ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

A TWO STEP APPROACH FOR TIGRIGNA TEXT CATEGORIZATION

BY

GEBREHIWOT ASSEFA BERHE

JUNE 2011

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

A TWO STEP APPROACH FOR TIGRIGNA TEXT CATEGORIZATION

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By

GEBREHIWOT ASSEFA BERHE

JUNE 2011

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

A TWO STEP APPROACH FOR TIGRIGNA TEXT CATEGORIZATION

By

GEBREHIWOT ASSEFA BERHE

Name and signature of Members of the Examining Board

Name	**Title**	**Signature**	**Date**
_____	Chairperson	_____	_____
_____	Advisor(s),	_____	_____
_____	Advisor(s),	_____	_____
_____	Examiner,	_____	_____

Declaration

I declare that the thesis is my original work and has not been presented for a degree in any other university.

_____

Date

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

_____

Advisor

ACKNOWLEDGEMENT

First, I am deeply grateful for my advisor Dr. Million Meshesha for his constructive comments, suggestions and full devotions from beginning to end of my thesis work.

My special gratitude goes also to Mekelle University for its printing service, material supports, and the opportunity that helps me to engage in this program.

I owe my deepest heartfelt appreciation to my friend Haftom Gebreegzabher for his material support and encouragement in conducting this study. You are always in my heart where you are, Haftish.

It is a pleasure to thank Education Bureau of Tigray region, Capacity building of Tigray region, Demtsi Woyne and Mekalh Tigray for letting me use their Tigrigna text documents for my thesis work.

I would like to thank ato Yohanes Birhane for labeling the test data set of the present study.

I would like also to thank Addis Ababa University for its financial support. It would not come to this end without its support. During my stay at AAU, I got a lot of experiences that could be a turning point of my career.

Last but not least, I would also like to thank to my family, my friends and my classmates for their love and support to conduct this study.

TABLE OF CONTENTS

CONTENTS PAGES

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ASCII American Standard Code for Information Interchange

DF Document Frequency

ENA Ethiopian News Agency

IDF Inverse Document Frequency

IR Information Retrieval

ML Machine Learning

SERA System for Ethiopic representation in ASCII

SVM Support Vector Machine

TC Text Categorization

TF Term Frequency

TF*IDF Term Frequency by Inverse Document Frequency

ABSTRACT

Tigrigna language is a Semitic language spoken by the Tigray people in Northern Ethiopia and Eritrea which has more than six million speakers worldwide. There are large collections of Tigrigna document available in web, in addition to hard copy document in library, and documentation centers. Even though the amount of the document increase, there are challenging tasks to identify the relevant documents related to a specific topic. So, a text categorization mechanism is required for finding, filtering and managing the rapid growth of online information.

Several researches have been done on text categorization, especially news text classification with the help of different machine learning approaches; and good results were found. However, with the growth of text corpus the text classification using a predefined category is an extremely costly and time-consuming activity. The need for classifiers that can learn from unlabeled data is required. Hence, this study attempts to design a two step Tigrigna text categorization system. First, clustering is used to find natural grouping of the unlabeled Tigrigna text documents. Here, repeated bisection and direct k-means clustering algorithms are used to obtain documents of natural group of the Tigrigna data set. The repeated bisection clustering algorithm outperforms the direct k-means clustering algorithms. So the repeated bisection clustering algorithm results are selected for classification task.

For the classification task decision tree and support vector machine techniques are used in the present study. The SMO support vector machine classifier performs better than J48 decision tree classifier. SMO registers 82.4% correct classification. However, there are challenges in designing a Tigrigna text categorization system; worth to mention are the mismatch encountered between clustering and classification algorithms, and the Tigrigna language ambiguity which demands further research to apply ontology-based hierarchical text categorization.

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

The globalization era provides a growing amount of information and data coming from different sources. As a result, it becomes more and more difficult for target users to select contents they desire. This has a negative effect for searching the relevant documents from the entire collection. Supporting the target users to access and organize the enormous and widespread amount of information is becoming a primary issue. As a result, several online services have been proposed to find and organize valuable information needed by the target users (Addis, 2010). However, these services are not capable to fully address the users' interest. So, a mechanism is required for finding, filtering and managing the rapid growth of online information. This mechanism is called text categorization (Maron and Kuhns, 1960). Text categorization (TC) can be defined as the task of determining and assigning topical labels to content (Addis, 2010). It is also defined by Berger (2004) as the task of automatically sorting a set of documents into categories from a predefined set. The categories can be pre-defined or automatically identified. The text categorization task based on the predefined categories is called text classification. While the text categorization task based on automatically identified categories is called text clustering (Baker and Kachites, 1998).

Until the late 1980s the most popular approach to TC in the operational community was a knowledge engineering (KE) which manually defines a set of rules encoding expert knowledge on how to classify documents under the given categories. However, from the early 1990 a machine learning (ML) approach has become the major research area. Machine learning has considered on a general inductive process that automatically builds an automatic text classifier by learning from predefined set of documents (Sebastiani, 2002).

Currently, the categorization task falls at the crossroads of information retrieval (IR) and machine learning (ML). IR researchers believe that it is the user who can say whether a

given item of information is relevant to a query issued to a Web search engine, or to a private folder in which documents should be filed according to their contents. Wherever there are predefined classes, documents manually classified by the user are often available. As a result, the predefined data can be used for automatically learning the meaning that the user assigned attributes to the classes. However, reaching levels of classification accuracy that would be impossible if this data were unavailable. In the last few years, more and more ML researchers adopt TC as one of their benchmark applications of choice which are being imported into TC with minimal delay from their original invention (Addis, 2010). In ML approaches, the task that deals with classification is called supervised learning, whereas the task that deals with clustering is called unsupervised learning. In addition, application developers interest mainly due to the enormously increased need to handle larger and larger quantities of documents. This need is emphasized by increased connectivity and availability of document corpus of all types at all levels in the information chain.

Most of the time, the categories are just symbolic labels and their meaning is usually available. However, it is often the case that metadata (such as publication date, document type, and publication source) is unavailable to categorize them. In these cases, categorization must be accomplished only on the basis of knowledge extracted from the documents themselves (Sebastiani, 2000). For a given application when either external knowledge or metadata is not available, heuristic techniques of any nature may be adopted in order to leverage on these data, either in combination or in isolation from the IR and ML techniques (Sebastiani, 2005).

Text categorization can be done manually or automatically. Each of them has advantage and pitfalls to the target users. Automated text categorization is attracting more research these days because it reduces human intervention from manually organizing documents which is too expensive, and error prone (Sebastiani, 2002).

Most of automatic text categorization is done by assigning predefined categories to a given text documents. Such concept of text categorization is called text classification (John, 1998). Within this approach, set of data sets are first manually classified and

labeled with predefined categories by human experts. A learning algorithm is then applied to learn the characteristics of each category, and a classification model (classifier) is automatically built to decide the categories of future unknown data. Text classification is generally divided in to two main categories. These are flat text classification and hierarchical text classification (Addis, 2010). In flat text classification, categories are treated in isolation of each other and there is no structure defining the relationships among them. A single huge classifier is trained which categorizes each new document as belonging to one of the possible basic classes. The hierarchical text classification addresses this large categorization problem using a divide-and-conquer approach (Tikk et al., 2001). This approach utilizes the hierarchical topic structure to decompose the classification task into a set of simpler problems, one at each node in the classification tree. Text classification in hierarchical setting provides an effective solution for dealing with very large problem. By treating the problem hierarchically, it can be decomposed into several sub-problems, each of them involving a smaller number of categories. Moreover, decomposing a problem can lead to more accurate specialized classifiers.

The text classification approach provides a conceptual view of document collections and has important applications in the real world. For example, news stories are organized by subject categories (topics); academic papers are often classified by technical domains; patient reports in health-care organizations are often indexed from multiple aspects such as using taxonomies of disease categories, types of surgical procedures, insurance reimbursement codes and so on. In practice, it is not always easy to apply this approach to natural language processing tasks. Because the classification of documents to their predefined categories requires a large amount of hand labeled texts. As a result, it is quite difficult to collect the required amounts of hand labeled data when there are large amounts records in the entire corpus. On the other hand, unlabeled text collections are easily available in the real world. A text categorization approach that uses the unlabeled text collections is called document (text) clustering (McCallum et al., 2000).

Text clustering is used to assign some similar properties of text documents into automatically created groups. It is used to improve the efficiency and effectiveness of text categorization system such as time, space, and quality. The standard text clustering

algorithms can be categorized into partitioning and hierarchical clustering algorithms (McCallum et al., 2000). Partitioning clustering algorithm splits the data points into k partition where each partition represents a cluster. Whereas hierarchical clustering algorithm groups data objects to form a tree shaped structure. It can be bottom up approach which each data points are considered to be a separate cluster and clusters are merged based on a criteria or top down approach where all data points are considered as a single cluster and they are splited into number of clusters based on certain criteria. Karypis et al. (2004) has compared partitioning and hierarchical methods of text clustering on a broad variety of test data set. It concludes that k-means of the partition clustering algorithm clearly outperforms the hierarchical methods with respect to clustering quality. In addition, a variant of k-means called repeated bisection k-means is introduced and yields even better performance.

There are different procedures involved in text categorization (Addis, 2010). First, there is a need to collect the necessary data set of the given language. Second, the given documents will be processed using different techniques. Third, document indexing will be applied to organize the given documents. Fourth, the high dimensionality resulted from document indexing will be reduced using suitable algorithms. Finally, different classifier learning algorithms will be used to create models and measure their performance.

Even though more and more organizations are automating all their activities using different text categorization approaches, a number of challenges are remained for text categorization research. The first challenge is to develop a text categorization system that has high accuracy in all application areas. Effective classifiers have been developed for a certain application domains such as the classification of news stories. However, the produced systems are not satisfying the target users in other domains. This application area includes the classification of web pages, spam filtering, authorship attribution, and sentiment classification. A second challenge is the lack of labeled training documents. Labeling the training document requires a large amount of time and costs since most of the documents in the real word are available in unlabeled format. As a result, unsupervised method has been proposed by Rafael et al. (2004) that can learn from

unlabeled documents. However, the problem of building a text classifier from unlabeled training data set is still an open issue in the text categorization researches.

1.2 Statement of the Problem

Tigrigna language is a Semitic language widely spoken in the Tigray region of Ethiopia and Eritrea. Tigrigna is spoken by large immigrant communities around the world, including Sudan, Saudi Arabia, the United States, Germany, Italy, the United Kingdom, Canada and Sweden. This language has more than six million speakers worldwide. As a result, the Tigrigna documents are increasing in size from time to time (Sahle, 1998). This shows that there are large collections of Tigrigna document available in web, in addition to hard copy document in library, and documentation centers. Even though the amount of the document increase, there are challenging tasks to identify the relevant documents related to a specific topic (Leslau, 1998). First, the process of searching relevant documents from large collection becomes too expensive as it has to search every document in the entire collection. Sometimes relevant key words are used in irrelevant document and often omitted in relevant documents because the context is not clear to the target audience (Feng, 2005). In such situations, making the search easier, fast and efficient is quiet difficult for Tigrigna documents.

Second, a classification model is required for correctly predicting the class of different objects of Tigrigna language. Most existing techniques use manual classification. So, incorporating a new document dynamically in to existing document categories is not easily possible.

Third, most of the Tigrigna documents are not organized well so that it becomes quite difficult to access relevant once. This has a negative impact for searching the document in the dynamic world. So it is necessary to organize the documents such as books, newspapers, etc. properly depending on their textual contents.

It is therefore important to pay attention to text categorization which carries out tasks like extracting useful information by analyzing documents content and organizing them under specific themes (Sebastiani, 2002). Even though, assigning theme manually to the given

set of documents by looking at its context is possible with the enormous flow of documents, this manual approach becomes impractical, ineffective, inconsistent, and error prone. To overcome this problem, the automated categorization of texts into predefined categories has witnessed a booming interest in the last 10 years, due to the increased availability of documents in digital form and the ensuring need to organize them (Sebastiani, 2002).

Different researches have been conducted on text categorization for Amharic language using different techniques. These researches are conducted for Amharic text news classification (Zelalem 2001; Surafel 2003; Yohannes 2007; Worku 2009; Alemu 2010). However, there is no any research done in Tigrigna text classification system until the knowledge of the researcher. Even the main focuses of local researches have been on news text classification.

Since there are a number of documents produced by different subject domains in the form of articles, research results, and reports that are electronically available, there is a need to design text categorizer for efficient search and retrieval. Hence, the present study aims to develop Tigrigna text documents categorization system using clustering and classification approach.

To this end, this study attempts to address the following research questions:

- Which clustering algorithm is more suitable for labeling unlabelled Tigrigna text documents?
- Which learning approach is more appropriate for creating classification model that helps in Tigrigna text categorization?
- To what extent the proposed model is able to predict according to experts judgment?

1.3 Objective of the Study

In an attempt to explore and design Tigrigna text categorization system, the general and specific objectives of this study are formulated as follows.

1.3.1 General objective

The general objective of this research is to design a text categorization system for Tigrigna language. This is of paramount importance for target users to find relevant information from a collection of Tigrigna documents.

1.3.2 Specific objectives

In order to achieve the above stated general objective, the following specific objectives are formulated.

- To conduct a thorough review of literature on the existing text categorization techniques and methods in general, and application of text classification by clustering in particular.
- To assess different techniques used for text categorization and select more suitable once for categorization.
- To identify meaningful patterns and relationship of the different categories of the Tigrigna text.
- To prepare the appropriate data sets from different documents for training and testing purposes.
- To preprocess the data in order to select discriminating terms of the Tigrigna text documents.
- To cluster the preprocessed Tigrigna text documents in to their natural groups (categories) which can be used for text classifier learning.
- To construct a classification model that helps to correctly categorize text documents written in Tigrigna language.
- To evaluate the performance of the proposed model using the selected evaluation metrics and recommend future research direction to improve the performance of the proposed system.

1.4 Methodology

Methodology provides an understanding of how a proposed research is conducted in order to obtain information for developing the proposed systems (Dawson, 2002). So the methodology contains tools and techniques that can be used for conducting the study. As

a result, the present study uses the following important aspects for constructing a Tigrigna text categorization system.

1.4.1 Literature review

In order to get a good understanding of text categorization and the Tigrigna language relevant published text documents are reviewed. Different books, journal articles, news, previous related research works and electronic publication on the web have been consulted in order to design an effective Tigrigna text categorization system.

1.4.2 Design procedures

In the present study, Tigrigna text categorization system is designed in three steps: these are preprocessing, clustering, and classifier building and evaluation. During preprocessing step, the relevant terms that represent and discriminate the Tigrigna text documents are selected. Generally, the preprocessing step includes tokenization, stemming, removing stop words, term weighting and dimension reduction for feature selection.

After preprocessing, the data is clustered into different natural groups using the clustering algorithm. From the clustered data set, the training is created. The training data set is used to build a learning classifier. Finally, testing set is prepared by an expert to evaluate the performance of the system.

1.4.3 Data source and data set preparation

The data sets are collected from different books, articles, journals etc. These data sets collected are in word document format. The format of data sets is converted in to text for preprocessing. Then the data was preprocessed through two phases. In the first phase, python programming language is used to remove extraneous characters from the collection. Python is used because the researcher is familiar with the programming language and it is an effective tool for text processing. In the second phase, the preprocessing is done using python and Perl programming. Like python, Perl is also used based on its familiarity with the researcher. The researcher uses python programming to remove the irrelevant term of Tigrigna language from the text and the Perl programming to create a document term matrix that is used for the clustering purpose.

1.4.4 Tools

The main tasks performed for developing Tigrigna text document categorization are clustering and classification using gCluto and Weka, which are the most freely available tools of text categorization. gCluto is selected for clustering Tigrigna text documents because it has an intuitive graphical user interface, and interactive visualizations of the clustering solutions(Karypis et al., 2004). While Weka is used for classification of Tigrigna documents because it allows users to quickly try out and compare different machine learning methods on new data sets and it has also several graphical user interfaces that enable easy access to the classification functionality (Baker and Kachites, 1998).

1.4.5 Evaluation procedures

In the present study, the evaluation procedure is performed in two stages. In the first stage the clustered data is evaluated using the entropy and purity. In the second stage, the text classifier is evaluated using test set to evaluate the effectiveness of the categorization system. The performance of text categorization measures using accuracy, error rate, precision, recall, and F-measure.

