
19	S	s	Consonant
20	T	t	Consonant
21	U	u,uu	Vowel
22	C	c	Consonant
23	W	w	Consonant
24	X	x	Consonant
25	Y	y	Consonant
26	Z	z	Consonant
27	CH	ch	Double Consonant
28	DH	dh	Double Consonant
29	NY	ny	Double Consonant
30	PH	ph	Double Consonant
31	SH	sh	Double Consonant

3.4 Afan Oromo Word Class

Recent works have stated that Afan Oromo has five word classes: noun, ad-position, adverb, conjunction and verb. Each of these classes again can be divided into other sub-classes. For instance, noun class is categorized as proper noun, common noun, pronoun, Preposition and postpositions are sub classes of ad- positions. The subclasses in turn can be divided into subclasses and the subdivision process may continue iteratively depending on the level and aim of the investigation (Mamo and Meshesha, 2009).

3.4.1 Noun (‘*Maqaa’*)

Nouns are any word that can be used to name or identify place, object or ideas. Two types of grammatical genders exist in Afan Oromo nouns (Tesfaye and Abebe, 2010). These are masculine and feminine, and the entire nouns of the language belong to one of these gender categories. Similarly, there are two numbers (singular and plural) which can be identified by the morpheme it adds. Plural form of a given noun can be formed by adding suffix to the root noun. Various types of suffixes can be added to transform a singular noun to its plural form. All of these suffixes change singular noun to plural without variation in meaning. The last vowel of the singular noun is dropped before the suffix is added. The suffixes which include -oota;- [w]wan,-een, -eelee, -iin,-[a]an,-oolee, eewwan,-iilee etc are used to form plural.

English language uses two types of articles known as definite article (the) and indefinite article (a, an, some, any). In case of Afan Oromo, there are no articles that will be inserted before nouns. However, the suffix – (t) icha can be used in the same context as English language ‘the’ in masculine and – (t) ittii for feminine nouns. Ending vowels of nouns are dropped before these suffixes are added to the noun. For example: mucaa meaning ‘boy’, *mucicha* ‘the boy’,*mana* ‘house’ ,*manicha* ‘the house’ ,*durba* ‘girl’ *durbitti* ‘the girl’.

3.4.2 Pronoun (‘*Maqdhaala’*)

Pronouns are words that can be used in place of nouns. Similar to that of nouns, pronouns have number and gender. For example, *ishee*/*isii* which means ‘she’ is feminine (singular) whereas ‘isa’ which means 'he' is masculine (singular) and ‘*isaan’* which means 'they' is plural can be masculine or feminine. Pronouns can also be categorized based on their functions and meanings in the sentence. These are personal pronoun, possessive pronoun, demonstrative pronoun, relative pronoun or reciprocal pronoun (Nagi, 2006).

3.4.3 Verb (‘*Xumura*’)

In Afan Oromo verbs are words that are used to indicate some action or event occurrence within time boundaries (Nagi, 2006). It can be transitive, intransitive, modals and auxiliary verbs. Transitive verbs are those verbs which transfer message to complement or object whereas, intransitive verbs do not transfer message to complement and hence, do not have complement or object. The following examples illustrate this fact. *Tolaan Konkolataa bite*. This means ‘Tola bought a Car. Since the action of buying was transferred to object *Konkolataa* ‘Car, bite is transitive verb.

3.4.4 Adverbs (‘Ibsa *Xumura*’)

Adverbs are any words that explain or modify verbs (Nagi, 2006). These can be adverbs of time, place, manner, frequency etc. Adverbs precede verbs they modify in Afan Oromo. For example *Tolaa dafe dhufe*. This means ‘Tola came quickly’. *Dafe*‘quickly’ is an adverb. *Jaalannen bor deemti*. This means ‘*Jalane* will go tomorrow’. *Bor*‘tomorrow’ is an adverb. *Kananiisaan yeroo hunda ni mo’a*ta. This means ‘*kananisaan* wins every time. *Yeroo hunda*‘every time’ is an adverb.

3.5 Afan Oromo Punctuation Marks

Punctuation is placed in text to make meaning clear and reading easier. Analysis of Afan Oromo texts reveals that different punctuation marks follow the same punctuation pattern used in English and other languages that follow Latin Writing System. Like English, the following are some of the most usually used punctuation marks in Afan Oromo language (Tesema and Tamirat, 2017). For example, ‘*qooduu’* comma (,) is used to separate listing of ideas, concepts, names, items, etc and the full ‘*tuqaa’* stop (.) in statement, the ‘*mallattoo gaaffii’* question mark (?) in interrogative and the ‘*mallattoo raajeffannoo’* exclamation mark (!) in command and exclamatory sentences mark the end of a sentence.

3.6 Afan Oromo Morphology

Like in a number of other African and Ethiopian languages, Afan Oromo has a very complex and rich morphology (Tune et at., 2008). It has the basic features of agglutinative languages involving very extensive inflectional and derivational morphological processes. In agglutinative languages like Afan Oromo, most of the grammatical information is conveyed through affixes, such as, prefixes, infix and suffixes attached to the root or stem of words. Although Afan Oromo words have some prefixes and infixes, suffixes are the predominant morphological features in the language.

Almost all Afan Oromo nouns in a given text have person, number, gender and possession markers, which are concatenated and affixed to a stem or singular noun form. In addition, Afan Oromo noun plural markers or forms can have several alternatives. For instance, in comparison to the English noun plural marker, *s (-es)*, there are more than ten major and very common plural markers in Afan Oromo including: *-oota, -oolii, -wwan, -lee, an, een, -eeyyii, -oo,* etc.). As an example, the Afan Oromo singular noun *mana* (house) can take the following different plural forms: *manoota (mana + oota), manneen (mana + een), manawwan (mana + wwan)*. The construction and usages of such alternative affixes and attachments are governed by the morphological and syntactic rules of the language(Tune et at., 2008). Afan Oromo nouns have also a number of different cases and gender suffixes depending on the grammatical level and classification system used to analyze them. Frequent gender markers in Afan Oromo include *-eessa/-eettii, -a/-ttii* or *–aa/tuu.* Consider the following example.

Table 3. 2: Gender and suffix construction in Afan Oromo letters.

Afan Oromo	**Construction**	**Gender**	**English**
Sangoota	*Sangaa +oota*	Male	Ox
Jaarsolii	*Jaarsa +olii*	Male	Elder
Obboleessa	*Obol +essa*	Male	Brother
Beektuu	*Beek +tuu*	Female	Knowledgeable

3.7 Numerals ('*Lakkoofsa'*)

Numerals include words that refer to number or quantity of something (Mamo and Meshesha, 2009). It can be cardinals such as *tokko*(one), lama(two) or it can be ordinals like tokkoffaa(first) *lammaffaa*(second). As discussed in (Tilahun, 2001), numerals in a sentence follow the category they describe their quantity or amount. Ordinals in Afan Oromo are formed by adding suffix **–ffaa** to the cardinal numerals. Consider the following sentences:

1. ***Tolasaan sangoota lama bite*:** This means 'Tolasa bought two oxen' .The word *lama* is cardinal numeral in the sentence. It describes the quantity of *sangootaa* "oxen".
2. ***Caaltuun daree isheetti tokkoffaa baate*:** This is to mean 'Chaltu stood first from her classes'. In this case, the word *tokkoffaa* is ordinal which is to mean 'first'. It is formed from cardinal *tokkoo* 'one' by adding affixes **–ffaa**.

Afan Oromo spellchecker uses the collected Afan Oromo corpus to detect and correct the misspelled words. However in Afan Oromo the absence of standard collected corpus is big challenge to design context based spellchecker for Afan Oromo. Furthermore, the other difficulty for the Afan Oromo language is the different letters repeated that describe a single word. In this case the similar word can be written more than once with different letters like '*eessa'* and 'eecha'. In data preparation, we tried to cover and include repeated letters and morphological complex.

CHAPTER FOUR

MODEL OF CONTEXT BASED SPELLCHECKER FOR AFAN OROMO WRITING

4.1 Introduction

In the previous chapters we describe some of the related works on spell error detecting and correcting mechanisms for different languages and basic features of the Afan Oromo language to be taken into consideration before designing the model of the Afan Oromo spell checker and corrector are discussed. This chapter describes method employed in this research, model architecture, techniques used and algorithm selected for context based spell checker for Afan Oromo writing.

4.2 Spell checking Model

Context based spellchecker for Afan Oromo (CBSCAO) model was developed to detect and correct non-word and real-word error. CBSCAO designing followed three steps process which involves: (a) Error detection for identifying misspelled words from user words, (b) Error correction for finding candidate for correcting the misspelled word and (c) Ranking candidates for selecting the best result with high similarity score and maximum bigram probability from the candidate suggestions. Generally, the Figure 4.1 describes the architecture flow to design Context based spellcheck for Afan Oromo writing system. As shown in figure 4.1, first the model detects the misspelled words from the preprocessed user words. From user word non-word error detected by using dictionary lookup method from prepared dictionary. Whereas, real word error detected by using bigram sequence analysis from bigram model. On other hand, the second step for the CBSCAO model is error correction module try to provided candidate for misspelt words. The model applied Levenshtein algorithm to correct non-word error by computing distance between user word (misspelt word) and dictionary words (correct words). While, bigram probabilities applied to correct real word errors, by computing the statistical information of each individual words.

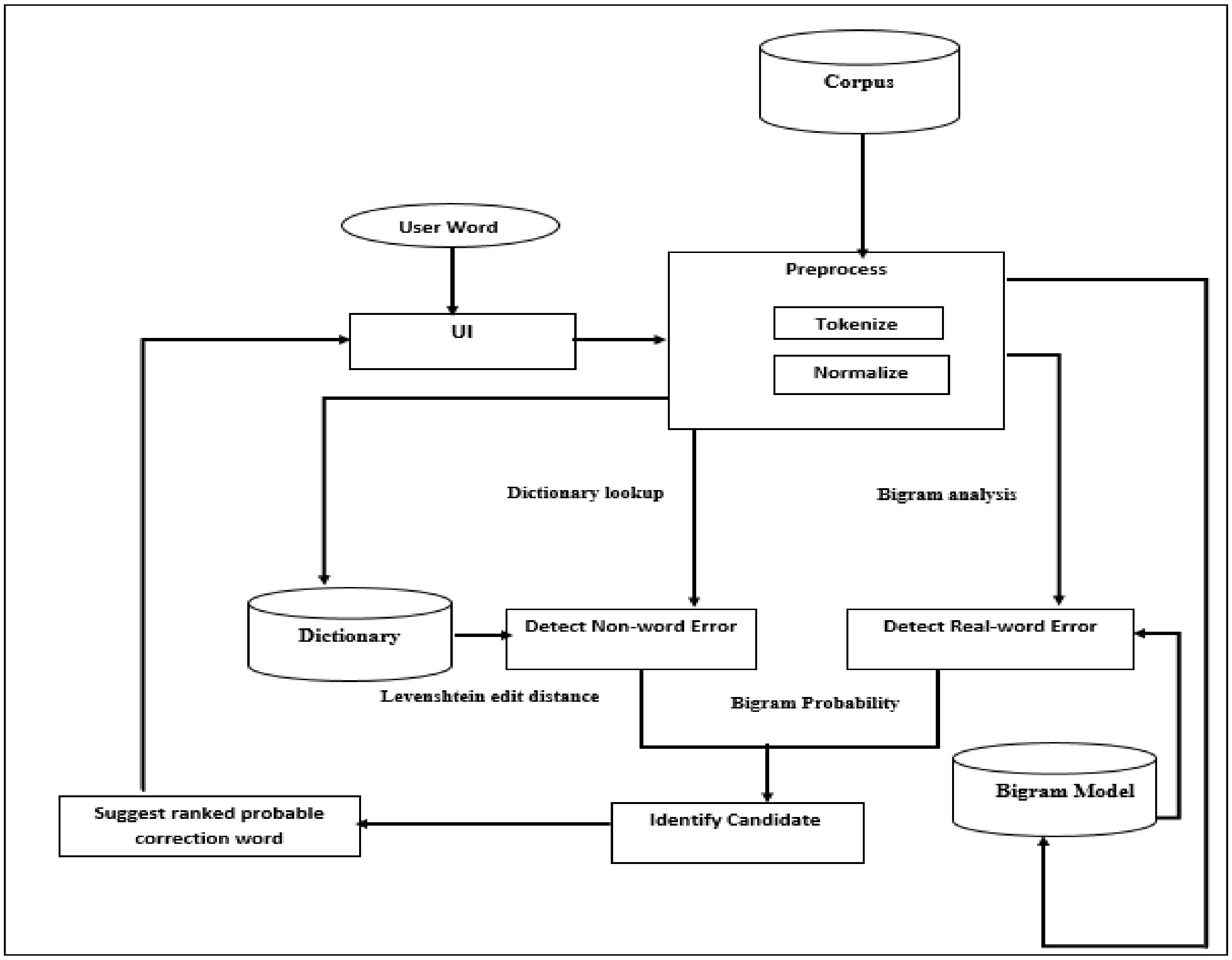


Figure 4.1: Context Based Spell Checker for Afan Oromo Architecture.

4.3 Dictionary and Bigram model Construction

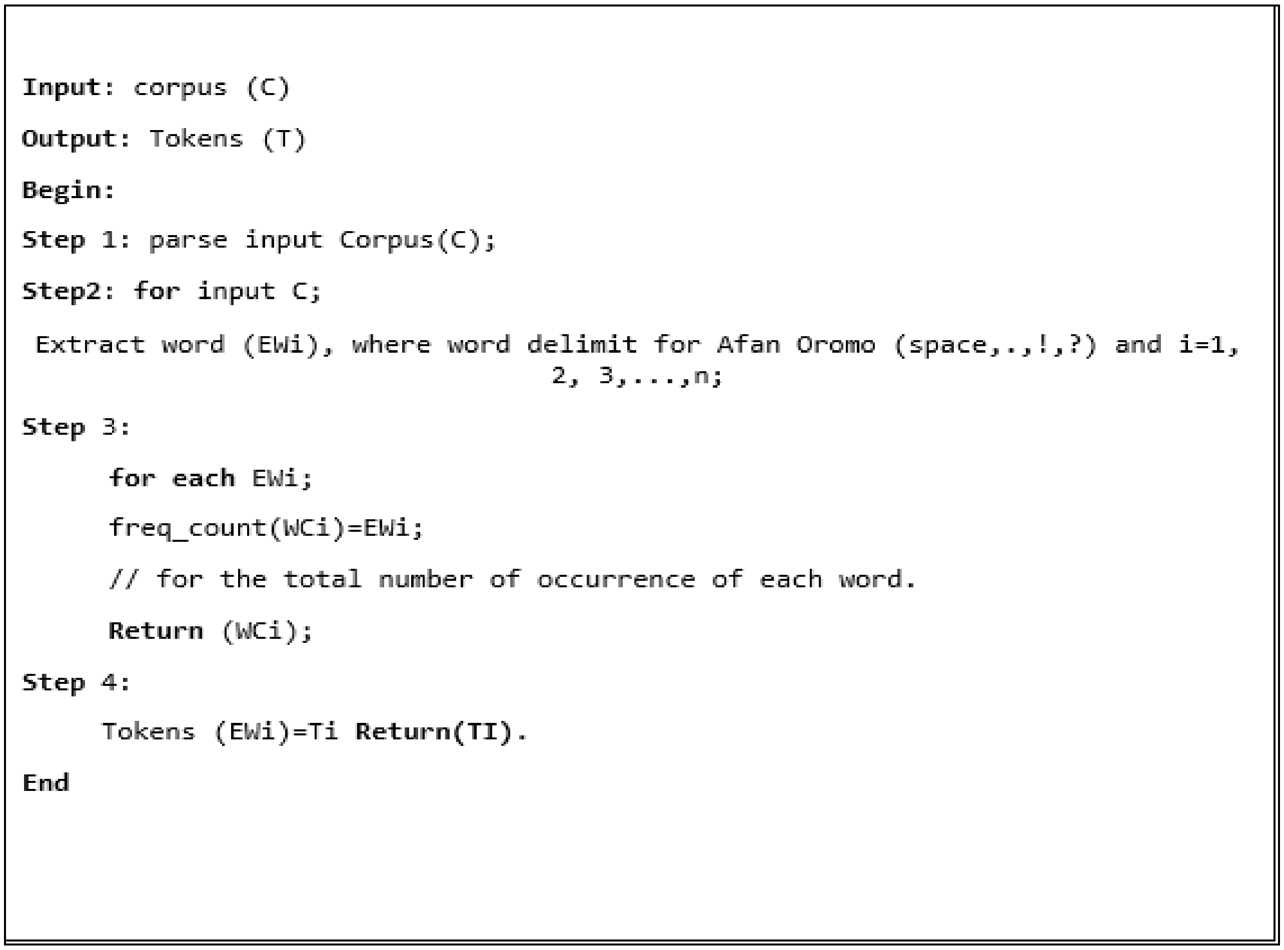
As describes in section 1.6.3, corpus was collected from different sources. Collected corpus was preprocessed (tokenized and normalized) and manually cleared from any kind of unnecessary errors and each word in the corpus were free from spelling errors which is valid to represent Afan Oromo words as shown in above figure 4.1. Well prepared words were stored in the form of the dictionary and accepted as corrected words and also the corpus used for constructing bigram model to detect and correct real word error(see figure 4.7).

Dictionary is list of corrected words help to detect and correct misspelled words for non-word error. As describe in section 2.4.1 dictionary size affect the response time, since it check user word against all dictionary words as shown in figure 4.4. On another hand, bigram model stored with the sequence of two words and accepted as contextually goes with each other in Afan Oromo writing system. It help to detect and correct real word error from the user prepossessed words.

4.3.1 Preprocessing (Tokenization and Normalization)

In order to detect user words, as well as to prepare dictionary and compute bigram, it is needed to split the sentence into small parts called tokens (Asghar et al., 2013).This is the initial module of the framework used to break up the sentences into tokens (words) and identifies the typographical errors (focal words) using dictionary (Kundi, et al., 2014). Understanding word boundary is crucial in the tokenization process (Jurish and Würzner 2013). According to Tesema and Temirat (2017), Afan Oromo writing system has its own word boundary and nearly a word boundary of sentence in Afan Oromo is similar with the English language. Afan Oromo uses the space to shows the end of one word and also, exclamation point (!), period (.), question mark (?), brackets (), quotes (‘’) are being used to show a word boundary.

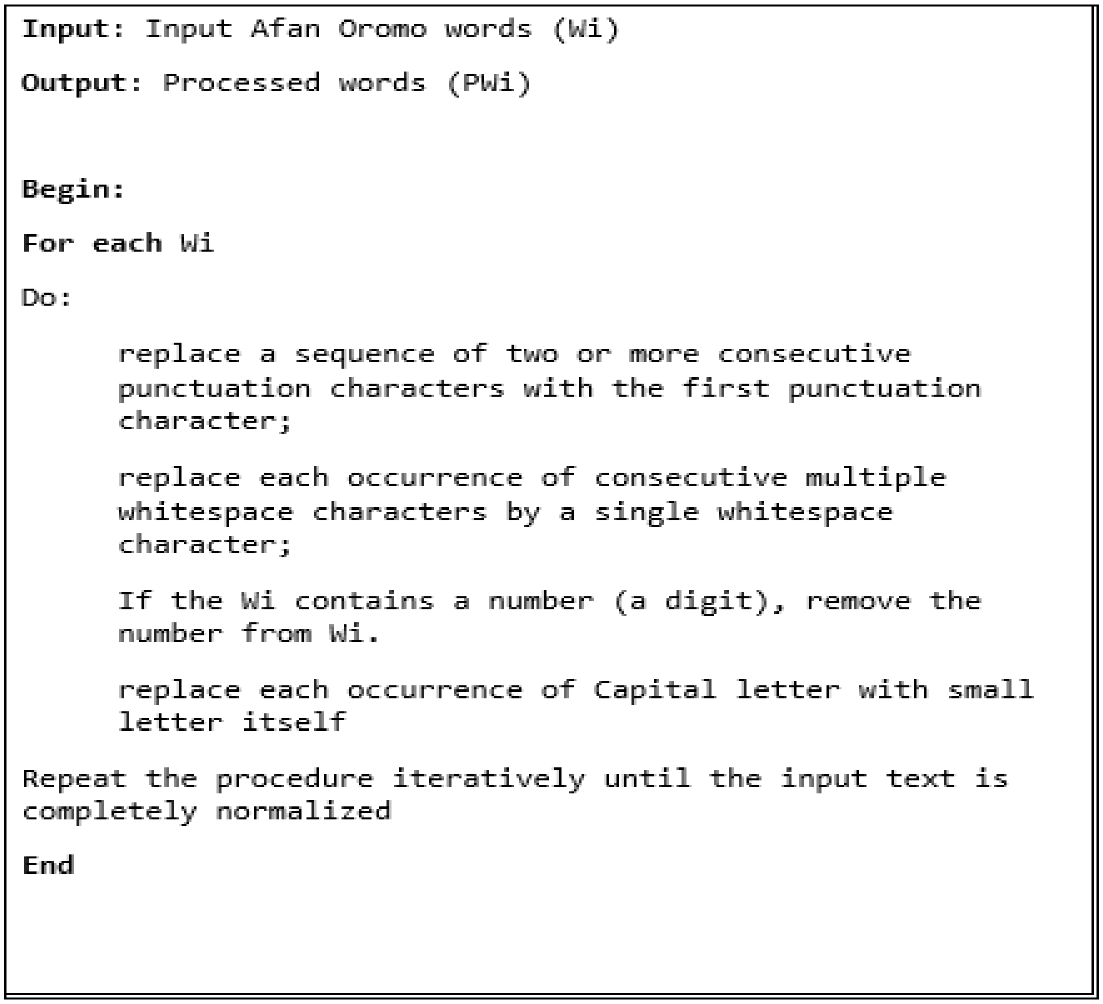
As the described in Figure 4.2, Afan Oromo sentence was split by using tokenize algorithm done by the String Tokenizer of java built in class. For instance if we have the sentence ‘*tokkumman keenya humna’* in our corpus, the tokenized into set of words, like ‘*tokkumman’*, ‘*keenya’* and ‘*humna’.*



```
Input: corpus (C)
Output: Tokens (T)
Begin:
Step 1: parse input Corpus(C);
Step2: for input C;
 Extract word (EWi), where word delimit for Afan Oromo (space,.,!,?) and i=1,
                                 2, 3,...,n;
Step 3:
      for each EWi;
      freq_count(WCi)=EWi;
      // for the total number of occurrence of each word.
      Return (WCi);
Step 4:
     Tokens (EWi)=Ti Return(TI).
End
```

Figure4.2: Tokenized Algorithm.

On other hand, normalizer component mainly deals with parsing words of a document into constituent words, usually by considering white spaces and punctuation marks (punctuation mark usage in Afan Oromo is similar to that of English) (Seretan et al., 2014). Beside, text normalized help to the capital letter-small letter problems from the given words. To identify the misspelled words automatically the words are converted to small letters. For instance, if the users enter "*baGa naGaan DhuftE*", the model automatically normalize to the "*baga nagaan dhufte*". And also it will normalize non-standard words like number, "*Kottu*11", into "*Kottu*". This aspect was handled using Java's HashMapdata structure.



```
Input: Input Afan Oromo words (Wi)

Output: Processed words (PWi)


Begin:

For each Wi

Do:

      replace a sequence of two or more consecutive
      punctuation characters with the first punctuation
      character;

      replace each occurrence of consecutive multiple
      whitespace characters by a single whitespace
      character;

      If the Wi contains a number (a digit), remove the
      number from Wi.

      replace each occurrence of Capital letter with small
      letter itself

Repeat the procedure iteratively until the input text is
completely normalized

End
```

Figure 4.3: Normalized Algorithm.

4.4 Error Detection Algorithm

As discussed in section 2.4, error detection process usually consists of checking if an input string is a valid dictionary word or not valid (for non-word) and check if bigram words available or not (contextually). For those not valid words or not available, the model accept as the incorrect words and flagged as misspelled words for the user. As shown in the figure 4.1, dictionary lookup method was used to identify non-words error whereas, bigram analysis identify real words error from the user words.

4.4.1 Dictionary Lookup Algorithm

As described in section 2.4.1, dictionary look up method used to identify the word that do not found in lexicon entries (non-word error) from the user word. Preprocessed user words detected by using dictionary lookup method in order to identify non-word error (see figure 4.1). As shown in figure 4.4 the dictionary lookup compare each token (word) in a user words against to dictionary words which contains correct spelled words. The tokens that match elements of the dictionary are considered as correct words, otherwise the tokens that didn't match the list of words in the dictionary are flagged as misspelled words.

To implement dictionary lookup Hash function was used in this study. As discussed in section 2.4.1 Hash function is preferable in order to minimize response time of dictionary lookup method. As a result for this study dictionary lookup method that implemented by java Hash function was selected as shown in figure 4.5.

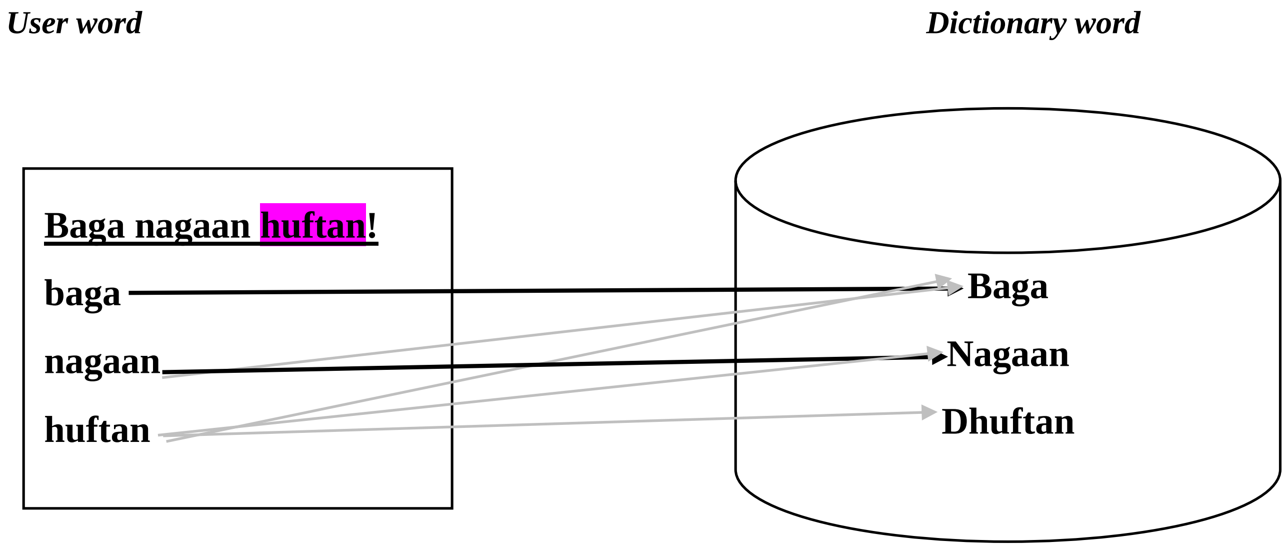
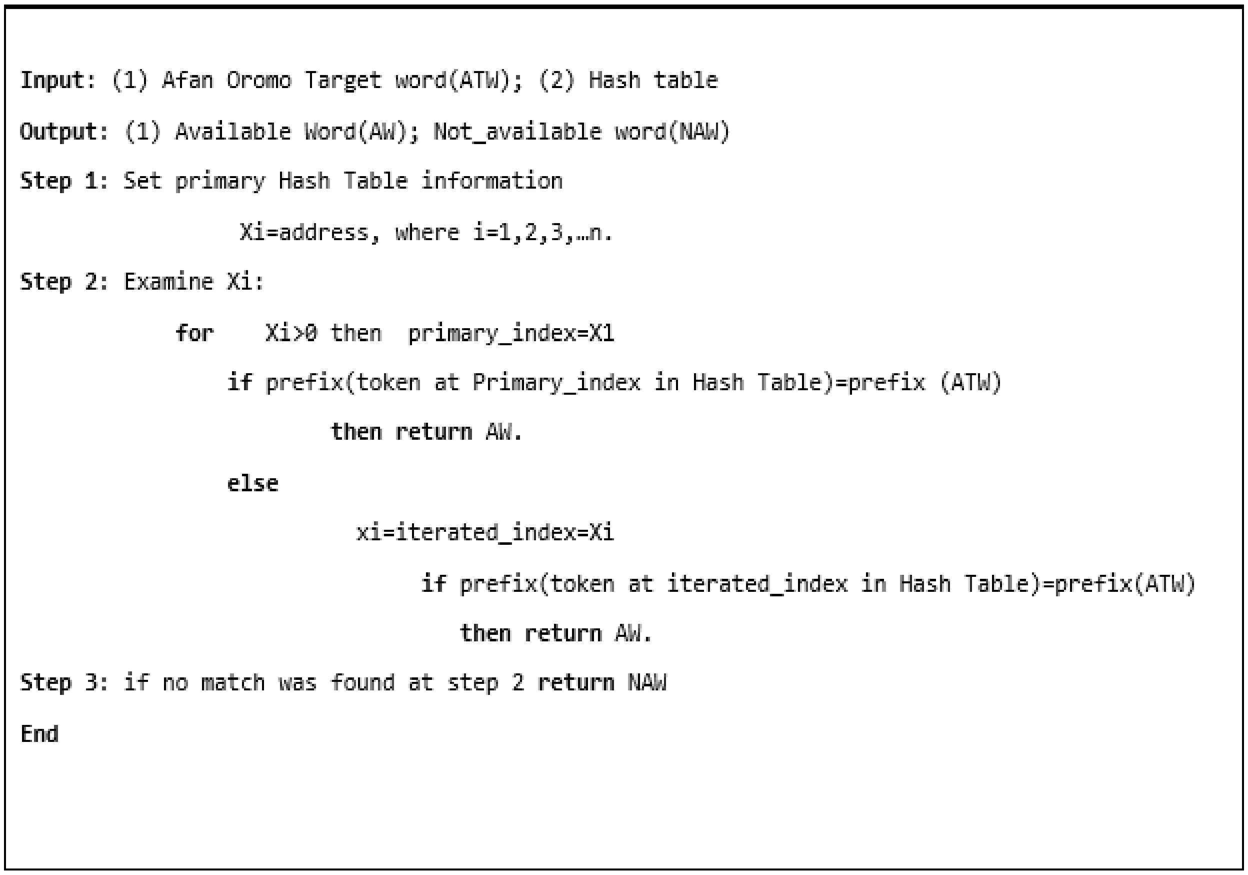