1.5 Scope and Limitation of the study

The scope of the present study is limited to design text categorization systems for Tigrigna language. To come up with an optimal categorization, a two-step approach is explored using first clustering approaches and then classification techniques. Among the available techniques the k-means and its variant repeated bisection clustering, as well as decision tree and support vector machine classifiers are used. The data set composition of this study is only text file formats; other multimedia formats are not considered.

The system is trained and tested using limited amount of Tigrigna text documents collected from Tigray Region and Tigrigna news media. This is due to lack of standard Tigrigna corpus prepared for text categorization purpose. Since there is no normalization algorithm available we are unable to standardize abbreviated words occurring in Tigrigna documents.

1.6 Significance of the Study

Text categorization is a supporting technology in several information processing tasks including controlled vocabulary indexing, routing and packaging of news and other text streams, content filtering (spam, pornography, etc.), information security, help desk automation, and others(Sebastani, 2002). Hence, the results of this research will be used as an input to the development of full-fledged text categorization of Tigrigna language.

1.7 Organization of the Thesis

This thesis is organized into five chapters. First chapter introduces the general concepts of the thesis which contains background of the study, statement of the problem, objective of the study, methodology, scope and application of the study.

Chapter two describes the concept and techniques of text categorization in general and text clustering and classification in particular. Text categorization phases, Tigrigna writing systems, and related literature reviews are also described in details.

In chapter three, the methodology followed in developing the proposed system is elaborated. It contains the algorithms and methods used in designing the proposed systems.

Chapter four is the experimentation and evaluation of the study. This part of the thesis shows the implementation details, experimental results, analysis and finding of the study.

Chapter five clearly elaborates the concluding remarks and the recommendations for further research work.

CHAPTER TWO

LITERATURE REVIEW

With the rapid growth of document collection, the existing retrieval system has difficulty for retrieving the relevant document. To improve this difficulty of the retrieval system, text categorization (TC) plays key roles for handling and organizing the text data. Most of the categorization systems categorize documents into a fixed number of predefined categories. Each document can be categorized in to multiple, exactly one, or no category at all (Sebastiani, 2002).

Text categorization techniques are necessary nowadays because most information is produced and stored digitally. There are different electronic text collections available in the form business and personal correspondence, scientific and entertaining articles, conference proceedings, and patient data.

2.1 Text Categorization

Text categorization is the task of automatically assigning input text to a set of categories. Text categorization can be divided in to text classification and text clustering based on the category it uses (Sebastiani, 2002). Text clustering is the automatic identification of a set of categories and assigns set documents under the automatically identified categories. The main idea in text clustering is to find which documents have many words in common, and place the documents with the most common words into the same groups. As a result, this text categorization approach categorizes documents without having a pre-defined category. It is an example of unsupervised learning text categorization approaches. As noted by Bharati (2002), the unsupervised learning does not require an external knowledge to categorize the set of documents. However, text classification classifies the set of documents in to a predefined category which is labeled by a domain expert. Text classification is an example of supervised learning since it requires an external knowledge to construct a predefined category for each document.

According to Cheng et al. (2001), text categorization is the capability of labeling natural language texts from a domain D with thematic categories C = {c_1, c_2,. . . , $c_{|C|}$} . The construction of an automatic text classifier relies on the existence of an initial corpus Ω = {d_1, d . . . $d_{|\Omega|}$} of documents pre-classified under C. A general inductive process (called the learner) automatically builds a classifier for C by learning the characteristics of C from a training set Tr = {d1, . . . ,$d_{|Tr|}$} of documents (Deng et al., 2002).

In order to understand the fundamental of text categorizations, the following two observations are important (Maron and Kuhns, 1960):

- The categories used in the text categorization process are symbolic labels and the meaning of the text is not assumed in building the text categorizer.
- The attributes of text documents to categories should be realized on the basis of the semantics of the documents, and not on the basis of metadata (e.g., publication date, document type, publication source, etc.).

Generally, the categorization of a document should be based solely on endogenous knowledge. The knowledge can be extracted from the document itself rather than on exogenous knowledge (i.e. data that might be provided for this purpose by an external source).

However, the semantics of a document is an inherently subjective notion, which follows the fundamental notion of text categorization. The semantics of the documents is the internal representation of the documents. Most of the time documents are represented as vectors in the term space. The term space contains two major issues; a set terms and their weight in the documents. The term and its weight are used as an input to determine documents category.

Text categorization is the activity of automatically building text classifier by means of machine learning (ML) techniques (Sebastiani, 2000). The automatic text classifier is capable of labeling natural language texts with thematic categories from a predefined set. A commonly used text classifier builds independent class which should be capable of deciding whether a given document should or should not be classified under a specified

category. Text categorization can be divided into different categories using different criteria (Sebastiani, 2002). Depending on the application area, text categorization can be single-label or multi-label, on the other hand depending on the use of a text classifier text categorization can be document-pivoted, or category pivoted, further more based on the automation of the system text categorization can be hard or soft.

2.2.1 Single-label versus Multi-label text categorization

The single-label text categorization problem assigns only one predefined category to each "unseen" natural language text document and often defined as non-overlapping (Antonie, and Zaiane, 2002). In this label for a given integer k each element of C must be assigned to exactly k (or < k, or >k) elements of D. For instance, this happens when the category needs to be evenly populated (Berger, 2004). So, it assigns an object to exactly one category when there are two or more categories in the category spaces.

However, it is impossible to categorize each document under a single label because of the nature of the text overlapping each other in the category spaces. For example, the economics field often overlaps (relates) with the political science field. This fact forces the different constraints on single-label text categorization task.

Whereas the multi-label case is general case in which any number of categories from 0 to M (M is at least one) may be assigned to the same document (Popa et al., 2007). The multi-label text categorization assigns more than one predefined category to an "unseen" document and it is called as overlapping text categorization tasks because it is the task of assigning an object simultaneously to one or multiple category.

According to Addis (2010), the single-label is more general than the multi-label categorization. Because the algorithm for single-label can be used for multi-label by transforming a problem of multi-label with categories $\{c_1,c_2\ldots,c_m\}$ into m independent problems of single-label classification with categories $\{c_i\}$, for i = 1,2,…,m. This can be done if the categories are stochastically independent of each other. However, the converse is not true in general. If there is algorithm for performing the multi-label, it is not always true that it can use for single-label categorization.

2.2.2 Category-pivoted versus Document-pivoted text categorization

The category-pivoted categorization (CPC) fills the decision matrix one row at a time where as the document-pivoted categorization (DPC) fills the decision matrix one column at a time. According to Larkey (1999), DPC is commonly used when documents are available at different moments in time, for example in filtering e-mail. However, CPC is used mostly when a new category $c_{|c|+1}$ is needed to add to an existing set after a number of documents have already been classified under it ,and these documents need to be reconsidered for classification under $c_{|c|+1}$.

2.2.3 Hard categorization versus Soft categorization

The hard categorization completely automates the text categorization which requires a true or false decision for each pair (d_j, c_i) where soft categorization uses partial automation of the text categorization system which requires different methods. For a given document d_j in a documents D, the system may rank the categories in C = {c_1, c_2… $c_{|C|}$} according to their estimated appropriateness to dj without taking any hard decision on any of the categories. This is useful especially in critical applications in which the effectiveness of a fully automated system may be expected to be significantly lower than that of a human expert. This ranked list would have a great advantage for human expert for taking the final decision, because she/he would rank the categories based on his /her choice (Addis, 2010).

For automatic categorization text it is preferable to use hard categorization because the hard categorization fully automates the text documents of the specified language (Sebastiani, 2002). There are also different approaches of text categorization. These are flat and hierarchical text categorization. In flat categorization, the single-label is the commonly used mechanisms of categorizing documents. In hierarchical text categorization, the multi-label text categorization mechanisms are used commonly. The document-pivoted and category-pivoted can be used based on the document occurrence of the specified time.

2.2 Text Categorization Techniques

Text categorization techniques can be supervised, unsupervised and semi supervised learning (Ozgur, 2004). Supervised learning is the search for algorithms that reasons from externally supplied class to produce general hypotheses, which then make predictions about future instances. In other words, the goal of supervised learning is to build a concise model of the distribution of class labels in terms of predictor features (Ozgur, 2004). The resulting classifier is then used to assign class labels to the testing instances where the values of the predictor features are known, but the value of the class label is unknown. Text classification is an example of supervised learning.

The unsupervised learning does not require externally supplied knowledge for categorizing instances. The main aim of this learning algorithm is to generate a group of features that have similar properties. So the grouping is done without any supervision of human beings. Text clustering is an example of unsupervised learning. In semi-supervised learning, parts of the documents are required external knowledge and other does not require external knowledge in the categorization process. In this method of learning, training data contain both labeled and unlabeled. Most of the time, the categorization processed developed by semi-supervised learning is used as a method to extract information from the unlabelled data in order to enhance the classification task.

In automatic text categorization, different techniques are used to enhance the categorization of a large number of documents. The most commonly used techniques in the current text categorization processes are text clustering and text classification (Ozgur, 2004). In text classification, a document is assigned to a predefined category based on the likelihood suggested by labeled documents in training set. In text clustering, the systems automatically find natural grouping of documents.

2.2.1 Text classification

Text classification is an example of supervised learning where a given document is assigned to predefined categories based on the similarities of the labeled documents in the training set (Sebastiani, 2002). Text classification can be done manually or automatically. Traditionally, text classification has been performed manually (Sebastiani,

2005). The manual text classification uses expert (human being) to categorize the document in to predefined categories. However, as the number of documents explosively increases, the task becomes no longer amenable to the manual categorization, and it requires a vast amount of time and cost. This has lead to numerous researches for automatic document classification. The automatic text categorization is generally divided in to two main categories (Popa et al., 2007). These are flat text categorization and hierarchical text categorization.

Most of the research in text classification has been done using flat text categorization, which is concerned with classifying text documents into categories with no structural relationship among them (Popa et al., 2007). It has huge classifier trained for categorizing each new document as belonging to one of the possible basic classes. As the number of topics becomes larger, the flat categorization faces the problem of complexity. The complexity includes large space requirement and large amount of time for processing the stored text documents. A common way to solve this complexity is using hierarchical text classification. It arranged the different documents in a tree shaped hierarchy. In addition, internet directories and large on-line databases are often organized their text documents in hierarchical ways. When the web documents are organized in this way, the user would find it easier to navigate in the hierarchy of categories and restrict her (his) search to a particular category of interest (Sebastiani, 2002). Many learning algorithms are used for classifying the text documents. The most commonly used are k-nearest neighbor, support vector machines, neural networks, decision tree, boosting and naive Bayes (Addis, 2010).

A decision tree uses a binary classifier (True or False) for assigning the new category to the existing category (Deng et al., 2002). So, it builds a binary tree of the category and the decision is made whether true or false. If the decision is true, then the new category is relevant to the specified category. Otherwise, false value shows the need to investigate based on other criteria until a specific category is satisfied.

The k-nearest neighbor algorithm is one of the simplest automatic learning algorithms (Dhillon, 2003). It classifies a document by its neighbors and assigns the document to the most common category of its k-nearest neighbors. Here k in the class indicates positive

integer. The training examples are arranged in multidimensional vector space and the space is divided into class by locations and labels of the training samples. A document in the category is assigned to the class c if it is the most frequent class label among the k-nearest training samples. There are some drawbacks in this learning algorithm (Ozgur, 2004):- first, the frequent class dominates the prediction of the new vector (Dhillon, 2003). Second, there is no training identified in it and it also contains noise feature (Addis, 2010). However, it best performs when the amounts of data approaches to infinity.

A naïve-Bayes classifier uses the training data to predict the probability of each category in a given documents (Deng et al., 2002). It works on a simple concept. It makes use of the variables contained in the data sample by observing them individually and independent of each other. However, the assumption of independence is not always reliable for word appearance in documents.

In support vector machine the terms are represented in a vector space and δ_1, δ_2,… in |T|-dimensional space separates the positive training examples from the negative training examples (Sebastiani, 2002). So the δ_i separates the positive from the negative training by the margin. Support vector is best applicable when the positives and negatives are linearly separable. The support vector machine has advantage for text categorization because it does not require dimensional reduction of the given documents (Deng et al., 2002). In addition, it solves the problems of over-fitting and helps to understand categorization in an easier ways.

Boosting is used as a mechanism to increase the learning capability (Addis, 2010). This algorithm generally begins by building an initial model from the training data set. The entities in the training data set have boosted their weights and then, a new model is built with those boosted entities. This model is used to make decision on new data by combining the expertise of each model.

Text classification has several application in the real world such as automated indexing of scientific articles according to predefined thesauri of technical terms, filing patents into

patent directories, selective dissemination of information to information consumers, automated population of hierarchical catalogues of Web resources, spam filtering, identification of document genre, authorship attribution, survey coding, and even automated essay grading (Sebastiani, 2002). Text classification systems learn models of categories using a large corpus of labeled data. However, the labeling is done manually by domain experts. Due to tedious and subjective nature of manual labeling, labeled data are difficult and expensive to obtain. Whereas, unlabeled data are plentiful and easy to obtain.

2.2.2 Text clustering

It is easy to collect unlabeled documents for text categorization purposes. As a result, a text categorization mechanism is required for categorizing the unlabeled documents. This mechanism is called text clustering (Ozgur, 2004). Text clustering is an unsupervised learning which does not require pre-defined categories and labeled documents (Dhillon, 2003). The main aim of text clustering is to determine the intrinsic grouping in a set of unlabeled data. The intrinsic groups have high intra-group similarities and low inter-group similarities.

Text clustering algorithms are categorized into two main groups such as hierarchical clustering and partitioning clustering algorithms. Partitioning clustering algorithms create a cluster by splitting the data into k partition where each partition represents a cluster. While hierarchical clustering algorithms create a tree shaped structure of data by splitting (divisive approach) or merging (agglomerative approach) clusters based on the similarity among the groups (Ozgur, 2004).

Divisive clustering algorithms start with all documents in one cluster and split at the point where cluster are dissimilar in each iteration until stopping criteria k is achieved. Agglomerative clustering algorithms on the other hand start with each document in a separate cluster; the most similar pairs of clusters are merged in each of the iterations. Agglomerative clustering is the common hierarchical clustering algorithms. The main process of hierarchical clustering can defined as follows (Dhillon, 2003): start by assigning each item to a cluster; next find the closest (most similar) pair of clusters and

merge them into a single cluster; and then compute distances (similarities) between the new cluster and each of the old clusters; finally, repeat steps 2 and 3 until all items are clustered into a single cluster.

Computing distances between new cluster and each of the old clusters in the hierarchical clustering can be done in different ways. Based on this, the hierarchical text clustering can be divided in to three main categories (Baker and Kachites, 1998). These are single-linkage, complete-linkage and average-linkage clustering. In single-linkage, the distance between one cluster and another cluster is equal to the shortest distance from any member of one cluster to any member of the other cluster. In complete-linkage clustering, the distance between one cluster and another cluster is equal to the greatest distance from any member of one cluster to any member of the other cluster. In average-linkage clustering, the distance between one cluster and another cluster is equal to the average distance from any member of one cluster to any member of the other cluster.

The partitioning clustering algorithms group data into un-nested and non-overlapping groups that usually optimize a clustering (Ozgur, 2004). This technique usually produces clusters by optimizing a clustering criterion function. There are different partitioning clustering algorithms (Dhillon, 2003): k-medoids and k-means are commonly used clustering algorithms (source). K-means clusters a given data set through a certain number of clusters. The main aim of k-means clustering algorithm is to define k centroids which are one for each of the clusters created in the clustering process. The k-means clustering algorithm performs the clustering process as follows (Ozgur, 2004): first, k point of the cluster is selected as initial centroids. Second, it assigns all points to the closest centroid. Third, it again computes the centroid of each cluster. Finally, second step and third step are repeated until the centroids do not change. There are different variant of k-means clustering algorithms. The repeated bisection clustering algorithm is most commonly used in text categorization (Slonim, 2001). The repeated bisection algorithm first selects a cluster to split, and then employs basic k-means to create two sub-clusters, repeating these two steps until the desired number k of clusters is reached.

Like the k-means, k-medoids is also a partitioning clustering algorithm which breaks the data set in to groups. However, the k-mediods clustering algorithm uses the mediod as a centers rather than k centroids. The mediod in the clustering algorithm is the most centrally located point in the given data set (Baker and Kachites, 1998). In the k-medoids clustering algorithm, the actual objects are selected to represent the clusters using a representative object from each object. The remaining object is clustered with their representative object to which it has most similar properties. Generally, the clustering algorithm is done by minimizing the sum of dissimilarities between each object.