Figure 4.4: Dictionary lookup method

From above figure 4.4 "*baga*", "*nagaan*" and "*huftan*" are user words and "*baga*", "*nagaan*" and "*dhuftan*" are dictionary word (list of corrected words). Dictionary lookup identify the word "*baga*" are "*nagaan*" are correct words whereas, "*huftan*" misspelled word. According to the given example from above figure, the bold line show that the user word and dictionary words meet each other, and the fade line show that, the user word didn't meet with dictionary words, as a result the model detected as misspelled words.



```
Input: (1) Afan Oromo Target word(ATW); (2) Hash table
Output: (1) Available Word(AW); Not_available word(NAW)
Step 1: Set primary Hash Table information
              Xi=address, where i=1,2,3,…n.
Step 2: Examine Xi:
          for   Xi>0 then  primary_index=X1
              if prefix(token at Primary_index in Hash Table)=prefix (ATW)
                      then return AW.
              else
                        xi=iterated_index=Xi
                            if prefix(token at iterated_index in Hash Table)=prefix(ATW)
                               then return AW.
Step 3: if no match was found at step 2 return NAW
End
```

Figure 4.5: Dictionary Lookup Algorithm.

4.4.2 Bigram Analysis Algorithm

Real-word errors detection are aim at identifying the errors that occur when a user mistakenly types a correctly spelled word after another was intended(Hirst and Budanitsky, 2003). As mentioned in section 2.6 to detect the real word errors in the sentences we considering the preceding and following of words. For instance, "*He come form America"* instead of" *He come from America"*. 'Form' flagged as error but it found in the English words.

For this study, we implement Bigram statistics frequency count (shown in equation 4.1) which help to detect real word errors by counting how many times the occurrence of the word bigram with each other to decide misspelling words from the given words. If the words didn't occur the model flagged as real-word error.

$$Bigram(W1W2W3) = (\mathrm{W1W2})(\mathrm{W2W3}) \qquad (4.1)$$

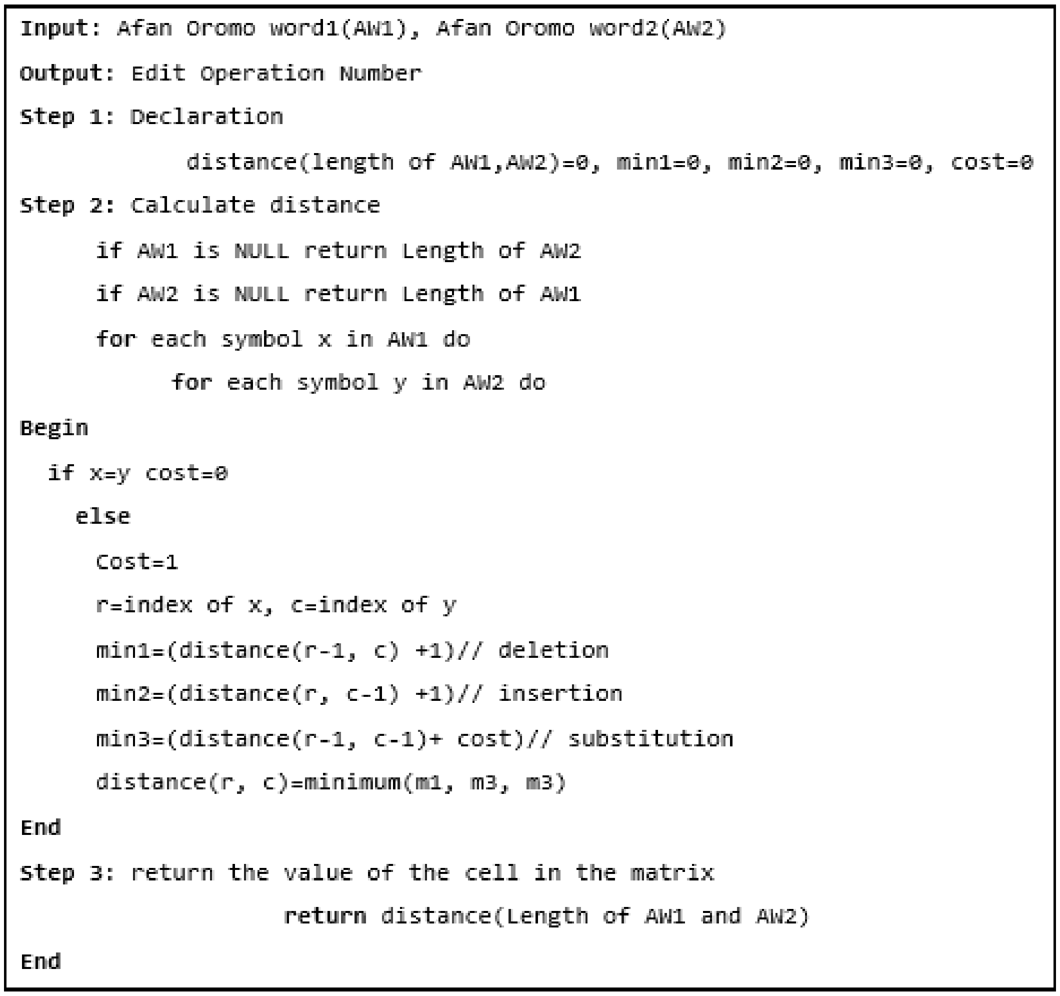
For example "*Bilisummaa ummata isaatiif arsaa of godhe"*. frequency(*Bilisummaa ummata*) = 1, frequency(*ummata isaatiif*) = 1, frequency(*isaatiif arsaa*) = 1, frequency(*arsaa of*) = 1, frequency(*of godhe*) = 1 from the given sentence. If the frequency of the bigram is zero, the given word is real word error.

4.5 Error Correction Algorithm

Error correction is the procedure of correcting an error once it has been detected (Hirst and Budanitsky, 2003). Once a string has been detected as an error, an error correction algorithm aims at finding candidate corrections for the erroneous word. Generally, the correction module used for this study was Levenshtein edit distance to correct non-word and Bigram probability to correct real word as shown in figure 4.1.

4.5.1 Levenshtein Edit Distance

As described in section 2.3.1, Levenshtein edit distance used to correct non-word error. For this study, Levenshtein algorithm was implemented to find the minimum operation which includes insertion, deletion and substitution to find the candidate correction for the misspelled one as shown in figure 4.6. Insertion occurs when a letter needs to be inserted a misspelled word resulting in a correctly spelled word. But deletion occurs when a letter needs to be deleted from a misspelled word in order to result in a correctly spelled word. Substitution indicates to the replacement of a letter in the erroneous word by a correct letter, thus the resulting in the correctly spelled word.



```
Input: Afan Oromo word1(AW1), Afan Oromo word2(AW2)
Output: Edit Operation Number
Step 1: Declaration
            distance(length of AW1,AW2)=0, min1=0, min2=0, min3=0, cost=0
Step 2: Calculate distance
      if AW1 is NULL return Length of AW2
      if AW2 is NULL return Length of AW1
      for each symbol x in AW1 do
            for each symbol y in AW2 do
Begin
  if x=y cost=0
    else
      Cost=1
      r=index of x, c=index of y
      min1=(distance(r-1, c) +1)// deletion
      min2=(distance(r, c-1) +1)// insertion
      min3=(distance(r-1, c-1)+ cost)// substitution
      distance(r, c)=minimum(m1, m3, m3)
End
Step 3: return the value of the cell in the matrix
                    return distance(Length of AW1 and AW2)
End
```

Figure 4.6: Levenshtein Algorithm.

As above Figure 4.6 describes, the (minimum) edit distance between two strings, s1 and s2 is the minimum number of basic operations to convert s1 to s2. Basically, in the model, there are two string; given word (misspelled word) and corrected word (from dictionary). Here in the model the algorithm needed to convert given word to dictionary word. For example, s1=”*rottu*”, represents misspelled word from the user and need to be correct. And s2=”*kottu*”, holds the correct word from the dictionary.

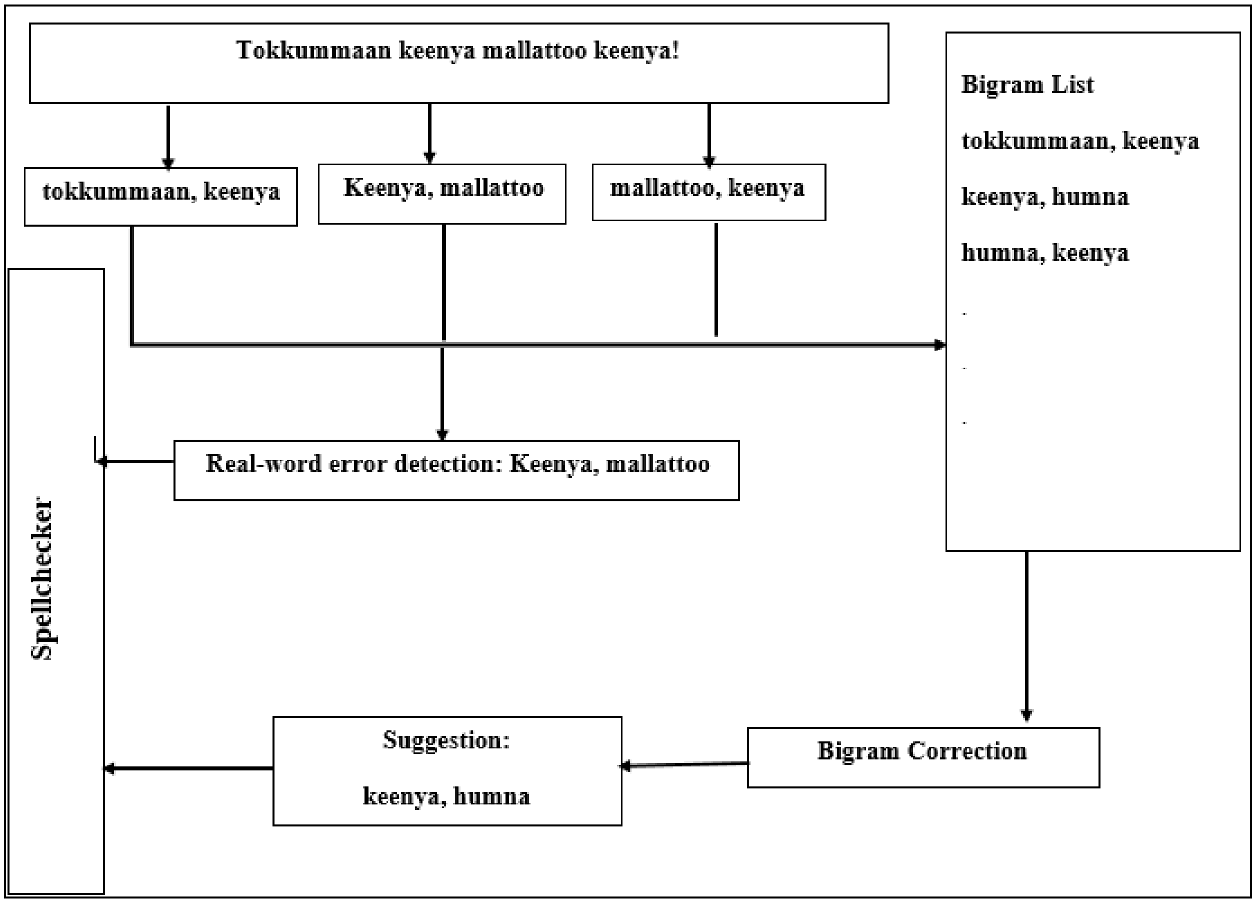


Figure 4.7**:** Bigram word detection and correction

4.5.2 Bigram Probability Algorithm

On the other hand, real word spelling errors could be corrected using bigram language model. In this research bigram word probability which computed from the collected corpus was applied to correct real word errors.

Here, count (w1) is the number of occurrences of w1 in the corpus, and count (w1; w2) is the number of times w2 immediately follows w1:

$$P(W2/W1) = \frac{count(w1, w2)}{count(w1)} \tag{4.2}$$

As describe on figure 4.7, user insert ‘*Tokkumman keenya mallattoo keenya’* words. But the bigram module that prepared from the corpus contains ‘*keenya, humna* ’, here ‘*mallattoo’* is taken as the error even if its correct Afan Oromo word, because of the limited corpus. When we apply above equation 4.2 we offered the following results:

$$\text{P (mallattoo/keenya)} = \frac{1(mallattoo,keenya)}{2(keenya)} = 0.5$$

4.6 Candidate Ranking

Candidates are those tokens with high similarity to the incorrect word (Ratnasari et al., 2017). In the case of non-word errors, candidate list gated from the stored dictionary by calculating the similarity between the misspelled and corrected list of words. On other hand, real-word errors the candidate token is the one that is more likely to be intended. Candidate list generated from the bigram model competed from the collected corpus, by calculating the probability of occurrence of preceding words after we write first words.

CHAPTER FIVE

EXPERIMENTATION AND EVALUATION

5.1 Introduction

In the previous chapter, an attempt was made to discuss the design of the model of the Context based spell checker for Afan Oromo writing. In this chapter, we implement the tools and algorithm that are describe in previous section to design a model and the experiment was conducted to demonstrate the spelling error detection and correction accuracy. The result of the experiment would be interpreted in this section and the performance of the spelling error detection and correction could be evaluated using evaluation method. Precision and recall were used to evaluate the accuracy, effectiveness and validity of detecting and correcting spelling errors based on the training and testing texts that have been used in this experiment.

As discussed in section 1.4, designing a prototype to detect the misspelling words from the given user words is one of the objectives of this work. In order to implement the algorithms and make the necessary experiment on the system we used Java Netbeans IDE 8.2. The developed interfaces which used to detection misspelling and provide the possible correction for Afan Oromo consists some user interface components which used to accomplish the task of this study. These components shown in figure 5.1

5.2 Trained Data

The corpus were prepared with linguistic expert depending on spelling features (Vincze et al., 2008). Similarly, for Afan Oromo we prepare error free corpus to maximize the accuracy and performance of the model by making well understandable and pure words of Afan Oromo vocabulary with linguistic expert on spelling feature. Since collected corpus is contains unnecessary characters and words text preprocessing is very important to clarify corpus.

Moreover, collected corpus used to prepare dictionary that contains the **14,896** list of Afan Oromo words arranged alphabetically as well as, **12, 797** bigram words computed from the collected corpus. The bigrams are generated at word level rather than character level which are used to detect and correct the real word errors (see table 5.1).

Table 5. 1: The ten most frequently occurring bigram from corpus.

Word Bigram	**Frequency**	**Probability**
kun bahee	201	0.026
mana jireenyaa	158	0.021
yoo jiraate	151	0.025
beeksifni kun	139	0.024
ta’uu ni	136	0.018
mormu yoo	123	0.016
kan mormu	101	0.017
bahee guyyaa	91	0.013
jiraate beeksifni	87	0.001
jireenyaa Magaalaa	76	0.01

5.3 Experiment and Result

As mentioned in section 1.6 the experiment was conducted in this work to determine the accuracy of the Context based spell checker for Afan Oromo. This section presents how the test data has been collected and the result found after experiment has been conducted. The interface of CBSCAO contains three different components: (1) Input Area, to write the desired words using an input method. In this case a keyboard is used as input method (2) Button, to check whether the inserted words is correct or not (3) Mouse pointer, to provide the candidate for the misspelled words when the user clicks on the error words as show in figure 5.1.

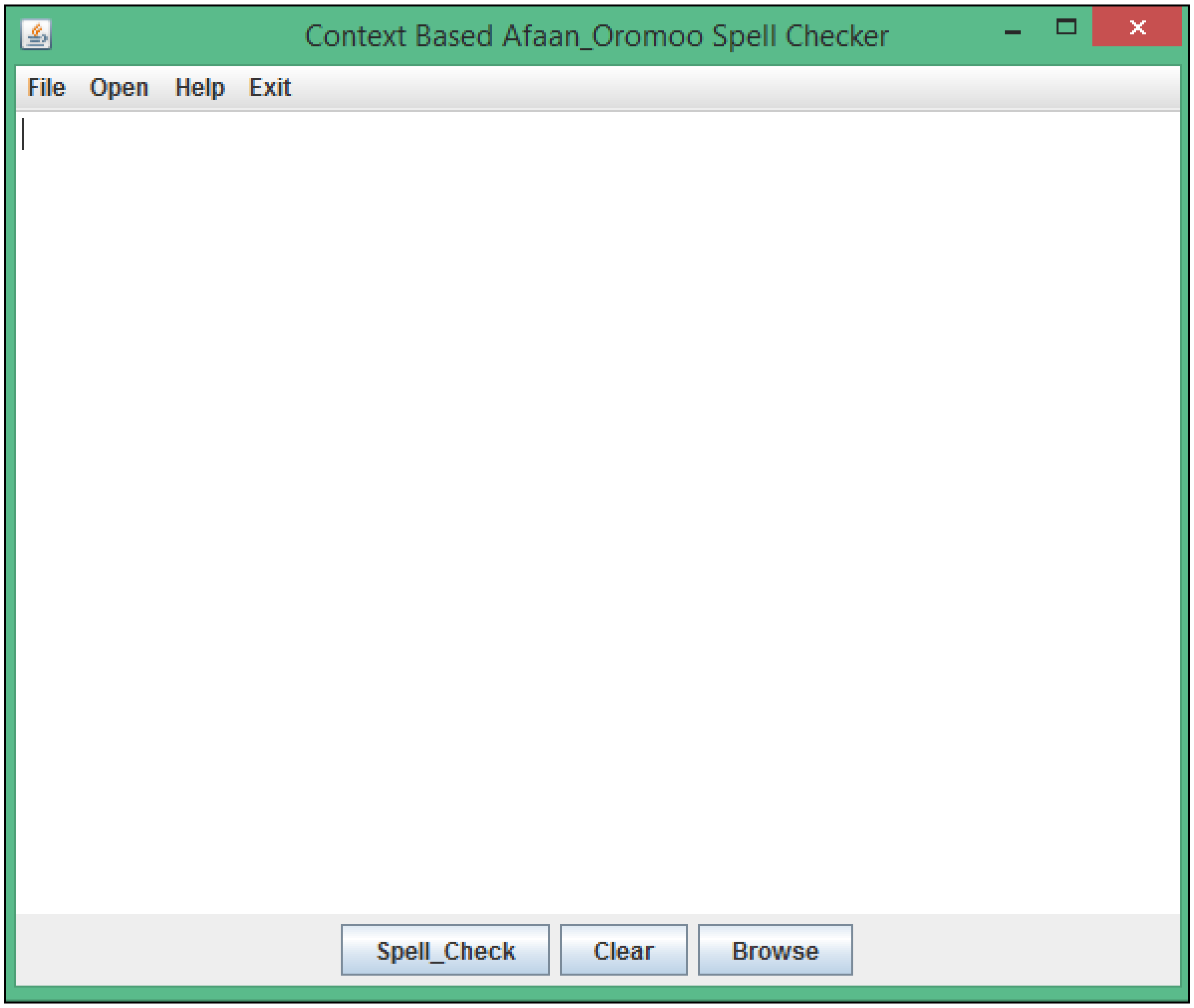
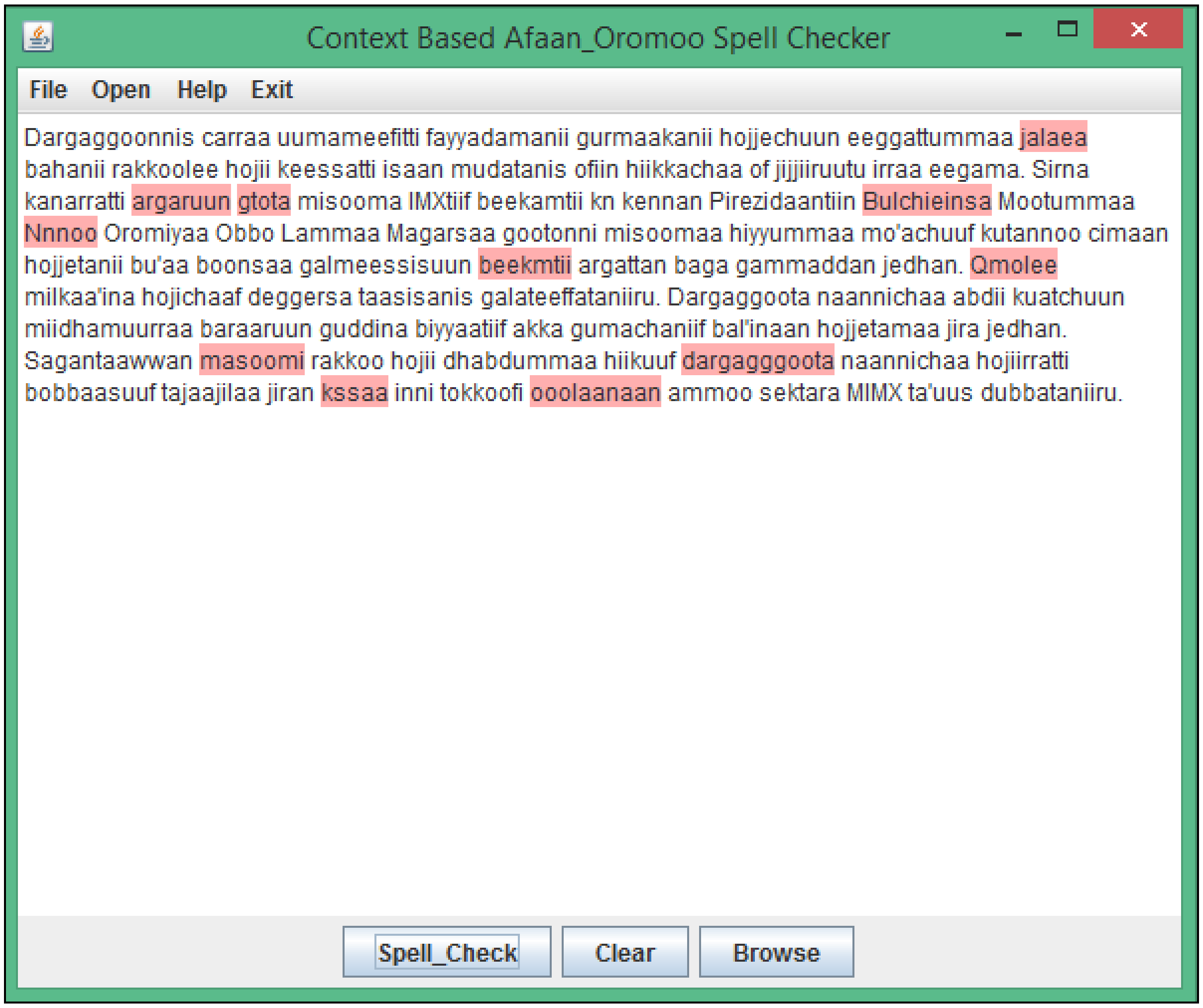


Figure 5. 1: Context Based Spellcheck for Afan Oromo

CBSCAO model has text area which takes input from the users and input text is directly typed in the text area. Since, the model is interactive it wait for button "spell_check" click in order to detect the error. The error detection module responsible to preprocessing and compare the inserted words with dictionary and bigram model. For those didn't found in the list the model accept as the misspelled words, but for others the model leave as it is. As the researcher mentioned in section 2.4, error detection was executed by using two ways. The first one is dictionary lookup for non-word and the second is of bigram analysis for real words.

5.3.1 Error Detection

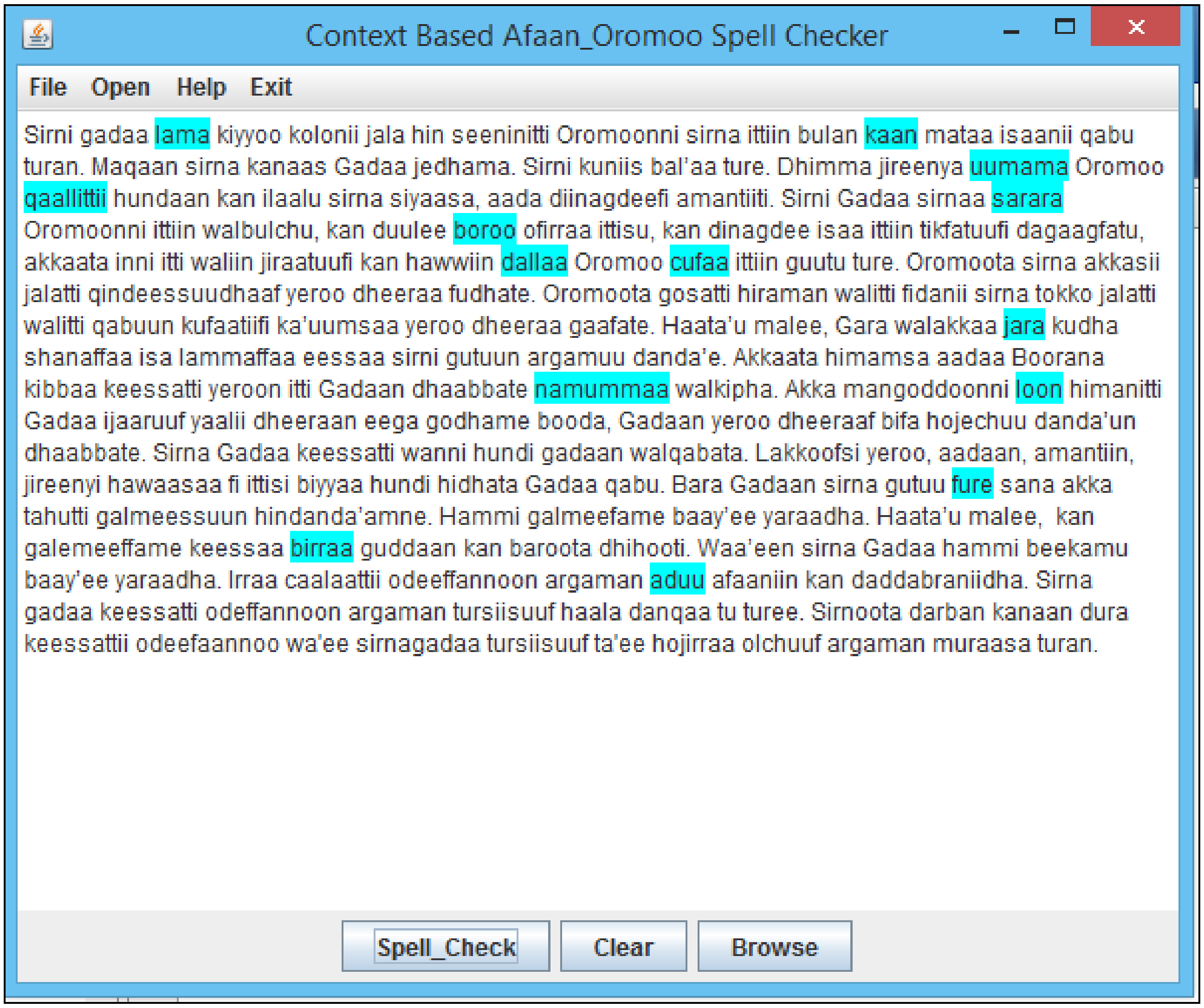
The model detect error at word level to identify non-word spelling errors that not found on the list of dictionary by dictionary lookup method (see section 2.4.1). In the user words the word that does not exist in the dictionary is detected as misspelled words and highlighted by the pink color, otherwise the model accept as corrected words. As shown in the figure 5.2, the word *'jalaea'*, *'argaruun', 'gtota', 'Bulchieinsa', 'Nnnoo', 'beekmtii', 'Qmolee', 'masoomi', 'dargagggoota', 'kssaa' and 'ooolaanaan'* are detected as misspelled words from the given sample test because they didn't found in the prepared dictionary. For misspelled words *'Bulchieinsa'* detected as misspelling words , because of extra letter 'e' added in the middle of 'ii' instead of '*Bulchiinsa'* which means government. The model specify the word didn't found in the list of the dictionary as non-word error and highlighted.