Most of the text categorization researches indicate that both k-means and repeated bisection clustering algorithms are relatively efficient and scalable, and their complexity is linear to the number of documents (Zhao, 2002). Slonim (2001) also shows that the repeated bisection algorithm performs better than other clustering algorithms in terms of accuracy and efficiency.

Text clustering has several applications such as query routing, cluster-based browsing, result set clustering, query refinement etc. However, there are number of problems in text categorization based on the clustering algorithms (Zhao, 2002). Some of them are: dealing with large number of dimensions and large number of data items is difficult because its time complexity; and the results of the clustering algorithm can also be interpreted in different ways.

2.2.3 Hybrid approaches for text categorization

In the real world, there is available unlabeled data but labeling them is expensive because it requires human expert. As a result, there are a limited number of labeled data (Ozgur, 2004). Using the unlabeled data for text categorization is not preferred due to its time complexity and interpretation problems. So a mechanism is required to combine clustering and classification algorithms for text categorization process (Slonim, 2001). First, clustering is used with text classification prior to a classifier to reduce feature dimensionality by grouping similar features into a much smaller number of feature clusters. These clusters are used to the original feature space (Zhao, 2002). Second, clustering is used with text classification as feature clustering and document clustering. In

this way a reduction for both dimensions is attained. Feature clustering generates coarser pseudo features, which reduce noise and sparseness that might be exhibited in the original feature space. In the second stage, documents are clustered as distributions over the "distilled" pseudo features, and therefore generate more accurate document clusters (Dhillon, 2003). Third, clustering is used in semi-supervised classification as a method to extract information from the unlabelled data in order to boost the classification task. Particularly clustering is used to create a training set from the unlabelled data, and to augment the training set with new documents from the unlabelled data (McCallum et al., 2000).

2.3 Text Categorization Phases

Text documents represented in a natural language are not easily used by the classifier for building algorithms. In order to solve this problem mapping a text document into a schematic representation of its content is required. As a result, the categorization algorithm transforms each document into a vector of weights corresponding to an automatically chosen set of keywords. This transformation has two main steps (Popa et al., 2007). First, suitable representation of the document has to be chosen. This representation is used for all documents to be indexed and it has all necessary words that can characterize the documents. The information retrieval and machine learning researchers have different views in representing strings (Sebastian, 2002). IR researcher suggests that word stems work well as representation units and that their ordering in a document is of minor importance for many tasks. In the machine learning research community, the dominant approach to this problem is based on machine learning technique which is a general inductive process that automatically builds a classifier by learning from a set of pre-classified documents, and characteristics of the categories. Second, it assigns weights to each representation term which shows the frequency of occurrence of the term in the indexed document. Even though the document indexing is performed, the results obtained has high dimension and take a large amount of storage space. To address these problems, techniques based on dimensionality reduction have been explored for capturing the concepts present in a collection. The main idea behind

these techniques is to map each document into a lower dimensional space that can potentially take into account the dependencies between the terms.

2.3.1 Document indexing

The task of document indexing involves mapping of a document d_j into suitable representation of its content that can be used for building the classifier algorithms. The choice of the representative term of a document depends on the individual choice of meaningful terms (Addis, 2010). Generally, there are two main steps for indexing documents in the given corpus (Sebastiani, 2002). Representative terms are selected from the documents. After selecting the representative terms, the non-discriminating terms are removed from the documents in the given corpus. This process is also called feature selection step. As a result, most researchers use representative term and feature selection interchangeably (Addis, 2010). The removed terms are both the frequently and very rarely appearing terms because such words or terms do not distinguish one document from other document in a given corpus. Most of the time, feature selection consists of the following process (Maron and Kuhns, 1960):-

- All documents in the corpus are processed and their matrix of all words, their frequency appearance is counted.
- Punctuation, numbers and special characters are removed from the matrix.

Second, weight is assigned to each document. This represents the documents by numeric vector. The numeric vector includes the weight of the term in a document. So, the weight factor should represent the importance of the term for the categorization of the document.

The most common term weighting approaches used in text categorization are Boolean weighting, term frequency weighting, and term frequency × inverse document frequency weighting (Buckley et al., 1998). There are certain concepts that need to define in term weighting:

- tf_{ij} is the frequency of term *i* in document $\mathbf{d_j}$;
- N is the total number of documents in the document corpus;
- N_i is the number of documents in the corpus where term *i* appears; and
- |T| is the number of distinct terms in the document collection (after stop word removal and stemming is performed).

Boolean weighting is the simplest method of term weighting and it assigns 1(existence) if the term in a document exists or zero (absence) if the term does not appear in the document. Here, the weighting is only existence or absence that does not show in which documents the term appears more or less. Due to this, this is not widely used term weighting approaches.

Term frequency weighting counts the appearance of the term in documents. In this term weighting, the weight of a term in a document is equal to the number of times the term appears in the document. Sometimes, the most frequent term could not discriminate one document from other documents. If the term frequency of the term is high, its discriminating power to the mean documents is low. So, this term weighting techniques is no mostly used in text categorization processes. As a result, the term frequency × inverse document frequency weighting (tf×idf) which uses the frequency of the most discriminating term in a given documents is most commonly used (Addis, 2010). Here, the weight of term i in document d is proportional to the number of times the term appears in the document and inverse proportional to the number of documents in the corpus in which the term appears. tf×idf function can be defined as follows:

$$Wtij = tfij \times log\left(\frac{N}{Ni}\right) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.1$$

Equation 2.1 indicates that $tfij$ is the term frequency term i in a document j, $\log\left(N/N_i\right)$ is the inverse document frequency of the term, N is the total document number in the corpus, and N_i is the number of documents the term appears. The tf×idf weighting approach weights the frequency of a term in a given document with a factor that discounts its important if it exists in most of the documents. At the end of this process, the index file is constructed using indexing structure such as inverted file, signature file etc.

An index file stores a mapping from contents such as terms to its locations in a document or a set of documents. The purpose of indexing is to have fast searching mechanism when there are a lot of documents in the database. The most widely used indexing structure is inverted file which can be represented in two ways (Popa et al., 2007): an inverted index

file containing a list of references to documents for each word and the inverted index file which contains the documents which the term appears, and the position of each word within a document.

2.3.2 Dimensionality reduction

After the index file is generated the dimension of the index file is reduce in order to save the storage space and enhance the processing speeds. The dimensionality reduction maps each document into lower dimensional space. This improves the categorization performance of a given text documents. There are various dimensionality reduction techniques that can be classified as either supervised or unsupervised (Karypis et al., 2004). Supervised dimensionality reduction techniques use the class-membership information for computing the lower dimensional space. Examples of supervised dimensionality reduction techniques are document frequency (DF) and information gain (IG). However, unsupervised dimensionality reduction techniques compute a lower dimensional space without using any class-membership information. These techniques are primarily used to improve the retrieval performance and rarely used for document categorization. Examples of such techniques include principal component analysis (PCA), latent semantic indexing (LSI), kohonen self-organizing map (SOFM), and multi-dimensional scaling (MDS). However, the unsupervised dimensionality reduction methods are not commonly used in text categorization process (Addis, 2010).

2.3.3 Classifier learning

A learning classifier is a function that maps an input attributes to the class membership it belongs to (Addis, 2010). The attributes in this classifier are the list of terms found in the document, whereas the classes are the predefined categories to which the documents belong. In automatic text categorization, a text classifier for a given category is automatically generated by a learner. The learner observes the characteristic of a given document under a pre-defined category and determines the new unseen document to specified category. Most of the time evaluation procedure of learning classifiers in text categorization uses three data sets (Addis, 2010). These are training set (Tr), validation set (Va), and test set (Te). The training set is the set of documents observed when the learner builds the classifier. After building the training set the validation set is used for

choosing for a parameter on which the classifier depends and for evaluating the effectiveness. Finally, the test set is used to evaluate the effectiveness of the classifier. The test set is the set on which the effectiveness of the classifier is finally evaluated. Both the validation set and test set are used for evaluating the effectiveness of the classifier.

2.3.4 Evaluation metric

Text categorization rules are typically evaluated using different performance measurement techniques. These techniques can be categorized in to two main categories (Ozgur, 2004): text clustering and text classification performance measures.

In clustering techniques, there are two types of measures for evaluating the cluster results. These are internal quality measure and external quality measure (Sebastiani, 2002). The internal quality measures do not use external knowledge set such as class label information to evaluate the clustering solution results. However, the external quality measure evaluates the clustering solution based on the labeled test document corpus. It compares the resulting clusters to labeled classes and measures the degree to which documents from the same class are assigned to the same clusters (Karypis et al., 2004). The most common evaluation metric for clustering techniques are overall similarity which is an internal evaluation measure, and purity and entropy which are an external evaluation measures of the clustering solution.

Overall similarity measure is an internal evaluation that uses weighted similarity of internal cluster to measure the cohesiveness of the cluster solution (Baker and Kachites, 1998). Internal cluster similarity I for cluster C_j can be computed as:

$$I_j = \frac{1}{n_j^2} \sum_{d \in C_j} \sum_{d' \in C_j} \cos(d, d') \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.2$$

Where, n_j is number of documents in cluster j.

Overall similarity of the clustering solution can be computed based on the above internal similarity as follows:

$$\text{overall similarity} = \sum_j \frac{n_j}{N} I_j \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.3$$

Where, N is the total number of documents in the corpus.

Entropy shows that the best clustering results is the one that leads to clusters that contain documents from a single class, in which the entropy is zero. Here, the better clustered result is assumed when the entropy value is smaller. Entropy measures how various classes of documents are distributed within each cluster of data set. In order to measure entropy, the class distribution is measured for each cluster and then this class distribution is used to calculate the entropy for each cluster as indicated in equation 2.4.

$$E_j = -\sum_i p_{ij} \log\left(p_{ij}\right) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.4$$

Here, the pij is the probability of the member of cluster j belongs to class i and E_j is the entropy of individual clusters. In addition, the entropy of all produced clusters in the given clustering algorithm is calculated as the sum of the individual cluster entropies weighted according to the cluster size, and as defined in equation 2.5.

$$E_{SC} = \sum_{j=1}^{m} \frac{n_{j \times E_j}}{n} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.5$$

Where n_j is the size of cluster j, n is the total number of documents, m is the number of clusters, and E_{sc} is the entropy of all clusters.

Purity measures the performance of clustering algorithms when each cluster contained documents from primarily one class. As a result, it measures the largest class for each cluster. Here, the better clustered result is assumed when the purity value is larger. Like the entropy, the purity of each cluster and purity all produced clusters is calculated in equation 2.6 and equation 2.7 respectively.

$$P_{(S_r)} = \frac{1}{n_r} \max(n_r^1) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.6$$

Equation 2.6 shows that S_r is a particular cluster of size n_r, $P_{(S_r)}$ is the individual cluster purity. The purity of all produced clusters is also computed as a weighted sum of the individual cluster purities and is defined as follows:

$$\text{Purity} = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.7$$

Finally, a perfect clustering solution is the one that leads to clusters that contain documents from only a single class, in which case the entropy is zero, and the purity is 1.

Performance of the classifier is also measured in accuracy, recall, precision, and F-measure. Accuracy refers to the percentage of correct predictions made by the model when compared with the actual classifications in the test data (Baeza-Yates and Ribeiro-Neto, 1999). There are also other different performance measurement of the classifier such as recall, precision, and F-measures. Precision (P) measures the percentage of documents assigned to category c that are correctly assigned to category c. More formally, the precision is defined as shown in equation 2.8. More formally, the precision is defined in equation 2.8.

$$P_i = \frac{TP_i}{TP_i + FP_i} \quad \ldots\ldots 2.8$$

On the other hand, recall measures the percentage of total documents that are assigned to category c. It is defined as:

$$R_i = \frac{TP_i}{TP_i + FN_i} \quad \ldots\ldots 2.9$$

Where TP_i (i.e., true positives) is the number of documents assigned correctly to category c_i, FP_i (i.e., false positives) is the number of documents assigned to category c_i that should have been assigned to other categories, and FN_i (i.e., false negatives) is the number of documents assigned to other categories that should have been assigned to category c_i.

The F-measure (F) is the harmonic average of precision and recall, and is defined as:

$$F_i = \frac{2 \times P_i \times R_i}{P_i + R_i} \quad \ldots\ldots 2.10$$

Where P_i and R_i are the precision and recall for category c_i respectively.

Most of the time, the F-measure is used to measure the performance of the text classifier because it is the combined approaches of precision and recall that favors point registering highest recall and precision.

2.4 Tigrigna Writing Systems

The Semitic languages are the Afro-Asiatic language family. The most widely spoken Semitic languages today are Arabic, Amharic, Tigrigna, Hebrew, and Aramaic. There are different Semitic languages in Ethiopia. These are Amharic, Tigrigna, Gurage, Argobba, Gafat, Ge’ez etc (Hammond,1999).

Tigrigna is a Semitic language which is spoken by the Tigray people located in northern Ethiopia and Eritrea. For those regions, Tigrigna is a working language. There are more than 6 million Tigrigna speakers worldwide (Leslau, 1998). Tigrigna language has its own characters (alphabets), punctuation marks, and numbers systems. Converting them into Ethiopic representation is required for text categorization purpose. SERA is a tool that facilitates conversion of Ethiopic writing systems for text analysis. It translates the Tigrigna text into suitable representation for computer understandability (Worku, 2009).

2.4.1 Origins of Tigrigna language

The Ethiopic writing system is used to represent the four Semitic languages. These are Ge'ez, Amharic, Gurage, and Tigrigna. These languages are confined to Ethiopia and Eritrea. Ge'ez is a dead language. It is no more mother tongue of any person, but it still has a very significant role in the traditional language of literature and religion. Amharic and Tigrigna are closely related to each other.

Ethiopic writing system is written from left to right. It does not make any distinction between upper and lower case letters and has no conventional cursive form. There are no systematic variations in the form of the symbol according to its position in the word. The Ethiopic system for Tigrigna language consists of alphabets, numbers, and punctuation marks (Briggs, 2002).

2.4.2 Alphabets

Alphabets are sets of letters arranged in fixed orders of the language they used to write. They are also called phonemes which contain consonants and vowels. There are different alphabets representations in the world. The most alphabets representation is Latin or Roman alphabets which have been adapted by numerous languages. The Ethiopic writing systems have also their own writing systems. Similarly, Tigrigna has its own alphabets (ፊደል) and they are used for writing different documents of Tigrigna language. It has thirty-five base symbols with seven orders which represent seven vowels for each base symbol (Leslau, 1998).

The Tigrigna writing system can be translated in Latin representation by finding a Latin letter with similar sound Tigrigna letter. For example the Tigrigna letter 'ረ' has a similar

sound with Latin letter ‘r’. As a result, the seven order of the letter ‘ረ‘ can be represented by combining the Latin letter ‘r’ with vowels as shown below in table 2.1.

Order	1st	2nd	3rd	4th	5th	6th	7th
Tigrigna letter	ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
Equivalent Latin letter	re	ru	ri	ra	rie	r	ro

Table 2. 1: Sample of Tigrigna letter and their corresponding Latin letter

As shown in table 2.1 the Tigrigna letter and its seven orders can be translated in to similar sound Latin letter. The seven orders of the Tigrigna letter can be translated by combining the similar sound Latin letter and vowels. One can understand from the above table that the first order in Tigrigna represents vowel /e/, the second vowel /u/, the third vowel /i/, the fourth vowel /a/, the fifth vowel /ie/, and the seventh vowel/o/. The sixth order represents the consonant with no vowel. The list of Tigrigna characters are described in appendix I.

2.4.3 Punctuation marks

Identifying punctuation marks is vital to know word demarcation for natural language processing. Most of the Tigrigna language punctuation marks are listed in table 2.2 below. The word separator mark (:) is used in the old literature to separate one word from other words. In the current literature, it is rarely used. As, a result a single space is used to separate words instead of this punctuation marks. The end of sentence mark (::) is used to shows when an idea is finished. The sentence connector mark (፡) is used to connect two sentences in to one sentence. The list separator mark (፣) is used to list things, separate parts of a sentence, and indicate a pause in a sentence or question. Like the other punctuation marks, the beginning of the list mark (፡-) is used at the beginning of the lists. In addition to these punctuation marks, the Tigrigna language is also borrowed some punctuation marks from English language such as ?, !, and ”.