Dargaggoonnis carraa uumameefitti fayyadamanii gurmaakanii hojjechuun eeggattummaa jalaea bahanii rakkoolee hojii keessatti isaan mudatanis ofiin hiikkachaa of jijjiiruutu irraa eegama. Sirna kanarratti argaruun gtota misooma IMXtiif beekamtii kn kennan Pirezidaantiin Bulchieinsa Mootummaa Nnnoo Oromiyaa Obbo Lammaa Magarsaa gootonni misoomaa hiyyummaa mo'achuuf kutannoo cimaan hojjetanii bu'aa boonsaa galmeessisuun beekmtii argattan baga gammaddan jedhan. Qmolee milkaa'ina hojichaaf deggersa taasisanis galateeffataniiru. Dargaggoota naannichaa abdii kuatchuun miidhamuurraa baraaruun guddina biyyaatiif akka gumachaniif bal'inaan hojjetamaa jira jedhan. Sagantaawwan masoomi rakkoo hojii dhabdummaa hiikuuf dargagggoota naannichaa hojiirratti bobbaasuuf tajaajilaa jiran kssaa inni tokkoofi ooolaanaan ammoo sektara MIMX ta'uus dubbataniiru.

Figure 5. 2:Non-word error detection.

On the other hand, the real word errors were detected under the consideration of bigram words sequences that comes together and sequence of bigram words does not exist in the bigram list, it's detected as real word errors. The input sentences were breakdown into bigram and bigram words were generated along with its probability information which is used to rank the candidate suggestion to correct the errors. If the bigram words found in the bigram model, there is no error which is considered as valid word but if one of the word does not exist in the bigram word list it is considered as misspelled word and the model flagged by highlight with the cyan color and display misspelled words as displayed on the figure 5.3 below.



Sirni gadaa lama kiyyoo kolonii jala hin seeninitti Oromoonni sirna ittiin bulan kaan mataa isaanii qabu turan. Maqaan sirna kanaas Gadaa jedhama. Sirni kuniis bal'aa ture. Dhimma jireenya uumama Oromoo qaallittii hundaan kan ilaalu sirna siyaasa, aada diinagdeefi amantiiti. Sirni Gadaa sirnaa sarara Oromoonni ittiin walbulchu, kan duulee boroo ofirraa ittisu, kan dinagdee isaa ittiin tikfatuufi dagaagfatu, akkaata inni itti waliin jiraatuufi kan hawwiin dallaa Oromoo cufaa ittiin guutu ture. Oromoota sirna akkasii jalatti qindeessuudhaaf yeroo dheeraa fudhate. Oromoota gosatti hiraman walitti fidanii sirna tokko jalatti walitti qabuun kufaatiifi ka'uumsaa yeroo dheeraa gaafate. Haata'u malee, Gara walakkaa jara kudha shanaffaa isa lammaffaa eessaa sirni gutuun argamuu danda'e. Akkaata himamsa aadaa Boorana kibbaa keessatti yeroon itti Gadaan dhaabbate namummaa walkipha. Akka mangoddoonni loon himanitti Gadaa ijaaruuf yaalii dheeraan eega godhame booda, Gadaan yeroo dheeraaf bifa hojechuu danda'un dhaabbate. Sirna Gadaa keessatti wanni hundi gadaan walqabata. Lakkoofsi yeroo, aadaan, amantiin, jireenyi hawaasaa fi ittisi biyyaa hundi hidhata Gadaa qabu. Bara Gadaan sirna gutuu fure sana akka tahutti galmeessuun hindanda'amne. Hammi galmeefame baay'ee yaraadha. Haata'u malee, kan galemeeffame keessaa birraa guddaan kan baroota dhihooti. Waa'een sirna Gadaa hammi beekamu baay'ee yaraadha. Irraa caalaattii odeeffannoon argaman aduu afaaniin kan daddabraniidha. Sirna gadaa keessatti odeffannoon argaman tursiisuuf haala danqaa tu turee. Sirnoota darban kanaan dura keessattii odeefaannoo wa'ee sirnagadaa tursiisuuf ta'ee hojirraa olchuuf argaman muraasa turan.

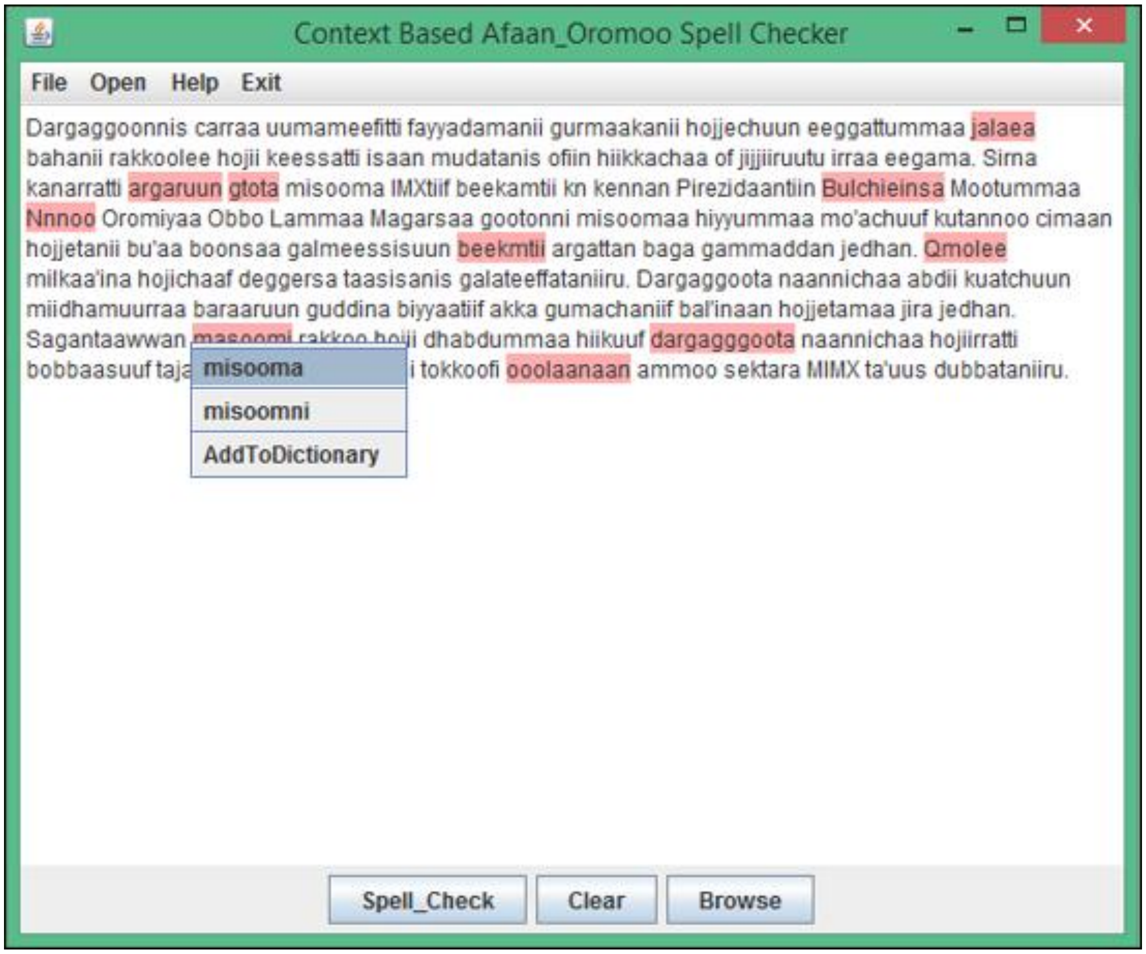
Figure 5. 3: Real-word error detection

As above figure 5.3 shows, the model detect 14 words as misspelled words. For instance, word ‘*kaan’* detected by computing bigram of word *’bulan, kaan’* and ‘*kaan, mata’* and the word ‘*kaan’* contextually not goes with the above mentioned words. In Afan Oromo ‘*bulan kaan mata’* is contextually not goes with each other. However, ‘*bulan kan mata*’ is contextually correct. Therefore, the model detect ‘*kaan’* as the real word error by learning from the collected corpus.

5.3.2 Error Correction

To correct the misspelled words the corrector algorithm is provide a set of possible candidate corrections (see figure 5.4 below). After a word is flagged as wrongly spelled, possible a set of suggestion is provided for the user. For non-word error Levenshtein edit distance responsible to generate the suggestion for misspelled words and take minimum values and rank them accordingly.

Correction module uses dictionary list to provide a spelling suggestions for each word errors flagged as misspelled in the given inputs of words. The errors were corrected and modified through the suggested words that displayed in the popup menu. In the CBSCAOW the model provided at most four candidate list displayed depending on the value they return by computing the distance between them. Additionally, if the user aware of the misspelled words, the model give chance to include the misspelled words as corrected words in to the dictionary.



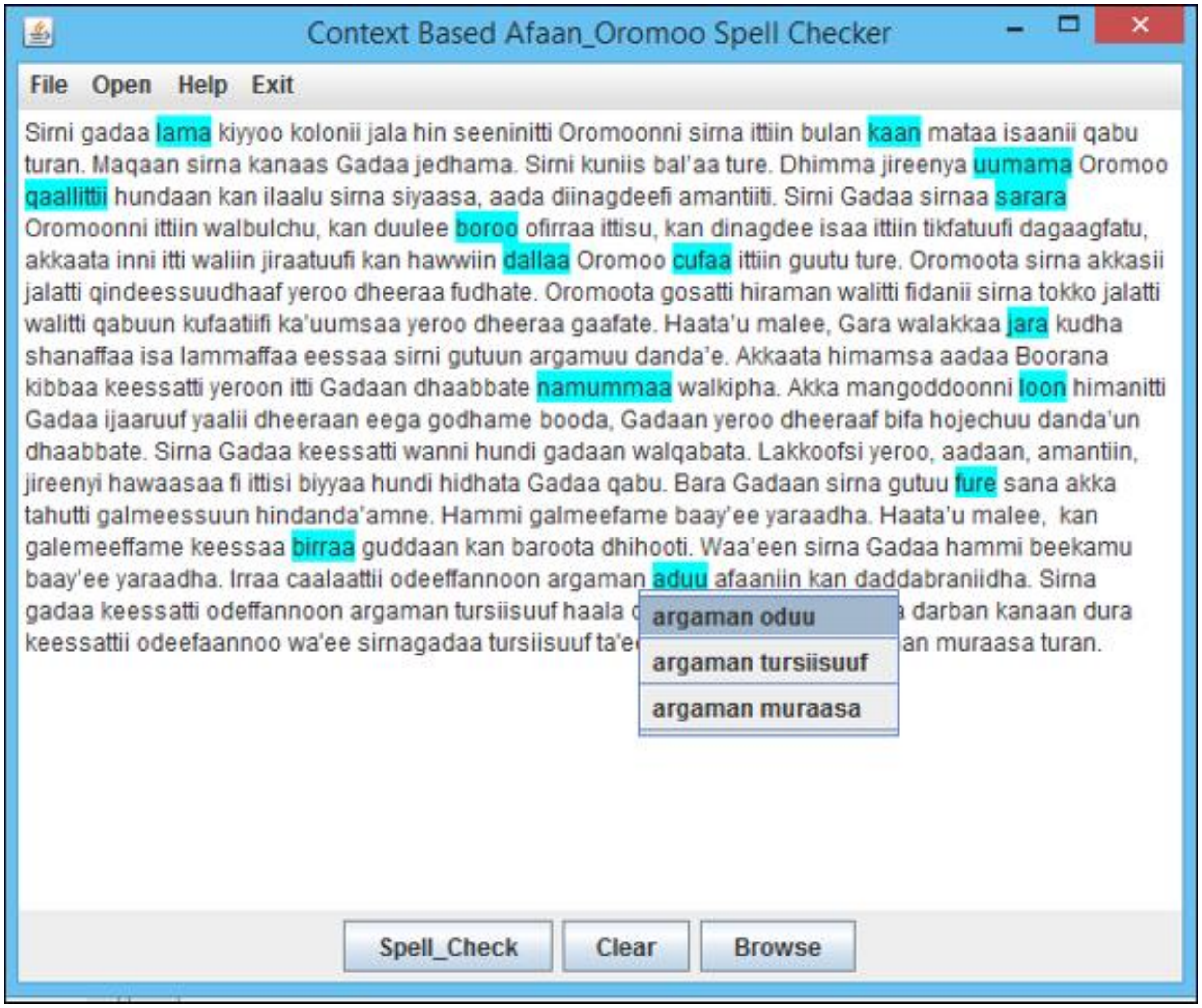
Dargaggoonnis carraa uumameefitti fayyadamanii gurmaakanii hojjechuun eeggattummaa jalaea bahanii rakkoolee hojii keessatti isaan mudatanis ofiin hiikkachaa of jijjiiruutu irraa eegama. Sirna kanarratti argaruun gtota misooma IMXtiif beekamtii kn kennan Pirezidaantiin Bulchieinsa Mootummaa Nnnoo Oromiyaa Obbo Lammaa Magarsaa gootonni misoomaa hiyyummaa mo'achuuf kutannoo cimaan hojjetanii bu'aa boonsaa galmeessisuun beekmtii argattan baga gammaddan jedhan. Qmolee milkaa'ina hojichaaf deggersa taasisanis galateeffataniiru. Dargaggoota naannichaa abdii kuatchuun miidhamuurraa baraaruun guddina biyyaatiif akka gumachaniif bal'inaan hojjetamaa jira jedhan. Sagantaawwan masoomi rakkoo hojii dhabdummaa hiikuuf dargagggoota naannichaa hojiirratti bobbaasuuf taja i tokkoofi oolaanaan ammoo sektara MIMX ta'uus dubbataniiru.

Figure 5. 4: Non-word error correction

To correct the misspelled words, the model provide two candidates for misspelled '*masoomi'* as shown on above figure 5.4, '*misooma'* is first candidate, because it score 2 cost when calculated by the Levenshtein algorithm. Moreover, the cost of substitution 'a' to 'i' and 'i' to 'a' is one (1+1=2). Furthermore, the model provides '*misoomni'* as second candidate for misspelled word, since it score 3 cost when the similarity calculated. The cost of substitution for 'a' to 'i' and 'i' to 'n' are one, also the adding 'i' character also one (1+1+1=3). As a result, the value with cost 2 comes first and the value score 3 stand at second position, because the Levenshtein algorithm take the mini value first.

Additionally, the model provide '*AddToDictionary'* method for the user, if the user correctly know the misspelled flagged by the model is correct Afan Oromo word, flagged words accepted as corrected word by clicking on the 'AddToDictionary', the words automatically added to the dictionary and accepted as correct word.

On the other hand, real word errors were corrected using bigram probabilistic information. After misspelled words was flagged the correction module provides candidate list for real-word error from the corpus collected by computing the probability of the occurrence of words after another one. Then replace the invalid words by clicking each error words at any position and search the alternatives from the popup menu.