Punctuation marks	Meaning
፡	word separator
።	end of sentence
፤	sentence connector
፣	list separator marks
፡-	beginning of the list mark
?	end of question
!	end of an emphatic declaration, or command.
”	quote some words or sentences taken from other

Table 2. 2: List of Tigrigna punctuation marks

2.4.4 Number system

Like Amharic numbering systems, Tigrigna number system uses Ge’ez numbering systems. It has twenty characters. They represent numbers from one to ten (፩-፲), twenty to ninety (፳-፺), hundred (፻) and thousand (፼). However, these are not suitable for arithmetic computation purposes because there is no representation for zero (0), decimal points. As a result, Tigrigna numbers are used for calendar purposes. For arithmetic computation, western numbers are used in the Tigrigna literature. The complete Tigrigna writing system is depicted in appendix II.

2.4.5 Problem of Tigrigna writing system

Leslau (1998) notes that there are a number of challenges in Tigrigna language for text processing.

2.4.5.1 Redundancy of some characters

Sometimes more than one letter is used to represent similar sound in Tigrigna language. For instance, letters “ሀ” and “ኀ”; “ጸ” and “ፀ”; “ሰ” and “ሠ” have similar sounds. In the old literature of Tigrigna texts, the use of various forms of characters for the same sound has a problem in the process of feature preparation for the classifier learning. However, current literatures do not have such problems since it has only one letter for one sound. As a result, the alphabet “ኀ”, “ጸ” and “ሠ” are no more in use in writing Tigrigna document.

2.4.5.2 Spelling variation of the same word

Even though there are feasible problems of spelling variation in current literature of Tigrigna. A word may be translated by different persons using different spelling variation. For instance, when “television” word is translated into Tigrigna, it may be written as “ተለበሾን”, “ተለቭሾን or “ተለበጅን”. All these words are used to mean the word “television” in Tigrigna. The translation of English words into Tigrigna words increases ambiguity and inconsistency in the categorization of Tigrigna text documents.

2.4.5.3 Abbreviation

The abbreviations of Tigrigna words follow different formats. Some time full stop ‘.’ is used to abbreviate, while other time ‘/’ symbol is used to abbreviate. The abbreviated words can be written without separators. For example, “ገብረሂወት” (Gebrehiwot) can be written as “ገ/ሂወት” (G/hiwot) or “ገ.ሂወት” (G.hiwot). The inconsistency in the abbreviation creates inconsistency in text categorization processes.

2.4.5.4 Compound words

Tigrigna compound words are written in different format. Mostly space and hyphens are used to separate them. When the hyphen is used the two words are treated as one word. However, when they are separated by space their meaning differ. For example, “ቤት ትምህርቲ”, “ስነ ስርዓት”, “መራሕቲ ስድራ”, and “ኣብያተ ፅሕፈት” are compound words separated by space. However, words “ቤት”, “ስነ”, “መራሕቲ”, and “ኣብያተ” do not have meaning when they are used separately. So this can create problems in text categorization.

2.4.6 System for Ethiopic representation in ASCII (SERA)

SERA is a tool used for processing Tigrigna documents into Latin representations (Worku, 2009). It ensures the integrity format and contents of Tigrigna documents and can be easily represented in all computer mediums. The reason for translating Tigrigna document into Suitable representation is that due to the large ASCII size. Generally, the standard ASCII uses 7-bit which requires 128 addresses to assign letters whereas Tigrigna requires 9-bits ASCII size systems which need 360 addresses. So the SERA translates the 9-bits ASCII size in to 7-bits ASCII size which can be easily represented in a computers medium. SERA is used in the research to translate Nyala Unicode format of Tigrigna script to Latin Script for processing of Tigrigna text documents.

2.5 Related Literature Works

Different researches have been done for categorizing text documents in well organized manner (Tikk et al., 2001). Most of the text categorization had applied unsupervised and supervised learning methods independently. The commonly used techniques include k-means, repeated bisection, nearest neighbor classifiers, Bayesian classifiers, decision trees, and support vector machine. On the other hand, text categorization that combines text classification and clustering is an emerging topic of text categorization (Slonim, 2001).

Different research has been conducted on Semitic language text categorization. For example, Duwairi (2007) has conducted a research on Arabic text categorization using three classifiers; namely, naïve Bayes, k-nearest neighbors, and distance-based classifiers. The performance of the classifier has been measured using recall, precision, error rate and fallout. The experimentation made on in-house collected Arabic text and result shows that the naïve Bayes classifier performs better than other two.

There are different text categorization researches done worldwide that combine clustering and classification approaches. For instances, Kyriakopoulou (2008) has conducted a text categorization research for spam detection that uses clustering as a feature extraction method. Here, features are clustered into groups based on selected clustering criteria. The important of the clustered features are evaluated and the irrelevant features are removed. The relevant features obtained from text clustering are used for classification task. As a result, the results show substantial improvements on classification performance. However, there is no any local research done using clustering and classification techniques for text categorization purposes.

Different researches have been done in Ethiopia in different research institutions for news items classification. Most of the researches are conducted in Amharic news classification, including (Zelalem 2001; Surafel 2003; Yohannes 2007; Worku 2009; Alemu 2010).

Zelalem (2001) has conducted a research on Amharic news classification. This research uses a statistical method for automatically classifying the Amharic news. Stop-words and

rarely occurring words are removed from the collection. The revenant terms are stemmed using a simple depluralization. The representatives of the documents were indentified using tf×idf weighting techniques. Centroid vectors are generated by computing the average value of the document vectors. So the learning data set was generated in centroid vectors. The cosine similarity was used to automatically classify the test set. 273 out of 321 news items were used for training of the specified classifiers and the remaining for testing purposes. The researcher used a three categories classifier and achieved 85.05% accuracy level. This performance is promising for a fewer number of data set. However, this performance may decrease when a number of documents and categories increase. There is a need to use other machine learning techniques for automatically classifying large data sets.

Surafel (2003) has conducted an experiment on the 11,024 Amharic news articles. In the experiments text preparation and preprocessing was done. Stop-words and rarely occurring words are removed from the collection. Sixty seven percent of the data was used for training purposes and the remaining thirty-three percent was used for testing purposes. A naïve Bayes and k-nearest neighbor learning algorithms were used to categorize the Amharic news items. The results of the experiment shows that best result achieved by naïve Bayes and k-nearest neighbor is on three categories (95.80% vs. 89.61%) and the least performance is obtained in 16 categories (78.48% vs. 64.50%). The reason for the drop of performance when the number of category increases is that when the number of category increases the categories contain unevenly distributed data set. The researcher also reported that the naïve Bayes classifier has high performance for automatic Amharic news categorization.

Yohannes (2007) has conducted a research on 69,684 Amharic news items. The researcher uses a text classification tools called WEKA. He measures the performance using the Logic Model Tree (LMT) and Support Vector Machine (LibSVM). The LMT is a type of decision tree learning algorithms available in WEKA tool. Both LMT and LibSVM classifier showed good performance accuracy; 79.72% and 81.15% in the 15

news categories respectively. However, the time required to build a model is very due to the large number of data set of the experiment.

Worku (2009) has also conducted research on Amharic news classification with the aim of classifying Amharic text news automatically using learning vector quantization. He first applied text preprocessing techniques such tokenization, removing stop-words, and stemming. The remaining terms are organized according their frequency. The result shows that the term frequency weighting scheme outperforms term frequency ×inverse document frequency weighting scheme in most of the categories.

Alemu (2010) attempts a hierarchical classification of Amharic news items using support vector machine. The research has conducted on the aim of constructing hierarchical classifier and it has evaluated the performance of the hierarchical classifier over the flat classifier with same data set. The result of the experiment shows that the performance of the classifier increases as it moves down through the hierarchy. Besides, the hierarchical classifier out performs the flat classifiers with same data set.

There is no standard local text documents prepared for text categorization purposes except the ENA Amharic news corpus. As a result, all the above researchers use a categorical (labeled) data available at from Ethiopian News Agency (ENA) for their training and testing the performance of Amharic news classification systems. However, in a real life there are diversified document collections available in Tigrigna language that are written by various governmental, nongovernmental and research institutions. Most of these documents are unlabeled. So conducting a text categorization research from the unlabeled data requires the application of unsupervised learning. Hence, the current study attempts to design a two step approach that first apply text clustering to obtain training set from the unlabeled data and then classification techniques for categorizing Tigrigna text documents.

CHAPTER THREE

TECHNIQUES FOR TIGRIGNA TEXT CATEGORIZATION

To design Tigrigna text categorization system, different techniques, and tools are used for preprocessing, document clustering, and classifier model building. In order to preprocess the Tigrigna documents, different text preprocessing techniques such as tokenization, stemming, and stop word removal are used. There are different document clustering and classifier model building algorithms used for categorizing Tigrigna documents. To implement the document clustering and classification algorithms, clustering and classification tool kits such as gCluto and Weka are available to facilitate the text categorization processes.

3.1 Text Preprocessing

Text preprocessing is crucial step for the subsequent text clustering and classification tasks. During text preprocessing, there are a sequence of steps applied to generate content-bearing terms and assigns weights that shows their importance for representing the document they identified from.

First, tokenization is performed which attempts to identify words in the document corpus. The common method of representing the document text is using the bag of words approaches where the word from the document corresponds to a feature and the documents are considered as a feature vector (Kiritchenko, 2006). This indicates that the words can only discriminate the given documents in the categorization process. So, the punctuation marks and digits are irrelevant component of the text documents. In the present study, the punctuation marks and digits are removed and replaced with space.

The words are considered as relevant for the given documents and they are separated by space. Algorithm 3.1 illustrates the tokenization process of the present study. First, the content of file is read line by line. Second, split them by space in to list of words. Third, check whether the word within the list contains punctuation marks of Tigrigna language; if punctuation marks exist within the word replace it with space. This step continues until end of line reaches.

Algorithm 3.1: Removing punctuation marks

```
Open the file for processing
Do
        Read the content of the file line by line
        Assign the content to string
        For word in string split by space
                If word contains punctuation marks
                        Replace punctuation marks with space
        End for
While end file
```

The above algorithm tokenizes the text documents as follows: first, the content of file is read line by line. Second, split them by space in to list of words. Third, check whether the word within the list contains punctuation marks of Tigrigna language; if punctuation marks exist within the word replace it with space. This step continues until end of line reaches. Similarly, digits are also removed using the python built in function called "sub" which takes a digits symbol "d+" as argument and removes digits from the whole list of words.

Tigrigna language has compound words written in different format. Mostly space and hyphens are used to separate them. When the hyphen is used the two words are treated as one word. However, when they are separated by space their meaning differ (Leslau, 1998). For example, "ቤት ትምህርቲ", "ስነ ስርዓት", "መራሕቲ ስድራ", and "ኣብያተ ፅሕፈት" are compound words separated by space. However, words "ቤት", "ስነ", "መራሕቲ", and "ኣብያተ" do not have meaning when they are used separately. This creates problem in text categorization process. Hence, this study prepares the compound words list file and combine them using algorithm 3.2 as follows.

```
Algorithm 3.2: Combining Tigrigna compound words
Open the file for processing
Do
        Read the content of the file line by line
        Assign the content to string
        For word in string split by space
                If word in compound word list
                        Combine first word with next word
        End for
While end file
```

Tokenization and combining compound words result in a bunch of words of Tigrigna language. They need further processing to remove stop words and apply stemming for grouping similar words together. Tigrigna language has words with the same root and written in different form. So the stemming uses to stem words of different form to their root words. In this study, the researcher stems the suffix and prefix of the term by identifying them in the given documents. The suffix is written at the end of the root word while the prefix is placed before the root word. Both prefix and suffix modify the meaning of the word.

The stemming process used in this study is based on the affix removal algorithm developed for Tigrigna language sentences (Girma, 2001). Generally, the procedure takes the word from the file as an input. The algorithm has both suffix removal and prefix removal. The prefix removal algorithm as shown in algorithm 3.3 below, it checks whether the word starts with the prefix list or not. If the word starts with the prefix list, it replaces the prefix of the word with space. If not it continues processing the text.

```
Algorithm 3.3: Prefix striping
Read prefix list file
Open the file for processing
Do
        Read the content of the file line by line
        Assign the content to string
        For word in string split by space
            If length of word is greater than two
                If word starts with prefix
                    Remove prefix from the word
                Else
                    Continue
            Else
                Continue
        End for
While end file
```

Like the prefix striping, the suffix striping algorithm shown in algorithm 3.4 also strips the suffix if the word ends with it.

```
Algorithm 3.4: Suffix striping
Read suffix list file
Open the file for processing
Do
        Read the content of the file line by line
        Assign the content to string
        For word in string split by space
            If length of word is greater than two
                If word ends with suffix
                    Remove suffix from the word
                Else
                    Continue
            Else
                Continue
        End for
While end file
```

As shown in algorithm 3.3 and 3.4, the affix removal algorithm checks the existence the prefix and suffix and then removes them from the word. The prefix striping algorithm removes the prefix which is placed before the root word. For example "ስለዝገበረ" (slez-gebere) contains "ስለዝ" (slez-) prefix. As a result, it is stemmed into "ገበረ" (gebere). Similarly the suffix striping algorithm 3.4 removes the suffix which is written at the end of the root word. For instance "ገዛውቲ" (geza-wti) has "ውቲ" (-wti) suffix at the end. So it is stemmed into "ገዛ" (geza). In Tigrigna language, prefix and suffix also exists in a single word. For example the word "መወርወርያ" (mewerwerya) has a prefix "መ" (me-) and suffix "ያ" (-ya). First, the prefix "መ" (me-) is removed using algorithm 3.3. As a result, the word "መወርወርያ" (mewerwerya) is stemmed into "ወርወርያ" (werwerya) but it has a suffix "ያ" (-ya). Then the suffix "ያ" (-ya) is removed from "ወርወርያ" (werwerya) and it is stemmed into "ወርወር" (werwer) using algorithm 3.4. As a result, the stemming helps to define words in the same context with the same term and consequently reduce their dimensionality.

Tigrigna language has a very frequently appearing terms which are called stop words. These terms that do not discriminate one document from other documents. The stop words are identified after stemming because some stop words exist in the Tigrigna text documents in their affix form. So stemming converts these terms in to their root terms. As shown in table 3.1 below, the Tigrigna stop words do not discriminate documents meaning but they provide structure in the language. These are proposition like "ናብ" (to), pronouns such as "ኣነ" (I), and verb to be like "እዮም" (are), etc.

Stop word	Meaning
ናብ	To
ኣብ	At
ኣነ	I
እዮም	Are
እቲ	The
ንስኺ	You

Table 3. 1: Sample stop word lists of Tigrigna documents

In the present study, stop words are identified manually by consulting books and dictionaries of the Tigrigna languages. The consulted books and dictionaries help to identify preposition, conjunction, articles, pronoun and auxiliary verbs of the Tigrigna language. After identifying the stop words in the Tigrigna documents, algorithm 3.5 remove the stop from the document corpus.

```
Algorithm 3. 5: Stop word removal
Read stop word list file
Open the file for processing
Do
        Read the content of the file line by line
        Assign the content to string
        For word in string split by space
                If word in stop word list
                        Remove word from the index term
                Else
                        Continue
                End if
        End for
While end file
```

As shown in algorithm 3.5 above, the system reads the list of stop word from stop word list file and iteratively removes them from the collection of index terms.

3.2 Term weighting

After preprocessing the Tigrigna text documents, the relevance of a word to the topic of a document is measured using the term weighting. So the term weighting measures the importance of the term to represent the given documents and is proportional to the number of times term appears in the document. Here, the documents are represented in a vector space d as shown in equation 3.1 below (Ozgur, 2004).

$$d = (w1, w2, \ldots, w|T|) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad 3.1$$

Where w_i is the weight of i^{th} term of document d and |T| is the number of unique terms in the document collection. There are different term approaches and most of them are based on the following characteristics (Mladenic, 1998):

- The importance of a word to the given documents is proportional to the number of times it exists in the documents.
- If the word appears in most of the documents, its discriminating power between documents is less.

The most common term weighting approaches used in text categorization are Boolean weighting, term frequency weighting, and term frequency × inverse document frequency weighting (Buckley et al, 1998). Boolean weighting is the simplest method of term weighting and it assigns 1(existence) if the term exists in a document or zero (absence) if the term does not appear in the document. Here, the weighting is only existence or absence that does not show in which documents the term appears more or less. Due to this, it is not widely used term weighting approaches.

Term frequency weighting counts the appearance of the term in documents. In this term weighting, the weight of a term in a document is equal to the number of times the term appears in the document. Sometimes, the most frequent term could not discriminate one document from other documents. If the term frequency of the term is high, its discriminating power to the mean documents is low. So, this term weighting techniques is not mostly used in text categorization processes. As a result, the researcher uses the term frequency × inverse document frequency weighting (tf×idf) which uses the frequency of the most discriminating term in a given documents. Here, the weight of term i in document d is proportional to the number of times the term appears in the document and inverse proportional to the number of documents in the corpus in which the term appears.

$$Wt_{ij} = tf_{ij} \times \log\left(N/N_i\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad 3.2$$

Equation 3.2 indicates that tf_{ij} is the term frequency term i in a document j, $\log\left(N/N_i\right)$ is the inverse document frequency of the term, N is the total document number in the corpus, and N_i is the number of documents the term appears. The tf×idf weighting approach weights the frequency of a term in a given document with a factor that discounts its important if it exists in most of the documents.

3.3 Dimensionality Reduction

Even though each word is stemmed to their root, they are still in high dimensionality. Reducing the dimension of the corpus is important in developing text categorization systems. There are various methods applied for dimensionality reduction in text categorization. The most widely used are information gain and document frequency (Ozgur, 2004).

In the present study, document frequency (DF) is used for dimensionality reduction of the given documents of the data set. The document frequency (DF) of a term shows the number of documents containing that term. The DF of each unique term is computed and terms appear in three or less than three documents are eliminated from the document corpus.

So the document number 3 is a threshold in the current study. The term appears only three documents or less is considered as a rarely appearing terms because they do not discriminate the content of the documents. Hence, they do not have contribution for text categorization and removing them lead to improvement in categorization accuracy.

3.4 Document Similarity Measure

In any text categorization process, a similarity measure between documents must be defined. In this study, the researcher uses the cosine similarity to measure the similarity of the documents. According to Karypis et al. (2004), the cosine similarity of the documents represented by vectors A and B can be defined as follows:

$$\cos(\theta) = \frac{A.B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.3$$

Equation 3.3 shows that the similarity of two documents is the cosine of the angle between the documents vectors. Here, the A and B are the term frequency × inverse document vectors of the documents. The resulting similarity ranges from −1 meaning exactly opposite, to 1 meaning exactly the same, 0 usually indicating independence, and between these values indicates intermediate similarity or dissimilarity.

3.5 Document Clustering

Document clustering is used to discover natural groups in data set without having any background knowledge of the characteristics of the data in the documents. There are different document clustering algorithms. They are mainly divided in to hierarchical and partitioning clustering algorithms (Matthew, 2004). Karypis et al. (2004) has compared partitioning and hierarchical methods of text clustering on a broad variety of test data set. It concludes that k-means of the partitioning clustering algorithm clearly outperforms the hierarchical methods with respect to clustering quality. In addition, a variant of k-means called repeated bisection k-means is introduced and yields even better performance. As a result, the present study uses partitioning clustering algorithms to cluster the Tigrigna documents in to different categories. The researcher uses the repeated bisection k-means and direct (basic) k-means of the partitioning clustering algorithms. The repeated bisection clustering algorithm clusters a set of documents using a sequence of repeated bisection and the direct (basic) k-means clustering algorithm clusters a set of documents directly in to a set of groups (clusters). The direct (basic) k-means clustering algorithm performs the clustering process as follows as shown in algorithm 3.6 by selecting k point of the cluster as initial centroids Karypis et al. (2004).

Algorithm 3.6 : Basic(direct) K-means clustering algorithms

1. Select k points as the initial centroids
2. Assign all points to the closest centroids
3. Recompute the centroid of each cluster
4. Repeat steps 2 &3 until the centroids don't change

The repeated bisection clustering algorithm starts by considering all the Tigrigna documents as a single cluster and works as shown in algorithm 3.7 below.

Algorithm 3.7: Repeated bisection clustering algorithms

1. Select a cluster to split
2. Find 2 sub-clusters using the basic K-means algorithm
3. Repeat step 2 which is called bisecting step, for a certain number of times and take the split that produces the clustering with the highest overall similarity
4. Repeat steps 1, 2 and 3 until the desired number of clusters is reached

Both the direct (basic) and repeated bisection K-means clustering algorithm use I_2 particular clustering criterion function in finding cluster. Where, I_2 is given by:

$$\text{maximize} \sum_{j=1}^{k} \sqrt{\sum_{u,v \in Ci} \text{sim}(u,v)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.4$$

In the above equation, *k* is the total number of clusters, *C* is the total objects to be clustered, *Ci* is the set of objects assigned to the *i*th cluster, *nj* is the number of objects in the *j* th cluster, *v* and *u* represent two objects, and sim *(v, u)* is the similarity between two objects.

3.6 Classification

The clustered documents are used to classify Tigrigna documents using the support vector machine and decision tree classifiers. As a result, the support vector machine and decision tree classification scheme constructs a model from the cluster categories of the training collection.

A decision tree classifier predicts the target value of the class based on various attributes of the data set. The decision tree classifier has internal and terminal (leaf) nodes. The internal node indicates the different attributes of the text classification, and the leaf nodes show the classification of the attributes. The researcher uses the j48 decision tree classifier for classifying the Tigrigna documents data set. The J48 decision tree classifier classifies a given data set using algorithm 3.8 below.

Algorithm 3.8: J48 decision tree classifier algorithm

1) Create a decision tree based on the attribute values of the available training data.
2) Identify the attribute that discriminates the training data sets.
3) Based on the identified attributes, classify the given data sets in to different classes (categories).
4) If there is any value for which the data instances are falling within its category that has the same value for the given class, then it terminates that branch and assigns to it the class until all the data instances are assigned to their class.

The core idea behind the SVM classifier is to fit the linear model to the mapped training data by maximizing the margin of the classifier (Ramirez, 2008). This shows that the value of the given parameter of hyper plane to the nearest training patterns from given classes is maximized as many training patterns as possible. In this study, the researcher classifies the Tigrigna documents using sequential minimal optimization (SMO) support vector machine classifiers. It has advantage over the other support vector machine classifiers (John, 1998). First, the amount of memory required for SMO is linear in SMO breaks the large quadratic programming problem into a series of smallest possible quadratic programming problems. The training set size which allows SMO to handle very large training sets. Second, it avoids matrix computation and this makes SMO to scale somewhere between linear and quadratic in the training set size for various test problems. Finally, it is the fastest classifier for linear SVMs and sparse data sets. Generally, the SMO algorithm involves three important components. These are an analytic solution to the optimization problem for the two chosen multipliers; a heuristic for choosing the two multipliers to optimize at a given step; and a step to compute the offset.

3.7 Tools

In current study, the researcher uses the gCluto 1.2 tool for Tigrigna text clustering and Weka 3.6.4 for the classification task.

3.7.1 Graphical Clustering Toolkits (gCluto 1.2)

gCluto is a standalone clustering software package designed to ease the use of clustering algorithms and their results. According to Matthew (2004), gCluto offers improvements over existing clustering tools with features such as an intuitive graphical user interface, interactive visualizations, and mechanisms for comparing multiple clustering solutions.

Karypis et al. (2004) argued that gCluto has an advantage that makes clustering practical for a wide variety of applications. First, it provides an array of clustering algorithms and options through the use its clustering library. This library provides highly optimized implementations of agglomerative, k-means, and graph clustering, especially in the context of sparse high-dimensional data. Second, it helps the user sort through the algorithm options and resulting data files by providing an intuitive graphical interface.

The main strength of gCluto is its ability to organize the user's data and work-flow in a way that eases the process of data analysis. This work-flow often consists of a sequence of stages which are importing and preparing data, selecting clustering options, interpreting solution reports, and concluding with visualization. Each stage of the process demands decisions to be made by the user that can alter the course of data analysis. Consequently, the user may want to backtrack to previous stages and create a new branch of analysis.

gCluto assists these types of work-flows by introducing the concept of a project. A project manages the various data files, solutions reports, and visualizations that the user generates by storing and presenting them in a single container. The work-flow of a user begins by creating a new project. gCluto will create a new directory to hold all project related files as well as a new empty project tree. Next the user imports one or more related data set. The data sets are represented by icons that appear directly under the project tree's root. After importing, it clusters the data set to produce a clustering solution. For each solution, a solution report is generated which contains statistics about the clustering. Clustering solutions are presented by an 'S' icon and are placed under the clustered data set's icon in the project tree. The work-flow concludes with interpreting a solution using one or more visualizations. In addition, gCluto allows exporting of solutions, data sets and printing of visualizations to standard formats for external use. The exported results can be used for other tools as an input of their processing purposes.

Based on the high dimensional nature of the research data and the gCluto capacity of statistics and unique visualization for interpreting clustering results, the researcher uses this tool with its wide range of options and factors that are involved in clustering processes.

3.7.2 Waikato Environment for Knowledge Analysis (Weka 3.6.4)

Weka (Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, developed at the University of Waikato, New Zealand (Witten and Frank, 2005). It is free software available under the GNU General Public License. It supports several standard text categorization tasks such as data preprocessing,

clustering, classification etc. All of the Weka techniques are predicated on the assumption that the data is available as a single flat file or relation which each data point is described by a fixed number of attributes (numeric or nominal attributes).

The Weka main user interface is the explorer. The same functionality can be accessed through the component-based knowledge flow interface and the command line. There is also the experimenter, which allows the systematic comparison of the predictive performance of the Weka machine learning algorithms on a collection of data sets.

The explorer interface has several panels that give access to the main components of text classification processes:

- The preprocess panel has facilities for importing data from a comma separated value (CSV) file and for preprocessing this data using a filtering algorithm. These filters can be used to transform the data and make it possible to delete instances and attributes according to specific criteria.
- The classify panel enables the user to apply classification algorithms (in Weka called classifiers) to the resulting data set, to estimate the accuracy of the resulting predictive model, and to visualize erroneous predictions.

Weka supports different classification schemes such as decision trees, naive bayes, support vector machine, etc. In order to classify the Tigrigna documents in to different categories, the researcher imports the clustered document records in ARFF format, preprocess them using the appropriate filtering algorithms, and classify the preprocessed document text in to different categories. Finally, the detail accuracy is measured using the standard performance measurement techniques.

3.8 Performance Measures

In the present study, the performance of the clustering and classification results is measured using different evaluation measures. The performance of any clustering algorithm is dependent on the quality of the produced results (Zhao, 2002). The two common performance measures in evaluating any clustering algorithms are entropy and purity. Entropy shows that the best clustering results is the one that leads to clusters that

contain documents from a single class, in which the entropy is zero. Here, the better clustered result is assumed when the entropy value is smaller. Entropy measures how various classes of documents are distributed within each cluster of data set. In order to measure entropy, the class distribution is measured for each cluster and then this class distribution is used to calculate the entropy for each cluster as indicated in equation 3.5.

$$E_j = -\sum_i p_{ij} \log\left(p_{ij}\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.5$$

Here, the p_{ij} is the probability of the member of cluster j belongs to class i and E_j is the entropy of individual clusters. In addition, the entropy of all produced clusters in the given clustering algorithm is calculated as the sum of the individual cluster entropies weighted according to the cluster size, and as defined in equation 3.6.

$$E_{SC} = \sum_{j=1}^{m} \frac{n_{j \times E_j}}{n} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.6$$

Where n_j is the size of cluster j, n is the total number of documents, m is the number of clusters, and E_{sc} is the entropy of all clusters.

Purity measures the performance of clustering algorithms when each cluster contained documents from primarily one class. As a result, it measures the largest class for each cluster. Here, the better clustered result is assumed when the purity value is larger. Like the entropy, the purity of each cluster and purity all produced clusters is calculated in equation 3.7 and equation 3.8 respectively.

$$P_{(S_r)} = \frac{1}{n_r} \max\left(n_r^1\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.7$$

Equation 3.8 shows that S_r is a particular cluster of size n_r, $P_{(S_r)}$ is the individual cluster purity. The purity of all produced clusters is also computed as a weighted sum of the individual cluster purities and is defined as follows:

$$\text{Purity} = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.8$$

Where n_r is the size of cluster r, n is the total number of documents, k is the number of clusters, and Purity is the purity of all clusters. Finally, a perfect clustering solution is the

one that leads to clusters that contain documents from only a single class, in which case the entropy is zero, and the purity is 1.

In addition to clustering, the performance of the classification is analyzed to measure the accuracy of the classifiers in categorizing the Tigrigna documents in to specified categories. Accuracy refers to the percentage of correct predictions made by the model when compared with the actual classifications (Baeza-Yates and Ribeiro-Neto, 1999). Besides, recall, precision, and F-measures of the classifier are also measured. Precision (P) measures the percentage of documents assigned to category c that are correctly assigned to category c. More formally, the precision is defined as shown in equation 3.9. More formally, precision is defined in equation 3.9

$$P_i = \frac{TP_i}{TP_i + FP_i} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.9$$

On the other hand, recall measures the percentage of total documents that are assigned to category c. It is defined as:

$$R_i = \frac{TP_i}{TP_i + FN_i} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.10$$

Where TP_i (i.e., true positives) is the number of documents assigned correctly to category c_i, FP_i (i.e., false positives) is the number of documents assigned to category c_i that should have been assigned to other categories, and FN_i (i.e., false negatives) is the number of documents assigned to other categories that should have been assigned to category c_i.

The F-measure (F) is the harmonic average of precision and recall, and is defined as:

$$F_i = \frac{2 \times P_i \times R_i}{P_i + R_i} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.11$$

Where P_i and R_i are the precision and recall for category c_i respectively.

CHAPTER FOUR

EXPERIMENT AND EVALUATION

This study uses unlabeled text documents collected from different sources for categorizing Tigrigna text documents. It follows a two step approach to categorize the Tigrigna documents into their proper categories. The first step clusters the Tigrigna text documents in order to find documents natural group. As a result, the Tigrigna text documents are clustered using the repeated bisection and direct k-means clustering algorithms. The results of the clustering are measured using purity and entropy. The second step uses different text classifier algorithms to predict the documents to their predefined categories. The clustered data sets are used for training the text classifiers. Hence, text classification model is developed from the training data sets using decision tree and support vector machine classifiers. Here after, the developed model is evaluated using the test data sets. Finally, the system comes with the category of Tigrigna text documents.

4.1 Proposed System Architecture

In this study, the proposed text categorization system is developed in four stages as shown in figure 4.1. These are preprocessing, clustering, classifier training, and testing the text categorization system. The preprocessing makes the raw data ready for the experiment. In this stage the irrelevant terms are removed from the documents and words of the same context with different forms are converted into the same word. The final goal of this stage is to convert the collection of text in to matrix of index terms with their tf×idf weight values.

In the experimentation, cluster of documents are created for training data set. The resulting clustering solution is used to train a text classification model. The developed model is tested using test data set. Finally, the system comes up with categories of Tigrigna text documents.