Sirni gadaa lama kiyyoo kolonii jala hin seeninitti Oromoonni sirna ittiin bulan kaan mataa isaanii qabu turan. Maqaan sirna kanaas Gadaa jedhama. Sirni kuniis bal'aa ture. Dhimma jireenya uumama Oromoo qaallittii hundaan kan ilaalu sirna siyaasa, aada diinagdeefi amantiiti. Sirni Gadaa sirnaa sarara Oromoonni ittiin walbulchu, kan duulee boroo ofirraa ittisu, kan dinagdee isaa ittiin tikfatuufi dagaagfatu, akkaata inni itti waliin jiraatuufi kan hawwiin dallaa Oromoo cufaa ittiin guutu ture. Oromoota sirna akkasii jalatti qindeessuudhaaf yeroo dheeraa fudhate. Oromoota gosatti hiraman walitti fidanii sirna tokko jalatti walitti qabuun kufaatiifi ka'uumsaa yeroo dheeraa gaafate. Haata'u malee, Gara walakkaa jara kudha shanaffaa isa lammaffaa eessaa sirni gutuun argamuu danda'e. Akkaata himamsa aadaa Boorana kibbaa keessatti yeroon itti Gadaan dhaabbate namummaa walkipha. Akka mangoddoonni loon himanitti Gadaa ijaaruuf yaalii dheeraan eega godhame booda, Gadaan yeroo dheeraaf bifa hojechuu danda'un dhaabbate. Sirna Gadaa keessatti wanni hundi gadaan walqabata. Lakkoofsi yeroo, aadaan, amantiin, jireenyi hawaasaa fi ittisi biyyaa hundi hidhata Gadaa qabu. Bara Gadaan sirna gutuu fure sana akka tahutti galmeessuun hindanda'amne. Hammi galmeefame baay'ee yaraadha. Haata'u malee, kan galemeeffame keessaa birraa guddaan kan baroota dhihooti. Waa'een sirna Gadaa hammi beekamu baay'ee yaraadha. Irraa caalaattii odeeffannoon argaman aduu afaaniin kan daddabraniidha. Sirna gadaa keessatti odeffannoon argaman tursiisuuf haala [illegible] darban kanaan dura keessattii odeefaannoo wa'ee sirnagadaa tursiisuuf ta'e[illegible] an muraasa turan.

Figure 5. 5: Real-word error correction

As shown from figure 5.5, the word '*aduu'* detected as the real word error, even if the word '*aduu'* is Afan Oromo words which means the sun, contextually '*argaman aduu Afaniin'* not goes with each other in Afan Oromo sentences . As a result, the model detect '*aduu'* as real word error and try to provide candidate list from the corpus for misspelled. To correct the error the model provides three candidate list which goes with the '*argaman'* and '*Afanin'* from the corpus. '*oduu'*, '*tursisuuf'* and '*muraasa'* are candidate suggested for the user consequently depending on the probabilistic information as shown in table 5.2 and ranked accordingly.

Table 5. 2: Candidate list for misspelled word 'aduu' from figure 5.5

Rank	**Candidate words between word *'argaman and afaanin' from Corpus***	**Probability from corpus**
1	Oduu	0.014
2	Tursiisuuf	0.0094
3	Muraasa	0.007

As shown in above table 5.2, '*Oduu'* score highest probability value (0.014), as a result it take first place nominee. Whereas, 'Tursiisuuf' hold second place, since it score second highest result (0.0094) and 'Muraasa' score list value than other (0.007), and hold third position nominee.

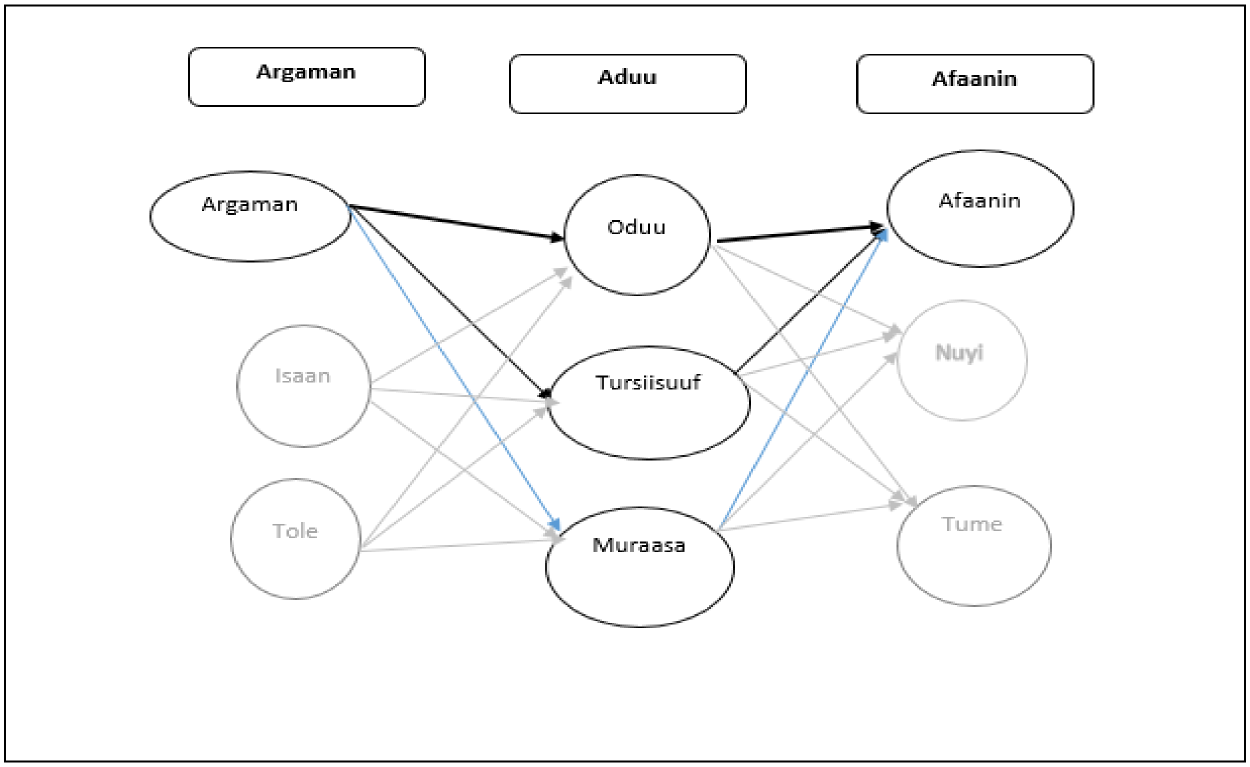


Figure 5. 6: Real word error correction illustrate from figure 5.4.

In above figure 5.6 candidate list '*argaman oduu Afanin', 'argaman tursiisuuf Afanin' and 'argaman muraasa Afanin'* are contextually correct in Afan Oromo sentences. However, depending on the corpus collected the degree of the words goes with each other are different. As shown in above figure, big bold line indicate that highest probability for "*Oduu*", whereas the next bold line indicate the second highest probability for "*Tursisuuf*" and the blue line shows list probability for "*Muraasa*".

5.4 Evaluation

According to Olson and Delen (2008), Precision, Recall and F-measurement the most frequent and basic statistical measures which are widely used to measure the effectiveness of spell checking. In this section the performance of the spell checking application, which was designed in this study is evaluated. The results of the current research can be divided into two parts: the performance of the dictionary based spell checker and the performance of the context-sensitive spell checker.

The evaluation technique has four categories: true positive (TP), false positive (FP), false negative (FN) and true negative (TN) as described in section (4.6). TP indicate valid words recognized by the spelling checker, resulting correct non-flags. TN invalid words recognized by the spelling checker, resulting correct flags (flagged). FN produced when valid words not recognized by the spelling checker, resulting incorrect flags (false flagged). FP invalid words not recognized by the spelling checker, resulting in incorrect non- flags (also called Missed flags). The performance of the spell checker have been developed were evaluated using evaluation metrics for both non-word and real word spelling errors in the written texts.

Moreover, Error precision(EP) also another metrics to measure the effectiveness of spell checker to test valid words which are not recognized by the spell checker system since the words do not exist in the training set but available in test set. To test the performance of the system researcher prepared testing data to measure the model. We collected text from the different sources and generated artificial spelling errors in the test set and marked these errors to evaluate the efficiency of our system. We prepared that, the data set consists of 3078 correctly spelled words and 153 misspelled words. In this sample, 2921 were accepted as a valid word; 157 words were flagged as misspelled words by the system due to the nonappearance of words in the dictionary. On other hand, for the real word errors we prepare 3231 words and out of these 3142 are correctly spelled words and 89 misspelled real words. From the sample test data 2897 accepted as corrected words, whereas 245 not detected by system. Generally, the summary of the test result is presented in table.

Table 5. 3: Experimental results of Afan Oromo spell checker.

Error	**TP**	**TN**	**FN**	**R**	**EP**	**P**	**Accuracy**
Non-word	2921	153	157	0.948	0.49	1	0.95
Real-word	2897	89	245	0.92	0.27	1	0.929
Average	**-**	**-**	**-**	**0.937**	**0.38**	**1**	**0.939**

5.5 Discussion

CBSCAO has done a good job of detecting, correcting and suggesting alternatives to the non-word spelling errors and real word errors. According to the experimental results, the test prediction of the system is 2921 and 3142 as corrected words for both errors and 153 as the misspelling non words and 89 real word error, accordingly the system achieved 93.9% performance accuracy for correction of both non-word and real word error. The result indicates that the model effectively and efficiently dictate and correct both non-word errors and real word error. The analysis coverage corrected words of the model were determined by recall that has the value of 93.7% and valid words not recognize by the model is score 49%, correct word that the spellchecker take as the misspelled words because of the small size of the dictionary. As a result, the model need additional data to increase the value scored by the recall and decrease error precision. Whereas, all misspelled words correctly flagged by the model score 100% according to the experiment presented. Also the study conducted by Attia et al., (2012) state that, the performance of spellcheck by construction dictionary for non-word error and bigram model for real word error score 82.86 % accuracy.

On the other hand, the performance of spell checker for real word error is 92.9% due to the size of corpus in training text. This result shows that the corpus size used for training the model is not cover all the words in the testing data. Therefore, the sequence of words in the sentences is depends on the bigram generated and needs more corpus to increase the efficiency and effectiveness of checking the spelling errors. The model coverage of corrected Afan Oromo words was determined by recall that has a value 92.7% and 26% error precision from the sample test set used to learn the model. This shows that based on the sample test used for testing the prototype system needs improvement because the real word errors were marked based on preceding of words in the sentences. The misspelled words that exist in the tested data were 100% checked and the system was correctly detect all errors not included in the bigram lists.

The result founded in current study support that, using bigram probabilistic approach is afford optimistic result to correct the real word misspelling. The study conducted by (Ghotit ltd., 2011) also support this result which is show around 93% of real word were corrected by using bigram probabilistic method. Moreover, bigram probabilistic approach is promising result as compared with restoring lexical Cohesion method, which focus on semantic relatedness to correct the real word error. As show by Hirst and Budanitsky (2003) the recall score 50% and precision 20% to correct misspelled. On other hand to correct non-word error Levenshtein provides expectant results as shown in Table 5.3. The study conducted by (Yitayal, 2016) shows that, to correct non-word error by using Levenshtein score 95.62 accuracy.

According to the evaluation of the spellchecker presented in Table 5.2 context based spellchecker for Afan Oromo is optimistic system in order to solve the problem of the misspelling words, the system score 93.9% accuracy to correct both non-word and real word error.

CHAPTER SIX

CONCLUSION AND RECOMMENDATION

This thesis has presented the development and experimental results of Context based for Afan Oromo. In the previous chapters of this document, the theoretical basis of spell checking have been reviewed and as required part of building our model, the preparation of the Afan Oromo text corpus is showed. Moreover, the construction of dictionary and bigram model from the corpus also shown. Our main achievements have been the experimental results of our work, as shown in chapter 5. This chapter presents our conclusive remarks and recommendations for future work.

6.1 Conclusion

The ultimate goal of research on Natural Language Processing is to parse and understand language, which is not fully achieved yet. For this reason, research in NLP has focused on intermediate tasks that make sense of some of the structure inherent in a language without requiring complete understanding. One such task is spelling correction.

Spell checking and correcting have become a part of everyday life for today's generation. Whether it be working with text editing tools, such as MS Word, or typing text messages on one's cellular/mobile phone, spell checking and correcting are an inevitable part of the process. In this study, Context based spell checker for Afan Oromo has been designed, developed and tested on both non-word and real-world errors. Two approaches of spelling checking are used in this research: dictionary lookup approach and N-gram approach. Dictionary lookup approach was implemented to detect and correct non-words error and it requires dictionary in order to cross check each user words. On the other hand, N-gram approach is designed and implemented for real word errors. The N-gram Statistical approach requires a good quality training data set. And also help us to check real word errors of a given words. Both approaches are capable to handle both non-word and real word errors from the Afan Oromo writing system.

As the performance of the context based spell checker for Afan Oromo writing shows, the system registered 93.9% accuracy. According to the result obtained the system was assured to detect and correct the misspelling words. However, regarding on the real-word the performance of the system indicates 92.9% accuracy and 92.7% recall, it needs to resolve enhance the performance of the context based spell checker.

Afan Oromo Spell checking model, did not recognize compound words as the single words. But, it detect as the different word by split each words. Moreover, abbreviations with capital letters and word with number are did not taken as correct word as the Microsoft Word (MS) does. In MS every capital letters abbreviation and word with number are accepted as corrected words. For instance in MS; ‘frm1’ is taken as correct word, even if the word did not found in English dictionary. However for CBSAO model, ‘kottu1’ is taken as misspelled words, because in this model every numbers are removed before checked.

6.2 Recommendation

The results of this research have resolve the problem of misspelling when the user write Afan Oromo words. However, this work could mainly benefit from integration of different areas of researches. This section lists a brief list of areas of improvements for this research work.

Based on the finding of this study the following research direction are suggested as future works:

- From the experiments that have been conducted, in this work absence of supervised approach degrades the performance of the real word spell checking, there is a need to use supervised approaches to maximize accuracy of detecting and correcting spelling errors for real-word error. Supervised approaches should be applied using annotated and tagged texts by integrating dictionary based with N-gram analysis.
- The gathering of large corpus would be a valuable contribution to enhance the performance of the spelling checker to provide best candidate suggestions for spelling errors especially for real word errors. Thus, preparing adequate and better size corpus must be one task in the future and having a standard dictionary with maximum word size is very important to increase the accuracy of spell checker.