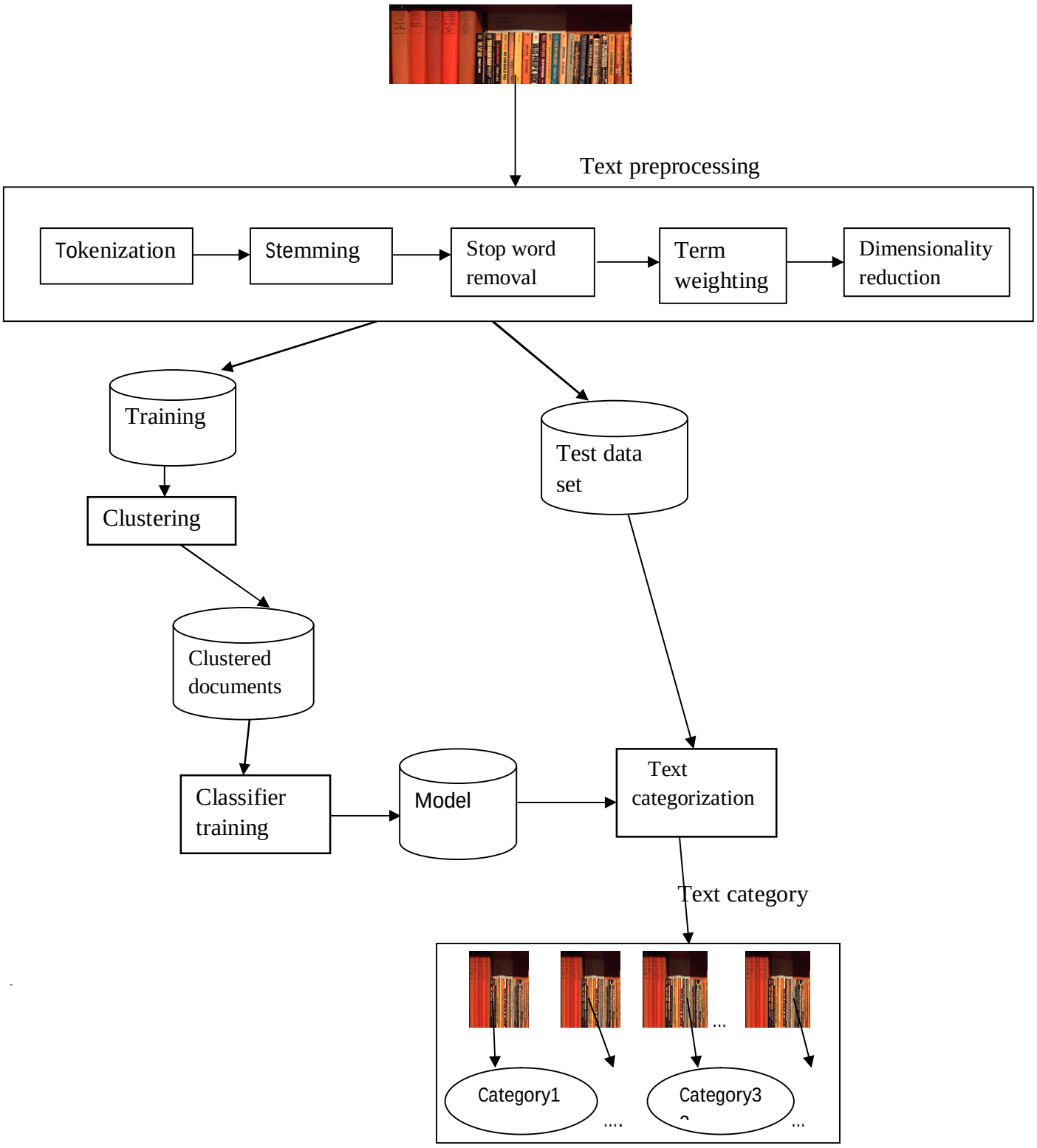

Figure 4. 1: Architecture of Tigrigna Text Categorization System

4.2 Data Set for Text Categorization

There is no Tigrigna corpus that contains Tigrigna documents with their specified categories. As a result, it is difficult to obtain Tigrigna documents with their specific categories. However, the researcher has collected the Tigrigna documents from different sources such as Education Bureau of Tigray Region, Capacity building of Tigray Region, and different Tigrigna news media such as Demtsi Woyne Tigray, and Mekalh Tigray. One thousand five hundred eighty seven (1587) documents are collected from the above sources. Out of these 1462 (one thousand four hundred sixty two) unlabeled Tigrigna text documents are used as training data set for developing model for the categorization task. 125 Tigrigna text documents are used to test the categorization systems developed from the unlabeled instances. Data cleaning is performed before applying the text categorization process. Since the aim of this research is to design Tigrigna text categorization system, documents written in bilingual language are removed manually from the data set. However, the manual cleaning is very difficult for large amount of data set. As a result, an automatic mechanism is required for handling documents written in bilingual language.

Tigrigna characters are translated in to Latin scripts for processing purposes. In this study, the researcher uses the SERA system that converts the Tigrigna characters in to ASCII. The Tigrigna language has 35 unique characters. However, Latin script has only 26 unique characters. As a result, there are no enough Latin scripts that represent the Tigrigna characters. Due to this reason, Latin words are used to represent the Tigrigna characters. For instance, the letter h and H is used to represent two different characters "ሀ" and "ሐ" respectively.

In the current Tigrigna literature, there is a unique sound for each Tigrigna alphabet. However, the old literature had three letters that have had the same sound but written in different forms. Since the current research data sets contain some old literature documents, the researcher has converted these letters to the same Latin words. For instance, the letter "ጸ" and "ፀ" have converted both to Se and their corresponding letters are also converted to same Latin representation.

4.3 Preprocessing Tigrigna Documents

In order to categorize the text documents by applying clustering and classifier algorithms, documents preprocessing is required to make ready the data set for training and testing. In this study, the Tigrigna text documents are preprocessed before using them for categorization purposes. Text documents are tokenized in to word-level data set that can be analyzed by a Machine Learning algorithm.

In tokenization stage, the Tigrigna texts are partitioned into discrete units. These units contain a list of words in the text. According to Kassa (2003), a word in Tigrigna language is a combination of two or more than two letter of Tigrigna word that has a meaning. However, it is not sufficient to split the Tigrigna text into tokens (words) for tokenization purposes because there are punctuation marks and digits embedded with each of the word. Hence, the punctuation marks and digits are tokenized from the Tigrigna corpus since they do not carry any information to describe the content.

After tokenizing, a bag-of-words are identified. However, there are terms of the same root written in different forms due to their grammatical use. In Tigrigna language one term has suffix, prefix and infix. These extra characters added to form word variants are stemmed from the document corpus using the stemming algorithm developed by Girma (2001). Though added extra characters are stemmed to come with the root of a word, there are also challenges because of affixes that change the structure of the term. For instance, words “ተሳሊጦም”, “ተሳሊጡ”, ”ይሳለጥ” and “ይሳለጡ” are the affixes of the word “ሰለጠ” (succede). The word “ተሳሊጦም” has both prefix (“ተ”) and suffix (“ም”) terms and it is stemmed in to “ሳሊጦ”. The second word “ተሳሊጡ” has only prefix terms “ተ” and stemmed in to “ሳሊጡ”. Both ”ይሳለጥ” and “ይሳለጡ” have “ይ” prefix terms and stemmed in to “ሳለጥ” and “ሳለጡ” terms. “ሳሊጦ”, “ሳሊጡ”, “ሳለጥ” and “ሳለጡ” require another morphological analysis of the terms in order to stem them in to their same root term “ሰለጠ”. Such problems need a better stemming algorithm that considers the various morphological structure of the Tigrigna language.

The terms that do not discriminate one document from other documents which are called stop words are identified after stemming words in to their root. Stop words are words that

frequently appear in nearly all the documents. In the current study, stop word list of common Tigrigna words are prepared and used for comparison. As a result, the term appears in the stop word lists are removed from the Tigrigna documents using python programming language.

Even though the words are stemmed to their root and stop word are removed from the documents, the Tigrigna text documents are in high dimensionality. In the current research, rarely appearing terms make the document to have a high dimensions. So, in this research words that appear in three or fewer than that are considered as rare words and hence removed them from terms that are used for document representation. Finally, term-document matrix of the relevant terms and their tf×idf with reduced dimension from 2623 size feature to 1200 size feature is created for all the Tigrigna documents which is used for text clustering and classification.

4.4 Clustering

Clustering becomes crucial in text categorization when document category is not known. This study also uses clustering as a complementary step to text classification. Since Tigrigna documents used in this study are not labeled. In order to make the data ready for clustering task, the researcher constructs the document term matrix. The rows represent the list of documents and the columns represent the list of terms. The value in the i^{th} row and j^{th} column represents the tf×idf of the term in a given documents.

The input file formats in gCLuto are sparse matrix format where the first line contains information about the size of the matrix, and the remaining N lines contain information for each row. The information for each row contains the position number of the term and its tf×idf in the given documents. As shown in figure 4.2, the first line of the input matrix file contains three numbers describe the number of rows in the matrix (n) (i.e. the number of documents in the given data set), the number of columns in the matrix (m) (i.e. the number of unique terms), and the total number of non-zero entries in the n×m matrix respectively. Next lines show the position number of the term in the document and their tf×idf value in each documents rounded to two digits.

```
1462  1200  83242
1 1.52 2 1.63 3 0.87 4 2.09 5 1.60 6 0.42 7 1.84 8 9.21 9 2.34 10 1.41 11 2.12 12 1.84 13
1.99 14 4.40 15 1.59 16 0.64 17 2.20 18 1.50 19 1.08 20 0.79 21 2.43 22 1.12 23 1.37 24
1.52 25 1.63 26 6.94 27 0.91 28 1.67 29 0.54 30 2.04 31 1.35 32 1.20 33 1.91 34 1.47 35
0.88 36 1.82 37 1.25
3 0.87 38 1.27 39 1.62 40 2.25 41 3.61 42 0.63 43 1.22 44 2.43 45 6.81 46 2.97
47 1.67 3 0.87 6 0.42 48 1.59 49 1.10 50 1.82 51 1.23 52 1.31 16 0.64 53 1.91 54 1.86 55
0.28 23 1.37 56 1.30 26 2.31 57 1.77 37 1.25
58 1.80 59 0.71 60 1.14 61 2.12 62 1.52 63 0.94 64 1.09 65 2.77 26 3.47 29 0.54 30 2.04
31 1.35 66 2.05 67 2.27 57 1.77
68 1.50 69 0.74 3 1.75 6 0.42 70 1.37 51 1.23 71 1.50 16 0.64 72 1.89 53 1.91 73 1.51 74
0.90 75 1.34 26 1.16 28 0.83 31 1.35 76 1.78
.  .  .
```

Figure 4. 2: Input file format for gCluto clustering tools

The gCluto clustering tool supports a number of clustering algorithms, including repeated bisection, and direct k-means which are used in the present study. The results for each of the clustering algorithms are presented in the experiment. The clustering algorithms cluster the Tigrigna text documents in to seven, eight, nine, ten, eleven and twelve subject categories. The performance of each subject category is measured. The eight subject categories result high purity and low entropy. So the researcher uses the eight subject categories for this study. The performance of the given algorithm is inversely proportional with entropy and directly proportional with purity. The clustering results of the clustering algorithms are also analyzed using mountain visualization (Zhao, 2002).

4.4.1 Repeated bisection clustering algorithm

The results for repeated bisection clustering algorithms are presented in table 4.1. This result contains statistics about the discovered clusters.

Cluster	Size	ISim	ISdev	ESim	ESdev	Entropy	Purity
0	86	0.535	0.156	0.015	0.003	0.053	0.977
1	111	0.107	0.041	0.011	0.003	0.412	0.703
2	118	0.076	0.028	0.013	0.004	0.431	0.763
3	167	0.050	0.019	0.009	0.004	0.713	0.311
4	196	0.050	0.017	0.011	0.004	0.441	0.755
5	243	0.049	0.019	0.011	0.004	0.595	0.638
6	327	0.043	0.017	0.014	0.006	0.761	0.398
7	214	0.032	0.011	0.012	0.004	0.819	0.322

Table 4. 1: Experimental results using repeated bisection clustering algorithm

The clustering statistics shows that cluster “7” and cluster “3” are less accurate than the other cluster because they have high entropy and less purity as shown in table 4.1 while cluster “0” has highest accuracy than the other clusters because it has low entropy (i.e. 0.053) and high purity (i.e. 0.977). As a result, the repeated bisection clustering algorithm gives a better performance of the cluster “0”, followed by cluster “2”, “4”, “1”, and ”5” while cluster ”6”,”7”,and “3” have least performance in repeated bisection clustering algorithm as elaborated in table 4.1. Results are also visualized using mountain visualization in figure 4.3 below.

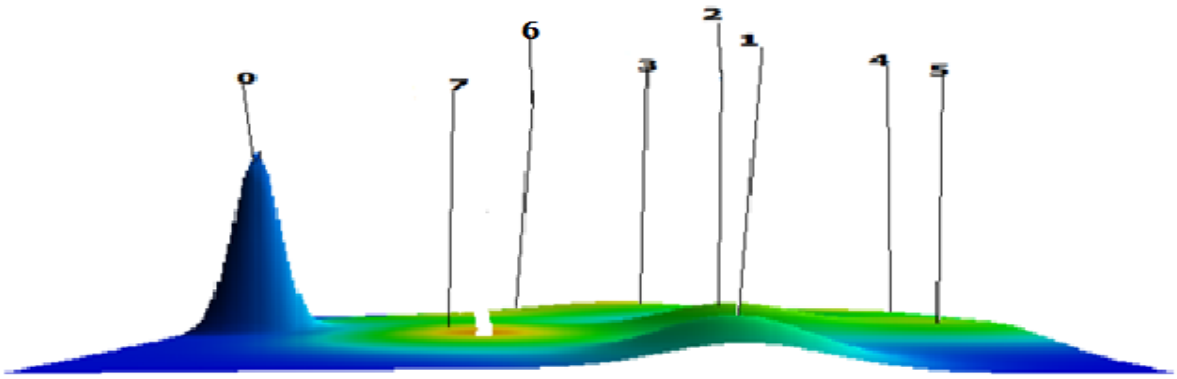

Figure 4. 3: Repeated bisection clustering algorithm results visualization

As shown in figure 4.3, each peak represents a single cluster in the clustering. The height of each peak on the plane of clustering result is proportional to the internal similarity of the corresponding cluster. As shown in table 4.1, the internal similarity ”ISim” elaborates that cluster “0” with 0.535 “ISim” has the highest similarity but cluster “7” with 0.032 “ISim” has the least internal similarity than other clusters. As a result, the height of peak in figure 4.3 also depicts accordingly that cluster “0”, cluster “1” has the highest internal similarity than other clusters whereas cluster “7” has the least internal similarity. This indicates that the documents clustered within the cluster “0” are more similar than the documents clustered within cluster “7”.

4.4.2 Direct k-means clustering algorithm

The direct (basic) k-means clustering algorithm results are also presented as shown in figure 4.2 below.

Cluster	Size	ISim	ISdev	ESim	ESdev	Entropy	Purity
0	91	0.486	0.173	0.014	0.003	0.051	0.978
1	111	0.091	0.039	0.011	0.004	0.353	0.793
2	114	0.076	0.022	0.011	0.005	0.649	0.544
3	197	0.056	0.022	0.014	0.005	0.597	0.442
4	152	0.053	0.033	0.012	0.004	0.804	0.303
5	211	0.041	0.014	0.011	0.004	0.824	0.365
6	282	0.040	0.013	0.011	0.004	0.489	0.727
7	304	0.041	0.017	0.014	0.007	0.802	0.329

Table 4. 2: Experimental results using direct k-means clustering algorithm

The direct k-means clustering algorithm statistics in table 4.2 shows that the cluster “0” is clustered more accurately than other cluster categories because it has low entropy (i.e. 0.051) and high purity (i.e. 0.978). As shown in the above table 4.2, cluster “4” is clustered less accurately than other clusters because it has low purity (i.e. 0.303). As a result, the direct k-means clustering algorithm gives a better performance of the cluster “0” followed by cluster “1”, “6”, ”2” and “3” where cluster “4”, ”5”, and ”7” have lower performances than other categories.

Like the repeated bisection clustering algorithm, direct k-means clustering algorithm results are also visualized using the mountain visualization as shown in figure 4.4 below. As a result, each cluster is represented in the figure by certain peak of the figure. As indicated in table 4.2, the cluster “0” documents has more internal similarity than other cluster documents because the cluster “0” has high internal similarity shown in the column “ISim” which is 0.486 where the cluster “5”, “6” and “7” have low internal similarity. Their internal similarities are 0.041, 0.040 and 0.041 respectively. As a result, the peak for cluster “0” is high and for cluster “5”, “6” and “7” are very low.

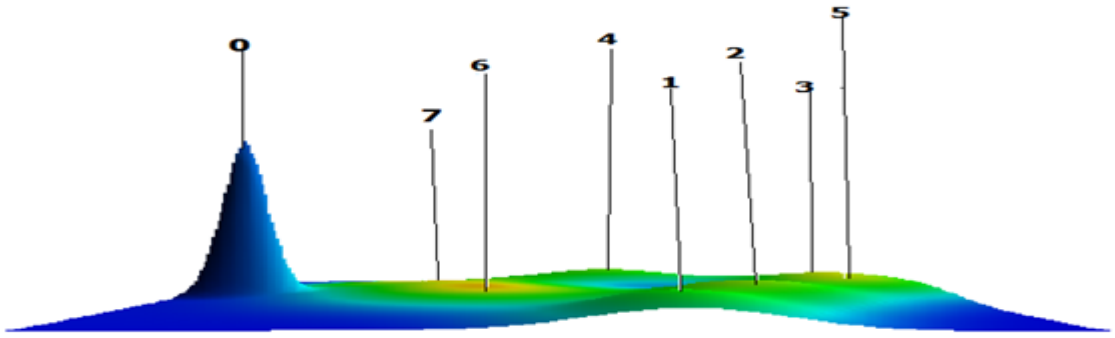

Figure 4.4: Direct k-means clustering algorithm results visualization

4.4.3 Discussion

Both repeated bisection and direct k-means clustering have high performance in cluster “0” and low performance in cluster “7”. The cluster “0” has deqeq (“ደቀቅ”), naaxte (“ኣናእሽ”), gdi (“ግዲ”), and kala (ትካላ) as descriptive and discriminating terms as shown in table 4.3. So cluster “0” documents illustrate about small enterprise, trade and company because these terms are the descriptive and discriminating terms of the cluster “0” as shown in table 4.3. These are similar topic features. Hence, the cluster “0” instances are clustered more accurately than other clusters as shown in table 4.3.

Cluster		Feature	%	Feature	%	Feature	%	Feature	%
0	Descriptive	Deqeq (ደቀቅ)	31.1%	naaxte (ኣናእሽ)	22.3%	gdi (ግዲ)	11.6%	kala (ትካላ)	3.2%
	Descriminating	Deqeq (ደቀቅ)	18.2%	naaxte (ኣናእሽ)	12.6%	gdi (ግዲ)	5.8%	kala (ትካላ)	1.7%
1	Descriptive	Hmam (ሕማም)	12.0%	Easo (ዓሶ)	9.8%	aiecyvi (ኤችኣይቪ)	9.4%	TEna (ጥዕና)	7.5%
	Descriminating	Hmam (ሕማም)	7.4%	Easo (ዓሶ)	6.3%	aiecyvi (ኤችኣይቪ)	6.1%	TEna (ጥዕና)	4.7%
2	Descriptive	tmhr (ትምህር)	31.9%	tmharo (ተምሃሮ)	9.4%	memhra (መማህራ)	5.2%	yuniversi (ዩኒቨርሲ)	4.6%
	Descriminating	tmhr (ትምህር)	22.1%	tmharo (ተምሃሮ)	6.3%	memhra (መማህራ)	3.6%	yuniversi (ዩኒቨርሲ)	3.3%
3	Descriptive	Hadega (ሓደጋ)	17.0%	meTQaE (መጥቃዕቲ)	5.4%	aaxeberti (ኣሸበርቲ)	4.7%	suda (ሱዳ)	3.6%
	Descriminating	Hadega (ሓደጋ)	11.5%	meTQaE (መጥቃዕቲ)	3.7%	aaxeberti (ኣሸበርቲ)	3.2%	suda (ሱዳ)	2.5%
4	Descriptive	mesno (መስኖ)	8.6%	hiektar (ሄክታር)	7.6%	kuntal (ኩንታል)	6.8%	hier(ሄር)	3.5%
	Descriminating	mesno (መስኖ)	6.1%	hiektar (ሄክታር)	5.6%	kuntal (ኩንታል)	5.0%	deqeq (ደቀቅ)	2.9%
5	Descriptive	priezidan (ፕረዝደን)	4.9%	parlama (ፓርላማ)	4.4%	mereSa (መረፃ)	4.2%	poletika (ፖለቲካ)	3.8%
	Descriminating	Priezidan (ፕረዝደን)	3.4%	parlama (ፓርላማ)	3.3%	mereSa (መረፃ)	3.2%	poletika (ፖለቲካ)	2.9%
6	Descriptive	frdi (ፍርዲ)	11.7%	ftHi (ፍትሒ)	6.2%	Hgi (ሕጊ)	3.6%	TrEa (ጥርዓ)	3.6%
	Descriminating	frdi (ፍርዲ)	9.7%	ftHi (ፍትሒ)	5.2%	Hgi (ሕጊ)	3.1%	TrEa (ጥርዓ)	3.0%
7	Descriptive	Sde(ስደ)	13.6%	Hbura (ሕቡራ)	9.1%	Aiertra (ኤርትራ)	9.0%	suda (ሱዳ)	6.6%
	Descriminating	sde(ስደ)	10.5%	aiertra (ኤርትራ)	7.1%	Hbura (ሕቡራ)	6.4%	suda (ሱዳ)	5.1%

Table 4. 3: Descriptive & Discriminating features of clustering algorithm

On the other hand, cluster “7” has sde (ስደ), Hbura (ሕቡራ), suda (ሱዳ), and aiertra (ኤርትራ) as descriptive and discriminating terms of this cluster as shown in table 4.3. So the cluster “7” documents elaborate about refugee, united nations, Sudan and Eritrea because these terms are the descriptive and discriminating terms of the cluster “7” as shown in table 4.3. The documents in cluster “7” discuss the political disorder of Eritrea and Sudan which leads their citizen to migration to other country. Besides, they also elaborate the help from united nation to the refugee of Eritrea and Sudan. As a result, these documents in cluster “7” discuss different topics: the political disorder of the two countries, the migration of their citizen and social aid from United Nations to migrants. Due to this the clustering algorithm faces difficulty to find common word between these concepts. This decreases the performance of the clustering algorithm.

Based on table 4.1 and 4.2 results, the total purity and entropy of the clustering algorithm is computed using the equation described in section 3.8, as depicted in table 4.4.

Clustering algorithm	Purity	Entropy
Repeated bisection	0.56	0.611
Direct k-means	0.516	0.624

Table 4. 4: Comparison of direct k-means and repeated bisection clustering algorithms

As shown in table 4.4, repeated bisection clustering algorithm has higher purity and lower entropy than the direct k-means clustering algorithm. Hence, the repeated bisection clustering algorithm is selected for training classification algorithms.

4.5 Data set preparation from the clustering results

Accordingly, training data set, which is depicted in table 4.5, is prepared with the help of repeated bisection clustering algorithm.

Category	Number of training set
0	86
1	111
2	118
3	167
4	196
5	243
6	327
7	214

Table 4. 5: Data set preparation from the clustered data sets

These data sets are used to train the classifier for building classification model for Tigrigna text categorization.

4.6 Classification

Classification helps to predict the class of documents to the predefined category. As a result, the present study uses different text classifier algorithms to predict the class of documents to their predefined categories. The results of clustering algorithms are used to classify Tigrigna text documents using Weka tool. The input file for the Weka is prepared in ARFF files format which have two distinct sections as shown in figure 4.5. The first section contains the header information, and the second section contains the data information. The header of the ARFF file contains the name of the relation (data), list of attributes (the columns in the data) and their types; while the data section shows the value of the attribute in the given documents.

```
@relation data

@attribute document
{doc1,doc2,doc3,doc4,doc5,doc6,doc7,doc8,doc9,doc10,doc11,doc12,
doc13,doc14,doc15,doc16,doc17,doc18,doc19,doc20,doc21,doc22,doc2
3,doc24,doc25,doc26,doc27,doc28,doc29,doc30,doc31,doc32,doc33,do
c34,doc35,doc36,doc37,doc38,doc39,doc40,doc41,doc42,doc43,doc44,
doc45,doc46,doc47,doc48,doc49,doc50,doc51,doc52,doc53,doc54,doc5
5,doc56,doc57,doc58,doc59,doc60,doc61,doc62,doc63,doc64,doc65,do
c66,doc67,doc68,doc69,doc70,doc71,doc72,doc73,doc74,doc75,.....,
doc1462}
@attribute Sra numeric
@attribute feTru numeric
@attribute seba numeric
@attribute meta numeric
@attribute rbu numeric
...
@attribute 'class label' {0,1,2,3,4,5,6,7}

@data
doc1,1.52,1.63,0.87,2.09,1.6,0.42,1.84,9.21,2.34,1.41,2.12,1.84,
1.99,4.4,1.59,0.64,2.2,1.5,1.08,0.79,2.43,1.12,1.37,1.52,1.63,6.
94,0.91,1.67,0.54,2.04,1.35,1.2,1.91,1.47,0.88,...,3
doc2,0,0,0.87,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,1.27,1.62,2.25,3.61,0.63,...,5
...
```

Figure 4.5: Input file format for Weka tool

The data set of the present study has two types of attributes. These are numeric attributes for the weight of the feature words and nominal attributes for the class labels and document labels. The classifiers used in this study are SVM and Decision Tree classifiers to build a classification model. However, the class labels are the cluster number of the repeated bisection clustering algorithms. Each cluster number has descriptive and discriminating features as shown in table 4.3. So the researcher uses these features to find corresponding class label for each cluster number. The cluster numbers are assigned meaningful labels based on their descriptive and discriminating features obtained from the clustering algorithm as shown in table 4.6.

Cluster	Corresponding class label
0	Trade
1	Health
2	Education
3	Accident
4	Agriculture
5	Law
6	Politics
7	Social

Table 4. 6: Labeling the cluster numbers

By using the corresponding labels of the cluster number, the results of the classifiers are interpreted.

4.6.1 Classification using decision tree classifier

Weka supports different decision tree classifiers that are used for classifying numeric and nominal attributes. In the present study, the J48 decision tree classifier is used for experimentation. This classifier requires a small amount of memory and time. The j48 classifies the 72% of 1462 instances correctly within 34.4 seconds as illustrated in table 4.11. Weka has a number of options for measuring the performance of a classifier out of them detailed accuracy by class and confusion matrix are shown below for the J48 classifier.

Trade	Health	Education	Accident	Agriculture	Law	Politics	Social	
79	1	0	1	1	0	2	2	Trade
0	96	2	3	0	2	4	4	Health
0	6	86	4	2	4	10	6	Education
0	4	4	123	1	15	8	12	Accident
2	1	4	0	139	7	17	26	Agriculture
0	4	6	22	1	176	11	23	Law
4	5	15	9	21	14	220	39	Politics
3	10	6	8	18	16	21	132	Social

Table 4. 7: Confusion matrix of the J48 classifier

Based on the above confusion matrix, the performance of the classifier is shown in table 4.8 using precision, recall, F-measure, and ROC-Area.

Class	TP Rate	FP Rate	Precision	Recall	F-measure	ROC Area
Trade	0.919	0.007	0.898	0.919	0.908	0.963
Health	0.865	0.023	0.756	0.865	0.807	0.925
Education	0.729	0.028	0.699	0.729	0.714	0.871
Accident	0.737	0.036	0.724	0.737	0.73	0.886
Agriculture	0.709	0.035	0.76	0.709	0.734	0.872
Law	0.724	0.048	0.752	0.724	0.738	0.878
Politics	0.673	0.064	0.751	0.673	0.71	0.848
Social	0.617	0.09	0.541	0.617	0.576	0.784
Weighted Avg.	0.719	0.049	0.723	0.719	0.72	0.866

Table 4. 8: Detail accuracy of J48 classifier by class

The performance of each class is computed from the confusion matrix as shown in table 4.7 by identifying the correctly classified instances diagonally against the actual number of instances in a category row-wise. As we understand from the confusion matrix in table 4.7, “trade” category registered the best accuracy of 92%, followed by “health” and “accident” with 86.5% and 73.7% accuracy. On the other hand, “social” category has least performance because of this category as shown in table 4.1 has low internal similarity. The performance of the classifier is also measured using F-measure value as shown in table 4.8. Based on the F-measure value, the J48 classifier in table 4.8 shows that the category “social” is classified less accurate than the other categories but the “trade”, ”health”, and “agriculture” are classified more accurately than other categories while the “social” category has low performance which is 57.6% F-measure. This indicates that the instances cluster in to the corresponding cluster of the “social” category have fewer words in common than others. Due to this, the “social” category is classified less accurate than other categories.

4.6.2 Classification using support vector machine

The Weka version 3.6.4 used for the experiment has different SVM (Support vector machine) classifiers. In the present study, SMO classifier has higher performance than

other support vector machine classifiers because it correctly classifies 1204 (82.4 %) out of 1462 instances as shown in table 4.11.

Trade	Health	Education	Accident	Agriculture	Law	Politics	Social	
78	0	0	0	2	0	4	2	Trade
0	88	1	5	1	5	1	10	Health
0	0	63	0	4	4	20	27	Education
0	0	0	137	1	11	14	4	Accident
0	0	4	0	157	0	15	20	Agriculture
0	0	1	8	1	212	9	12	Law
4	1	4	2	8	5	289	14	Politics
1	2	3	4	10	5	9	180	Social

Table 4. 9: Confusion matrix of SMO classifier

Based on the above confusion matrix, the performance of the classifier is shown in table 4.10 using precision, recall, F-measure and ROC-Area.

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
Trade	0.907	0.004	0.94	0.907	0.923	0.95
Health	0.793	0.002	0.967	0.793	0.871	0.966
Education	0.534	0.01	0.829	0.534	0.649	0.902
Accident	0.82	0.015	0.878	0.82	0.848	0.968
Agriculture	0.801	0.021	0.853	0.801	0.826	0.962
Law	0.872	0.025	0.876	0.872	0.874	0.963
Politics	0.884	0.063	0.801	0.884	0.84	0.945
Social	0.841	0.071	0.669	0.841	0.745	0.924
Weighted Avg.	0.824	0.034	0.833	0.824	0.823	0.948

Table 4. 10: Detail accuracy of SMO classifier by class

The performance of each class is computed from the confusion matrix as shown in table 4.9 by identifying the correctly classified instances diagonally against the actual number of instances in a category row-wise. As we understand from the confusion matrix in table 4.9, “trade” category registered the best accuracy of 90.7%, followed by politics and law with 88.4% and 87.3% accuracy. On the other hand, “education” category has least performance because of this category as shown in table 4.1 has low internal similarity.

The performance of the classifier is also measured using F-measure value as shown in table 4.8. Based on the F-measure value, the J48 classifier in table 4.10 shows that the category “social” is classified less accurate than the other categories but the “trade”, ”health”, and “agriculture” are classified more accurately than other categories while the “social” category has low performance which is 57.6% F-measure. This indicates that the instances cluster in to the corresponding cluster of the “social” category have fewer words in common than others. Due to this, the “social” category is classified less accurate than other categories.

4.6.3 Discussion

The SMO support vector machine classifier classifies 1204 instances out of 1462 where as the J48 decision tree classifier classifies 1051 instances out of 1462 as shown in table 4.11.

Instance status	J48		SMO	
Correctly Classified Instances	1051	72%	1204	82.4 %
Incorrectly Classified Instances	411	28%	258	17.6%
Time taken to build a model	34.4 seconds		32.68 seconds	

Table 4. 11: Comparison of J48 and SMO classifiers

This shows that the SMO classifier has better performance than the J48 classifier. The SMO classifier also requires less time to build the classification model as illustrated in table 4.11. Hence, this study selects the SMO classification model as the Tigrigna text categorization system due to its effectiveness and efficiency.

4.7 Testing Tigrigna Text Categorization System

A test data set is prepared to evaluate the performance of the Tigrigna text categorization system as shown in figure 4.6. They are labeled by language expert. The data set is strategically prepared to include all the 8 categories.

```
@relation 'testdata'

@attribute document
{acc1,acc2,acc3,acc4,acc5,acc6,acc7,acc8,acc9,acc10,agr1,agr2,agr3,agr4,agr6,agr7,agr8,ag
r9,agr10,agr11,agr12,...}

@attribute Sra numeric
@attribute feTru numeric
@attribute seba numeric
@attribute meta numeric
@attribute rbu numeric

...

@attribute 'class label' {}

@data
acc1,1.52,1.63,0.87,2.09,1.6,0.42,1.84,9.21,2.34,1.41,2.12,1.84,1.99,4.4,1.59,0.64,2.2,1.5
,1.08,0.79,2.43,1.12,1.37,1.52,1.63,6.94,0.91,1.67,0.54,2.04,1.35,1.2,1.91,1.47,0.88,1.82,
1.25,...,?

acc2,0,0,0.87,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1.27,1.62,2.
25,3.61,0.63,1.22,2.43,...,?

...
```

Figure 4. 6: Testing data set of Tigrigna text documents

The test data set is prepared in ARFF files format which have two distinct sections as shown in figure 4.6. The first section contains the header information, and the second section contains the data information. The header of the ARFF file contains the name of the relation (testdata), list of attributes (the columns in the data) and their types; while the data section shows the value of the attribute in the given documents. The value of the attribute as shown in figure 4.6 shows the document name followed with tf×idf values of the attributes and class label. For testing the text categorization purpose, the class label of each document is removed and replaced with "?" that shows no class label as shown in figure 4.6. To evaluate the Tigrigna text categorization system, a test data set is given to SimpleCLI interface of the Weka tool using *java weka.classifier.functions.SMO –p 1201 -l smomodel.model –T testdata.arff* command. The –p indicates the class label index number used to evaluate the test data (i.e. 1201) while –l and –T show that the classification model (i.e. smomodel.model) and test data (i.e. testdata.arff) respectively. Since the classes of the test documents are known, the researcher compares the accuracy of the classifier against "truth" by evaluating how often the classifier assigns the test documents to the classes from which they originally came. Using the SMO classification model, the performance of the system is presented in table 4.12.