- Another possibility for improving performance could perhaps be using word probability information greater than bigram (trigram). To gate correct real word error and high possible suggestion.
- The distance metrics used in this model support only the correction of single error (deletion, insertion, substitution). To handle multiple error the distance metrics should be modified and revised to gate best suggestions for misspelled.
- Furthermore, CBSAO focus only on the tokenization and normalization preprocess procedure. However, we need additional preprocess (stemming), in order to gate words by its root to save space (size) when we create dictionary.
- Mixed approach (rule based with N-gram) should be used to correct especially real word error happened with noun, place and gender.

REFERENCES

Achenkunju, A., and Bhuma, V. (2014). An Efficient Refromulated Model for Transformation of String. *International Journal of Engineering Research and Applications (IJERA)*, 2248-9622.

Ahmad, F., and Kondrak, G. (2005, October). Learning a spelling error model from search query logs. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, 955-962.

Ahmed, F., Ernesto, W., De L., and Andreas, N. (2009). Revised N-gram based Automatic Spelling Correction Tool to Improve Retrieval Effectiveness. Technical University of Berlin.

Al-bakry, P. A. M. (2014). Enhanced Levenshtein Edit Distance Method functioning as a String-to-String Similarity Measure, 48–54.

Amber W.o., Graeme H., and Alexander B. (2008). Real-word spelling correction with trigrams. Atkinson, K. (2004). GNU Aspell Spell Checker. Retrieved from http://aspell.net/ (accessed August 11, 2018).

Attia, M., Pecina, P., Samih, Y., Shaalan, K., and Genabith, J. (2012). Improved spelling error detection and correction for arabic. *Proceedings of COLING 2012: Posters*, 103-112.

Baik, H. S., Jeong, H. S., and Abraham, D. M. (2006). Estimating transition probabilities in Markov chain-based deterioration models for management of wastewater systems. *Journal of water resources planning and management*, *132*(1), 15-24.

Bassil, Y. (2012). Parallel spell-checking algorithm based on Yahoo! N-grams dataset. *arXiv preprint arXiv:1204.0184*.

Blunsom, P. (2004). Hidden Markov Models: pcbl@cs.mu.oz.au, accessed on August 19, 2018.

Bassil, Y., and Semaan, P. (2012). ASR Context-Sensitive Error Correction Based on Microsoft N-Gram Dataset, *4*(1), 34–42.

Borah, B. (2017). For Query-Based Text Summarization, 17–30. Retrieved from https://doi.org/10.1007/978-3-319-62407-5(accessed August 8, 2018).

Boswell, D. (2005). UCSD Research Exam (Summer 2004) . Speling Koreckslon : A Survey of Techniques from Past to Present, 1–22.

Brill, E., and Moore, R. C. (2000, October). An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics* (pp. 286-293). Association for Computational Linguistics.

Chakraborty R.C. (2010). Artificial Intelligence: Natural Language Processing.

Christen, P. (2012). A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication, (5).

Cohen, J. D. (2001). *U.S. Patent No. 6,169,969*. Washington, DC: U.S. Patent and Trademark Office.

Cucerzan, S., and Brill, E. (2002). Spelling correction as an iterative process that exploits the collective knowledge of web users.

Daiber, J., Jakob, M., Hokamp, C., and Mendes, P. N. (2013). Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems* (pp. 121-124).

Deksne, D., and Skadiņš, R. (2011). CFG Based Grammar Checker for Latvian.

Deorowicz, S., and Ciura, M. G. (2005). Correcting spelling errors by modelling their causes. *International journal of applied mathematics and computer science*, 15, 275-285.

Dickinson, B. (2013). Developing A Real-Word Spelling Corrector.

Evermann, G., and Woodland, P. C. (2000). Posterior probability decoding, confidence estimation and system combination. In *Proc. Speech Transcription Workshop*, 27, 78-81.

Flor, M., and Futagi, Y. (2012). On using context for automatic correction of non-word misspellings in student essays, 105–115.

Gersten, R. M., Fuchs, L., Coyne, M. D., and Greenwood, C. R. (2005). Quality Indicators for Group Experimental and Quasi-Experimental Research in Special Education.

Ghotit ltd. (2011). Ghotit Dyslexia Contextual Spell Checker.Retrieved from http://www.ghotit.com/home.shtml(accessed June 20, 2018).

Girma, T. (2014). Human Language Technologies and Affan Oromo Issn : 2278-6252 I . Introduction, *3*(5), 1–13.

Grinstead, C. M., and Snell, J. L. (2012). Introduction to probability. American Mathematical Soc.

Haldar R. and Debajyoti M.(2011). Levensshtein Distance Technique in Dictionary Lookup Methods: An Improved Approach.

Han, B., and Baldwin, T. (2011). Lexical Normalisation of Short Text Messages : Makn Sens a # twitter, 368–378.

Hilas, C. S., and Mastorocostas, P. A. (2008). Knowledge-Based Systems An application of supervised and unsupervised learning approaches to telecommunications fraud detection q, *21*, 721–726. Retrieived from https://doi.org/10.1016/j.knosys.2008.03.026(accessed July 11, 2018).

Hindle, A., Barr, E. T., Su, Z., Gabel, M., and Devanbu, P. (2012, June). On the naturalness of software. In *Software Engineering (ICSE), 2012 34th International Conference on*, 837-847.

Hirst, G., and Budanitsky, A. (2003). Correcting real-word spelling errors by restoring lexical cohesion, 1–44.

Inkpen, D. (2009). Real-Word Spelling Correction using Google Web 1T 3-grams, (August), 1241–1249.

Jurafsky, D., and Martin, J. H. (2009). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. *Prentice Hall series in artificial intelligence*, 1-1024.

Jurish, B., and Würzner, K. M. (2013). Word and Sentence Tokenization with Hidden Markov Models. *JLCL*, *28*(2), 61-83.

Kim, Y., and Rush, A. M. (2016). Character-Aware Neural Language Models, 2741–2749.

Kirthi J., Neeju N.J., and Nithiya P.(2011). Automatic spell correction of User query with semantic Information Retrieval and Ranking of search Result using WordNetApproach.

Koehn, P., Birch, A., Callison-burch, C., Federico, M., Bertoldi, N., Cowan, B., … Herbst, E. (2007). Moses : Open Source Toolkit for Statistical Machine Translation, (June), 177–180.

Liang, H. L. (2008). *Spell checkers and correctors: A unified treatment* (Doctoral dissertation, University of Pretoria).

Li, Y., Duan, H., and Zhai, C. (2012). A generalized hidden markov model with discriminative training for query spelling correction. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pp. 611-620.

Krogh, A., Larsson, B., Von Heijne, G., and Sonnhammer, E. L. (2001). Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes.*Journal of molecular biology, 305*(3), 567-580.

Mamo, G., and Meshesha, M. (2009). Part-of-Speech Tagging for Afan Oromo Language. *Proceeding International Journal of Advanced Computer Science and applications, special issue on Artificial Intelligence*, *1*(3), 1-5.

Manning R. and Schutze.(2008). An introduction to Information Retrieval.Cambridge University Press.

Markou, M., and Singh, S. (2003). Novelty detection : a review — part 1 : statistical approaches, *83*, 2481–2497. Retrieved from https://doi.org/10.1016/j.sigpro.2003.07.018(accessed July 11, 2018).

Màrquez, L., Escudero, G., Martínez, D., and Rigau, G. (2007). Supervised corpus-based methods for WSD. In *Word Sense Disambiguation* (pp. 167-216). Springer, Dordrecht.

Mihov S., Svetla K.(2004). Precise and Efficient Text Correction Models.Bulgarian Academy of Sciences.

Mihov, S., Schulz, K. U., Ringlstetter, C., Dojchinova, V., Nakova, V., Kalpakchieva, K., and Gercke, C. (2005, August). A corpus for comparative evaluation of OCR software and postcorrection techniques. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, 162-166.

Misha R., and Navjot K. (2013).”A Survey of Spelling Error Detection and Correction Techniques.” International Journal of Computer Trends and Technology, 3, 372-374.

Mishra, R., and Kaur, N. (2013). A Survey of Spelling Error Detection and Correction Techniques. *International Journal of Computer Trends and Technology, 4(3), 372-374.*

Mishra, R., Kaur, N., and Technique, A. D. L. (2013). A Survey of Spelling Error Detection and Correction Techniques, *4*, 372–374.

Momtazi S. (2012). Natural Language processing: introduction to Language Technology. University of Potsdam.

Nadkarni, P. M., Ohno-machado, L., and Chapman, W. W. (2011). Natural language processing : an introduction. Retrieved from https://doi.org/10.1136/amiajnl-2011-000464(accessed July 1, 2018).

Nagi, I. (2006). Caassefama Afan oromo. *Kuraz International, Addis Ababa*.

Navigli, R. (2009). Word Sense Disambiguation : A Survey, *41*(2). Retrieved from https://doi.org/10.1145/1459352.1459355(accessed June 19, 2018).

Neha G. (2006).Pratisthamathur Spell checking techniques in NLP: a survey, Department of computer science, Banasthalividyapith, India.

Nejja, M., and Yousfi, A. (2015). The context in automatic spell correction. *Procedia - Procedia Computer Science*, *73*(Awict), 109–114. Retrieved from https://doi.org/10.1016/j.procs.2015.12.055(accessed June 19, 2018).

Olson, D. L., and Delen, D. (2008). *Advanced data mining techniques*. Springer Science and Business Media.

Pagh, R., and Friche, F. (2004). Cuckoo hashing, *51*, 122–144. Retrieved from https://doi.org/10.1016/j.jalgor.2003.12.002(accessed July 12, 2018).

Raaijmakers S.(2013). “A Deep Graphic Model for Spelling Correction.”Proceedings of the 25th Benelux Conference on Artificial Intelligence.

Ratnasari, C. I., Kusumadewi, S., and Rosita, L. (2017). A Non-Word Error Spell Checker for Patient Complaints in Bahasa Indonesia.

Republic, C., and Science, I. (2015). Arabic spelling error detection and correction , (514). Retrieved from https://doi.org/10.1017/S1351324915000030(accessed July 1, 2018).

Ringlstetter, C., Hadersbeck, M., Schulz, K. U., and Mihov, S. (2007). Text Correction Using Domain Dependent Bigram Models from Web Crawls, 47–54.

Seo, H., Lee, J., Kim, S., Lee, K., Kang, S., and Lee, G. G. (2012, July). A meta learning approach to grammatical error correction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2* (pp. 328-332). Association for Computational Linguistics.

Seretan, V., Nerima, L., and Wehrli, E. (2004). A tool for multi-word collocation extraction and visualization in multilingual corpora.

Sidoti, C., Metsky, H., and Somogyi, J. (2008). Using Google to Create a More Accurate and Easily-Extensible Spelling Corrector.

Su, Z., Ahn, B. R., Eom, K. Y., Kang, M. K., Kim, J. P., and Kim, M. K. (2008, June). Plagiarism detection using the Levenshtein distance and Smith-Waterman algorithm. In Innovative Computing Information and Control, 2008. ICICIC'08. 3rd International Conference on (pp. 569-569).

Sundby, D. (2009). Spelling correction using N-grams. Retrieved from http://www.daviddlewis.com/resources/testcollections/re uters21578/.

Tavast, A., Muischnek, K., and Koit, M. (2012). *Human Language Technologies–The Baltic Perspective: Proceedings of the Fifth International Conference Baltic HLT 2012* (Vol. 247). IOS Press.

Tesema W., and Tamirat, D. (2017).Investigating Afan Oromo Language Structure and Developing Effective File Editing Tool as Plug-in into Ms Word to Support Text Entry and Input Methods. *American Journal of Computer Science and Engineering Survey.*

Tesema, W., Tesfaye, D., and Kibebew, T. (2016). Towards The Sense Disambiguation of Afan Oromo Words Using Hybrid Approach (Unsupervised Machine Learning and Rule Based).*Ethiopian Journal Of Education And Sciences*, *12*(1), 61-77.

Tesfaye, D.(2011). A rule-based Afan Oromo Grammar Checker. *(IJACSA) International Journal of Advanced Computer Science and Applications, 2(8).*

Tesfaye, D., and Abebe, E. (2010). Designing a Rule Based Stemmer for Afan Oromo Text. *International journal of computational linguistics (IJCL), 1(2).*

Tijhuis, L. J. (2014). Context-Based Spelling Correction for the Dutch Language: Applied on spelling errors extracted from the Dutch Wikipedia revision history.

Tilahun, G. (2001). Forms of Subject and Object in Afan Oromo. *Journal of Oromo*, *8*(1and2).
Treiman, R., and Kessler, B. (2005). writing System and Spelling Development.

Tune, K. K., Varma, V., and Pingali, P. (2007). Evaluation of Oromo-English Cross-Language Information Retrieval.*Language Technologies Research Centre IIIT, Hyderabad India*.

Tune, K. K., Varma, V., and Pingali, P. (2008). Evaluation Of Oromo-English Cross-Language Information Retrieval Evaluation Of Oromo-English Cross-Language Information Retrieval.

Verberne, S. (2002). Context-sensitive spell checking based on word trigram probabilities. *Unpublished master's thesis, University of Nijmegen*.

Vincze, V., Szarvas, G., Farkas, R., Móra, G., and Csirik, J. (2008). The BioScope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC bioinformatics*,*9*(11).

Wilde, O. (2017). Spelling Correction and the Noisy Channel.

Xie, W., Huang, P., Zhang, X., Hong, K., Huang, Q., and Huang, L. (2015). Chinese Spelling Check System Based on N-gram Model, 128–136.

Yitayal, N. (2016). Context based spell checker for Amharic. *Unpublished master's thesis, University of Jimma*.

Zue, V., Seneff, S., Glass, J. R., Polifroni, J., Pao, C., Hazen, T. J., and Hetherington, L. (2000). JUPlTER: a telephone-based conversational interface for weather information. *IEEE Transactions on speech and audio processing*, *8*(1), 85-96.

APPENDIX

I Sample code of context based spell check for Afan Oromo

```
package Afan_oromoSC;

import static Afan_oromo.ContextBasedCheck.concat;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.io.*;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Collection;

import java.util.Collections;

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

import java.util.StringTokenizer;

import java.util.TreeMap;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import javax.swing.*;

import javax.swing.text.DefaultHighlighter;

import javax.swing.text.Document;

import javax.swing.text.Highlighter;
import javax.swing.text.*;
//import javax.swing.text.
public class ContextBasedAOCheck {
    String ErrorWord="";

    JTextPane ta =new JTextPane();
      //String UserWord=ta.getText();
   String Userword=ta.getText();
        JButton check=new  JButton("Check");
            private HashMap<String, Integer> dictionaryData;
    public ContextBasedAOCheck()
        {
        JFrame frame= new JFrame();
          JPanel p2=new JPanel();
        JButton clr=new JButton("Clear");
        JButton br=new JButton("Browse");
        frame.setLayout(new BorderLayout());
         JMenuBar mb=new JMenuBar();
         JMenu file=new JMenu("File");
          JMenu open=new JMenu("Open");
           JMenu help=new JMenu("Help");
            JMenu exit=new
JMenu("Exit");mb.add(file);mb.add(open);mb.add(help);mb.add(exit);p2.add(chec
k);p2.add(clr);p2.add(br); frame.setJMenuBar(mb);
            frame.add(ta, BorderLayout.CENTER);
            frame.add(p2, BorderLayout.SOUTH);

           frame.setSize(600,500);
           frame.setTitle("Context Based Spellchecker for Afana Oromo");
           frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
           frame.setVisible(true);}
      public void reading() throws FileNotFoundException, IOException{
           check.addActionListener(new ActionListener() {
               public void actionPerformed(ActionEvent e) {
                   try {
                      dictionaryData = new HashMap<String, Integer>();
            BufferedReader br;
            String sCurrentLine;
                  br = new BufferedReader(new
FileReader("src/Afan_oromoSC/AO_Corpus.txt"));
           sCurrentLine=br.readLine();
       // int Bigram=2;
        int userBigram=1;
         // System.out.println(sCurrentLine);
        for (String useString : ngrams(userBigram, ta.getText()))
           //System.out.println(sCurrentLine);
              if(sCurrentLine.contains(useString)==false)
          //System.out.println("Misspelling " + useString);
                {  frame.setSize(600,500);
           frame.setTitle("Context Based Afaan_Oromoo Spell Checker");
           frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
           frame.setVisible(true); }
      public double totalWordsInDocument() {
            Scanner sc = readFile();
            double count = 0;
        while (sc.hasNext()) {
            String [] s = sc.next().split("d*[.@:=#-]");
            for (int i = 0; i < s.length; i++) {
                if (!s[i].isEmpty()){
                    count++; }  }   } return count;}
      public Map<String, Double> unigramCount() {
        Scanner input = readFile();
        Map<String, Double> wordCounts = new TreeMap<String, Double>();
        while (input.hasNext()) {
            String next = input.next().toLowerCase();
            if (!wordCounts.containsKey(next)) {
                wordCounts.put(next, (double) 1);
            } else { wordCounts.put(next, wordCounts.get(next) + 1); } }
        return wordCounts;
      }
      public double bigramCount(String word1, String word2) {
            String regex = word1.toLowerCase() + "\\s+" +
word2.toLowerCase();
            Scanner input = readFile();
            double count = 0;
            while(input.hasNextLine()) {
                  String line = input.nextLine().toLowerCase();
                  Pattern pattern = Pattern.compile(regex);
                  Matcher  matcher = pattern.matcher(line);
                  while(matcher.find()) {
                        count++     }}return count;}
      public double [][] getBigramCountTable(String sentence) {
            sentence = sentence.replaceAll("\\.", "").toLowerCase();
            String [] sentenceTokens = sentence.split("\\s+");
            double [][] bigramTable = new
double[sentenceTokens.length][sentenceTokens.length];
            for(int i = 0; i < sentenceTokens.length; i++) {
                  for(int j = 0; j < sentenceTokens.length; j++) {
                        bigramTable[i][j] = bigramCount(sentenceTokens[i],
sentenceTokens[j]);
                                System.out.println(bigramTable[i][j] );}}
      return bigramTable;}
      public double [][] getBigramProbabilityTable(String sentence) {
            sentence = sentence.replaceAll("\\.", "").toLowerCase();
            String [] sentenceTokens = sentence.split("\\s+");
            double [][] bigramTable = new
double[sentenceTokens.length][sentenceTokens.length];

            Map<String, Double> unigramCountMap = unigramCount();

            for(int i = 0; i < sentenceTokens.length; i++) {

                  for(int j = 0; j < sentenceTokens.length; j++) {

                        if(unigramCountMap.containsKey(sentenceTokens[i])) {

                              bigramTable[i][j] =
bigramCount(sentenceTokens[i], sentenceTokens[j]) /
unigramCountMap.get(sentenceTokens[i]);

                        }else {bigramTable[i][j] = 0; }}}

      return bigramTable;}

      public double [][] getAddOneSmoothingBigramProbabilityTable(String
sentence) {

            sentence = sentence.replaceAll("\\.", "").toLowerCase();

            String [] sentenceTokens = sentence.split("\\s+");

            double [][] bigramTable = new
double[sentenceTokens.length][sentenceTokens.length];

            Map<String, Double> unigramCountMap = unigramCount();

            for(int i = 0; i < sentenceTokens.length; i++) {

                  for(int j = 0; j < sentenceTokens.length; j++) {

                        if(unigramCountMap.containsKey(sentenceTokens[i])) {

                              bigramTable[i][j] = (
bigramCount(sentenceTokens[i], sentenceTokens[j]) + 1 ) /
(unigramCountMap.get(sentenceTokens[i]) + unigramCountMap.size());

                        }else {

                              bigramTable[i][j] = (
bigramCount(sentenceTokens[i], sentenceTokens[j]) + 1 ) /
unigramCountMap.size(); }}}

            return bigramTable;}

        public static void main(String args[])

        {

        new AfanO_Spell();

        }
```

II Sample Trained Data to Evaluated Model.

Sirna gadaa amma kiyyoo kolonii jala hin seeninitti Oromonni sirna ittiin bulan kan mataa isaanii qabu turan. Maqaan sirna kanaas Gadaa jedhama. Sirni kuniis bal’aa ture. Dhimma jireenya ummata Oromo kallattii hundaan kan ilaalu sirna siyaasa, aada diinagdeefi amantiiti. Sirni Gadaa sirnaa seera Oromonni ittiin walbulchu, kan duulee roorroo ofirraa ittisu, kan dinagdee isaa ittiin tikfatuufi dagaagfatu, akkaata inni itti waliin jiraatuufi kan hawwiin dhala Oromo cufa ittiin guutu ture. Oromota sirna akkasii jalatti qindeessuudhaaf yeroo dheeraa fudhate. Oromota gosatti hiraman walitti fidanii sirna tokko jalatti walitti qabuun kufaatiifi ka’uumsaa yeroo dheeraa gaafate. Haata’u malee, Gara walakkaa jaarraa kudha shanaffaa isa lammaffaa keessa sirni gutuun argamuu danda’e. Akkaata himamsa aadaa Boorana kibbaa keessatti yeroon itti Gadaan dhaabbate kanumaan walkipha. Akka mangoddoonni Booranaa himanitti Gadaa ijaaruuf yaalii dheeraan eega godhame booda, Gadaan yeroo dheeraaf bifa hojechuu danda’un dhaabbate. Sirna Gadaa keessatti wanni hundi gadaan walqabata. Lakkoofsi yeroo, aadaan, amantiin, jireenyi hawaasaa fi ittisi biyyaa hundi hidhata Gadaa qabu. Bara Gadaan sirna gutuu ture sana akka tahutti galmeessuun hindanda’amne. Hammi galmeefame baay’ee yaraadha. Haata’u malee, kan galemeeffame keessaa irra guddaan kan barootta dhihooti. Waa’een sirna Gadaa hammi beekamu baay’ee yaraadha. Irraa caalaattii odeeffannoon argaman oduu Afaniin kan daddabraniidha. Isaaniis bakka bakkati garagar tahu. Sunillee Gadaan Caffe Caffetti deebi’uu haa agarsiisu malee, tokkummaa sirna kanaaf raga kan tahu haftee sirna kan Gadaa Booranaa har’aalee haga taheefuu hojjataa jiruudha. Kanaaf waa’ee sirna Gadaarratti hangi hubatame xiqqaadha. Waliigalatti sirna kana caalatti hubachuudhaaf qormaata cimaa sirna Gadaa kan mul’isan armaan gaditti ilaalla. Hawaasa keessatti dhalli namaa kophaa hinjiraatu. Kophaas hindalagu. Namoota biraa wajji jiraata, hojjataas. Jireenyi dhala namaa walitti hidhaataadha. Hawaasa Oromo bara Gadaa keessaa yoo fudhannee akkaataa jiruufii jireenya Oromo, akkaataa qoodama dalagaa, hariiroo ummanni Oromo hawaasa keessatti waliinqabau, sirna Gadaa keessatti qaama tokko. Akkaataan polotikaa sirnichaas uunkaa jireenyya hawaasa Oromotti hidhamee jira. Uunkaa jireenya hawaasa sirna Gadaa keessaa hubachuuf waan kanaa gadii hubachuunbarbaachisaadha. Sirna gadaa keessatti odeffannoon argaman tursiisuuf haala danqaa tu turee. Sirnoota darban kanaan dura keessattii odeefaannoo wa'ee sirnagadaa tursiisuuf ta'ee hojirraa olchuuf argaman muraasa turan.

Mallattoon Booqaa Birraa, ibsaan Masqalaa ayyaanni Masqalaa kan itti kabajamu dha. Dargaggeessi xomboora guuree Daamaraa Masqalaa gubee, Kormoomsaa daaraa masqalaa addatti dibatee, sirba masqalaa wal jalaa qabee sirbaa yiikkisee dhiichisu, shamarran dibbeen lafa goosan jaarsi biyyaa ija irraa hafaa, horaa bulaa jedhee naannoo ibsaa masqalaa dhaabbatee yeroo itti eebbisu dha. Kanaaf hawaasni Oromoo Waaqa waa hunda keessa isa dabarsee birraa hawwataa kanaan isa ga'e Irreeffatee galateeffata. Akka amantaafi falaasama Uummata Oromootti, Uummamni hundi gochaafi humna waaqa isaan uume mullisu jedha. Iccitiin humna Waaqa tokkichaa karaa uumamoota kanaa ibsama. Kanaaf, Irreechi bifa aadaatiin sirna galateeffannaa fi kadhaa waaqaa taasifamu waan ta'eef iddoowwan humni uumuu Waaqaa adda ta'anii itti mullatan keessaa bakka lama ; Isaaniis Irreecha Tulluu fi Irreecha Malkaa jedhamu. Iddoon kun lamaan ballinaan waan achitti kabajamaniif jecha malee akkaatuma lafa Oromoottii bakkeewwan biroon Irreeffannan itti raawwataman hin jiran jechuu miti. Irreechi Tulluu kan itti kabajamu jiini bonaa darbee yeroo bokkaan afraasaa itti roobuu qaba jedhamee eegamu keessattidha. Tibbi bonaa goginsaan beekama. Aduufi hoo'a irraan kan ka'e jiidhi lafaa gogee, margi qooree, bishaan cooreefi malkaa jiru hunda keessatti qal'atee mul'ata. Yoo gogiin kun dheerate horiis ta'ee dhala namaa balaa irra buusa. Kun akka hin taaneef giddugala ji'a Bitootessaatti hangafootni biyyaa wal waamanii akka seera isaatti naannoo lafaa bakka jiidha qaba jedhamutti yaa'un harka isaanii gara waaqaatti qabanii "Jiidha nun dhowwatin, waqtiileen toora isaanii eeganii haa naanna'an, qilleensa roobaa nuuf kenni, araar" jedhanii kadhaa waaqaa taasifatu. Tulluutti ba'anii waaqa tokkicha sana kadhatu. Sirni kadhaa kun Aadaafi falaasama amantaa kuushota duriin kan wal qabatedha jedhama. Irreechi malkaa kan adeemsifamu, wayita jiini gannaa darbuuf jedhu Baatii Fulbaanaa gara walakkaa keessatti dha. Jiini Fulbaanaa jalqaba Waqtii Birraa waan ta'eef haalli qilleensa gannaa jijjiirama itti agarsiisu eegaludha. Kunis lafti xuuxxataafi sochiif mijataa, qilleensi duumessaafi huurrii ture ifaafi ho'a aduu Birraan bakka bu'aa deema. Dacheen daraaraa biqilootaan uwwifamtee bareeduu eegalti. Kanaaf, guyyaan irreechaa kun "Ayyaana Birraa" jedhamee beekama. Guyyaan Irreechaa kun bakka tokko tokkotti ammoo guyyaa "Xaddacha saaquus" ni jedhama. Kunis hoggaa dukkanniifi bacaqii gannaa darbee wal arguun itti danda'ame jedhamee amanama. Kanuma irraan kan ka'e Irreechi guyyaa galata Waaqaati jedhama. Galateeffannaa kana irratti namni hundi akkuma umurii isaatti uffata aadaa uffatee, dargaggoonni bareedanii, haawwanis, Wandaboo kuula bareedaan hojjetame irratti balees ta'ee kaabbaa, calleefi dooqaan faayame uffatanii coqorsa jiidhaa, keelloo, hadaa, daraaraa qabatanii malkaa gamaa gamanaan wal ga'anii Waaqa faarfatu, Maaramii galateeffatu, "Maareewoo, Maareewoo..."jedhu. Haala kanaan, gara malkaatti yaa'u. Abbootiin Gadaa, kallacha, bokkuu, marga, alangaa, jirma ykn ulee aangoo Gadaa ibsu qabatanii dura bu'anii tooraan seeraafi hayyataan sochii gara malkaattii ta'u sana geggeessu. Haalli uffannaa nama hundumaa bifa kabaja Waaqaa ibsuun ta'a. Haawwaniis ta'e shamarran mogalee gaditti uuffata dheeraa uffatu. Kunis " Yaa waaqi nun soqolin, gaaddidduu kee nun dhowwatin " jechuu dha jedhama. Malkaan Irreechi akka Oromiyaatti bal'inaan itti kabajamu Magaalaa Bishooftuu bakka Hora Har-sadii jedhamuttidha. Magaalaan Bishooftuu, Hora adda addaan badhaatee bifa bareedaafi uumama taatee akka kan godhe bishaan hedduu waan qabduufidha. Kanumaaf naannoo bishaanoftuu waan jedhamu irraa ka'uudhaan magaalaan kun Bishooftuu jedhamte jedhu Jaarsoliin Biyyaa. Horri harsadiis Horawwan magaalaatti marsanii jiran keessaa Malkaa Irreechaa beekamaafi bara dheeraan duraa kaasee itti Irreeffatamaa ture ta'uu dubbatama. Harsadiin abbaa malkaa hora sanaa gosa Liiban keessaa kan ta'e, harri kun maqaa isaarraa ka'uun kan moggaafame ta'uu jaarsoliin ykn maanguddoonni Galaan ni dubbatu. Oromoon golee Oromiyaa jiru hunduma irraa guyyaa Irreechaa kanatti seera guutee uffatee jiidha qabachuun Waaqa galateeffanna, ni kadhanna, Uumamaan uumaa isa ta'e waaqa raajeffanna! jechuun Hora Harsadii magaalaa Bishooftuutti argamu kana deemee galata galchata. Hawaasni waggaa waggaan achitti argamu bara baraan dabalaa deemaa jira. Yeroo ammaa kana uummata Miliyoona Afurii ol ta'utu achitti argama jedhama.