Class number	Class name	Actual	Predicted	Accuracy
0	Trade	15	12	80%
1	Health	19	12	63.2%
2	Education	18	11	61%
3	Accident	10	7	70%
4	Agriculture	15	10	66.7%
5	Law	16	11	68.8%
6	Politics	16	13	81.3%
7	Social	16	12	75%
Average accuracy				70.4%

Table 4. 12: Testing results of the text categorization system

The current study predicts 70.4% instances out of 125 in to their categories correctly as illustrated in table 4.12.

4.8 Discussion

As we observe from the evaluation results, the performance of the system greatly depends on the performance of clustering algorithm. Clusters with low entropy and high purity (like “trade” category) register high classification result as compared to classes (like “social”). Hence, the corresponding cluster “3” class label “education” has 0.649 F-measure value in the classification as shown in table 4.12. This shows that the performance of the “education” is lower than other category because classifier classifies the given instances based on the common words they have. As shown in the clustering algorithm of cluster “3”, the instances of the “education” category are distributed to various classes. Hence, they have fewer common words. This leads to lower performance in the classification of the “education” category documents.

There are also mismatches observed in categorizing documents between the clustering algorithm and text classifier. As a result, there are documents that cluster in one class in clustering algorithm but the classifier classifies them in to different class. The following two examples illustrate this situation:

ኣብ ሩስያ ሓንቲ ነፋሪት ወዲቓ ኣብኣ ዝፀንሑ ኩሎም 88 ሰባት ድማ ሞይቶም።እታ ቦይንግ 737 ዝዓይነታ ነፋሪት ካብ ከተማ ሞስኮ ምስተልዓለት ከምዝወደቐት ቢቢሲ ሓቢሩ።መንቐሊ እቲ ሓደጋ ከምዘይተፈለጠን መንግስቲ እታ ሃገር ምርመራ የካይድ ዝሞቱ ሰባት ዜጋታት ኣሜሪካ ፈረንሳይን ጀርመንን ይርከብዎም።

This statement has “ወዲቓ” (crushed), and “ሓደጋ” (accident) features and the document is cluster in to an “accident” based on these features. But it has also “ምርመራ” (investigation) which states about law so that the classifier classifies this document in to “law” category. However, this documents state the accident caused due to the crushed Russian plan and the people dead in it. But it also indicates the investigation about the causes of the accident. The general meaning of this document is about “accident”. So the clustering algorithm is right with respect to these documents.

ኣብ ክልል ሶማል ሕፅረት መግቢ ንዘጋጠሞም ነበርቲ እክሊ መግብን ማይን ይቀርበሎም ከምዘሎ ቢሮ ምክልኻል ሓደጋን ውሕስነት መግብን እቲ ክልል ኣፍሊጡ።እቲ ቢሮ ንዋልታ ከምዝበሎ መንግስቲ ነበርቲ ካብ መረበቶም ከይተሰደዱ ሓገዝ ዝረክብሉ ኩነታት ኣመቻችዩ ኣሎ። ዛጊድ ንልዕሊ 1 ሚሊዮን ፀገማት እክሊ መግብን ማይን ከምዝቀረበሎም ዘዘካከረ እቲ ቢሮ ኣብ ቀረባ ብዝተገበረ መፅናዕቲ ድማ ቁፅሪ ተሓገዝቲ 1.9 ሚሊዮን ምብፅሑ ከምዝተረጋገፀ ኣፍሊጡ።ኣብቲ ከባቢ ዝርከቡ ከፍትን ጠለ በግዕን ቅድሚ ምማቶም ናብ ዕዳጋ ንከወፁ ደገፍ ይግበር ከምዘሎ ድማ እቲ ቢሮ ኣፍሊጡ።

The clustering algorithm clusters this document in to “politics” but the classifier classifies this document in “social” class. However, the general meaning of this document is about social aid given to drought people of the Somalia region. It also states about the government readiness to the social aid. So this document is not clustered correctly but it is classified in to the right class by the classifier.

The Tigrigna text documents have various morphological structure words. First, some of the Tigrigna language affixes modify the structure of the root words differently. As a result, it is difficult to stem the affixes of some words to their root word. For instance, “ሰሊጦም”, “ተሳሊጦም”, and “ይሰልጡ” are the affixes of the “ሰለጠ” (succeed). However, they are stemmed into “ሰሊጦ” and “ሰልጡ” which are different from their root terms. Such cases create similar terms to be treated differently due to the natural language ambiguity. So this increases the dimensionality of the text categorization system and considers similar documents to be treated differently. So it requires a detail morphological understanding of the Tigrigna languages in order to preprocess the Tigrigna text documents.

Tigrigna language has different dialects according to the specific places. The dialects do not follow similar rules: some of them add or remove letter to/from the word. They

modify the structure of the word. For instance, the words “ሞይቱ” (dead) and “ሞተ” (dead) are the same words but written differently due to the Tigrigna language dialect. So this increases the dimensionality of the data set in categorizing the Tigrigna text documents.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

The explosion of the World Wide Web provides a growing amount of information and data coming from different sources. As a result, it becomes more and more difficult for target users to select contents they desire. Supporting the target users to access and organize the enormous and widespread amount of information is becoming a primary issue. As a result, several online services have been proposed to find and organize valuable information needed by target users. However, these services are not capable to fully address the user's interest. Therefore, a text categorization mechanism is required for finding, filtering and managing the rapid growth of online information. Several researches have been done on text categorization especially text classification with the help of different machine learning approaches; and good results were found.

However, with the growth of text corpus the text classification using a predefined category is an extremely costly and time-consuming activity. The need for classifiers that can learn from unlabeled training samples is required. This is an area of active research and several experiments have been conducted to enhance classifiers' performance by combining supervised with unsupervised techniques. The present study has vital role in reducing the expense requires for human expert and protects error that can occur in manual labeling of the given documents into specific category. So that this study uses unlabeled text documents collected from difference sources for categorizing Tigrigna text documents. To categorize the Tigrigna documents into certain categories, the present study uses a two step approach. First, clustering is used to obtain natural grouping of the unlabeled data. As a result, the researcher uses direct k-means and repeated bisection clustering algorithm. The direct k-means clustering algorithm has 0.516 purity and 0.624 entropy while repeated bisection clustering algorithm has 0.56 purity and 0.611 entropy. Hence, the repeated bisection clustering algorithm has better performance than the direct k-means clustering algorithm. So that the researcher selects the repeated bisection clustering algorithm results to train the text classifier.

The second step uses the SMO support vector machine and j48 decision tree classifier to build a Tigrigna text categorization system. The SMO classifier correctly classifies

82.4% within 32.68 seconds whereas the J48 classifier classifies 72% within 34.4 seconds. As a result, the SMO classifier is effective and efficient in classifying the Tigrigna text documents.

One hundred twenty five instances are labeled by an expert in to eight categories in order to test the text categorization system using the selected SMO model. The system predicts 70.4% out of 125 instances correctly. The result looks promising for designing an applicable categorizer for Tigrigna language.

From the result we observe that there are documents with multiple contents in Tigrigna literature. This is a challenge for the present study; as a result of which even the clustering algorithm and classifier differ in suggesting the category of such instances. Creating ontology-based hierarchical categorization of documents may solve these challenges.

5.2 Recommendation

The following recommendations are forwarded for the future research works.

- Tokenizing Tigrigna punctuation marks in the current research is difficult because some abbreviated terms have a punctuation mark between them. These punctuation marks represent one or more than words. If such punctuation marks are removed from the document, the meaning the term will lost. So a rule is required to convert the abbreviated term to their original representation.
- Some affix modified the root term structure, so removing them will not create the root. As a result, such case should be handled by deeply understanding the morphological structure of the term. A Tigrigna stemmer is required based on a detail morphological structure of the Tigrigna language that can handle the word variant of Tigrigna language.
- A standard Tigrigna document corpus should also be prepared for the text categorization purposes.
- Even though the current study is used to classify the Tigrigna document by incorporating text clustering approach in to text classification. But a semi-supervised approach that expands the clustered data before classification is recommended.

- In the current study, there are documents cluster in one category by the clustering algorithm but text classifier classifies them in different category. This reduces the performance of the text categorization system. So it is recommended to use hierarchical text classification instead of normal text classification because the hierarchical text classification splits one category in to specific sub category based on their specific contents of the given category.
- This study categorizes Tigrigna text documents based on the words occur among them. However, most of the Tigrigna language words have multiple possible meanings that can only be determined by considering the context in which they occur. So this has an effect in categorizing the given instances in to certain categories. The researcher recommends an ontology based text categorization system that considers the context of the words in the given documents.

BIBLIOGRAPHY

Addis, A. (2010) Study and Development of Novel Techniques for Hierarchical Text Categorization. University of Cagliari, Italy.

Alemu Kumilachew (2010) Hierarchical Amharic News Text Classification. MSc Thesis. Addis Ababa University, Addis Ababa, Ethiopia.

Andrew, K., Rosenfeld, R., Tom, M. and Andrew, Y. (1998) Improving text classification by Shrinkage in a Hierarchy of Classes. Proceedings of ICML-98, 15th International Conference on Machine Learning. San Francisco, USA: Morgan Kaufmann.

Antonie, M. and Zaiane, O. (2002) Text Document Categorization by Term Association. Proceedings of the 2002 IEEE International Conference on Data Mining. Maebashi City, Japan: IEEE, pp.30-50.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999) Modern Information Retrieval. New York, USA: ACM Press/Addison-Wesley.

Baker, D. and Kachites, A. (1998) Distributional Clustering of Words for Text Classification: ACM SIGIR, pp.96-102.

Berger, H. (2004) A Comparison of Text Categorization Methods Applied to N-Gram Frequency Statistics. Proceedings of the 17th Australian Joint Conference on Artificial Intelligence. Cairns, Australia: Springer, pp.4-10.

Bharati, A. (2002) A Document Space Model for Automated Text Classification Based on Frequency Distribution Across Categories. Mumbai: ICON.

Briggs, P. (2002) The Bradt Travel Guide to Ethiopia. England, UK: Bradt Ltd, pp.12-40.

Buckley, C., Hersh, W., Leone, T. and Hickarn, D. (1998) An Interactive Retrieval Evaluation and New Large Test Collection for Research. Proceedings of the

17th Annual International Conference on Research and Development in Information Retrieval. Dublin, Ireland: ACM, Springer, pp.127-140.

Cheng, C., Tang, T., Fu, A. and King, I. (2001) Hierarchical Classification of Documents with Error Control, Singapore: IEEE, 20(35), pp.10-19.

Dawson, C. (2002) Practical Research Methods. New Delhi: UBS Publishers.

Deng, Z., Tang, S., Yang, D., Zhang, M. and Wu, X. (2002) Two Odds-Radio Based Text Classification Algorithms. Proceedings of the Third International Conference on Web Information Systems Engineering Workshop. Singapore: IEEE, pp.20-25.

Dhillon, I. (2003) A Divisive Information Theoretic Feature Clustering Algorithm for Text Classification. Journal of Machine Learning Research, 3(42), pp.1265-1287.

Duwairi, R. (2007) Arabic Text Categorization. International Arab Journal of Information Technology, 4(2), pp.1-7.

Feng, Y. (2005) Multi-Label Text Categorization Using K-Nearest Neighbor Approach with M-Similarity. Proceedings of the 12th International Conference on String Processing and Information Retrieval. Buenos Aires, Argentina: Springer, pp.3-12.

Girma Berhe (2001) A Stemming Algorithm Development for Tigrigna Language Text Documents. MSc Thesis. Addis Ababa University, Addis Ababa, Ethiopia.

Hammond, J. (1999) A Chronicle of the Revolution in Tigray region of Ethiopia. Eritrea: Red Sea Press.

Joachims, T. (1998) Text Categorization with Support Vector Machines: Learning with Many Relevant Features. University of Dortmund, Germany.

John, C. (1998) Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. USA: Morgan Kaufmann.

Karypis, G., Steinbach, M. and Kumar, V. (2004) A Comparison of Document Clustering Techniques. New York, USA: ACM Press/Addison-Wesley Publishing Co.

Kassa Gebrehiwot (2003) A Tigrigna Language Dictionary. Addis Ababa, Ethiopia: EMAY Printers.

Kiritchenko, S. (2006) Hierarchical Text Categorization and Its Application to Bioinformatics. PhD Thesis. Duke University, Ottawa ,Canada.

Koller, M. (1997) Hierarchically Classifying Documents Using Very Few Words. San Francisco, USA: Morgan Kaufmann.

Kyriakopoulou, A. (2008) Combining Clustering with Classification for Spam Detection in Social Bookmarking Systems. Athens University of Economics and Business, Athens: ECML/PKDD.

Lang, K. (1995) Learning to Filter Net News. Proceedings of the Twelfth International Conference on Machine Learning. Tahoe City, USA: Morgan Kaufmann, pp.280-296.

Larkey, S. (1999) A Patent Search and Classification System. New York, USA: ACM.

Leslau, W. (1998) Documents Tigrigna. Paris: Libraire CKlincksieck.

Maron, M. and Kuhns, J. (1960) Probabilistic Indexing and Information Retrieval. London: ACM, 23, pp.30-35.

Matthew, R. (2004) A Graphical Interface to Clustering Algorithms and Visualizations. University of Minnesota, USA.

McCallum, A., Nigam, K., Thrun, S. and Mitchell, T. (2000) Text Classification from Labeled and Unlabeled Documents Using EM. Boston: Kluwer Academic Publishers, 39(2), pp.103–134.

Michelangelo, C. and Malerba, D. (2007) Classifying Web Documents in a Hierarchy of Categories: A Comprehensive Study. Italy: Springer.

Mladenic, D. (1998) Machine Learning on Non-Homogeneus, Distributed Text Data. PhD Thesis. University of Ljubjana, Slovenia.

Ozgur, A. (2004) Supervised and Unsupervised Machine Learning Techniques for Text Document Categorization. MSc Thesis. Bogazin University, Turkey.

Popa, I., Zeitouni, K., Gardarin, G., Nakache, D. and Metais, E. (2007) Text Categorization for Multi-Label Documents and Many Categories. Washington DC, USA: IEEE.

Rafael, A., Li, X. and Lee, J. (2004) Managing Content with Automatic Document Classification. Journal of Digital Information, pp.23-40.

Ramirez, R. (2008) A Roadmap to SVM Sequential Minimal Optimization for Classification. University of Central Florida, Orlando.

Sahle, A. (1998) Sawasasəw Tigrigna Besafih. Asmera, Eritrea: Red Sea Press.

Salton, G and Buckley, C. (1988) Term Weighting Approaches in Automatic Text Retrieval. New York, USA: Pergamon Press, pp.24-30.

Salton, G., Yang, C. and Wong, A. (1975) A Vector Space Model for Automatic Indexing. New York, USA: ACM Press.

Sebastian, F. and Giorgetti, D. (2003) Multiclass Text Categorization for Automated Survey Coding. Proceedings of the 2003 ACM Symposium on Applied Computing. Melbourne, USA: ACM Press, pp.230-235.

Sebastiani, F. (2000) A Tutorial on Automated Text Categorization. Consiglio Nazionale delle Ricerche, Italy: Istituto di Elaborazione dell' Informazione.

Sebastiani, F. (2002) Machine Learning in Automated Text Categorization. ACM Computing Surveys. Consiglio Nazionale delle Ricerche, Italy: ACM, pp.10-15.

Sebastiani, F. (2005) Text Categorization in Text Mining and Its Applications to Intelligence, CRM and Knowledge Management. South Hampton, UK: WIT Press.

Slonim, N. (2001) The Power of Word Clustering for Text Classification. European Colloquium on IR Research : ECIR , pp.30-45.

Sun, I. and Lim, E. (2001) Hierarchical Text Classification and Evaluation. Proceedings of the 2001 IEEE International Conference on Data Mining. Washington DC, USA: IEEE, pp.20-23.

Surafel Teklu (2003) Automatic Categorization of Amharic News Text: A Machine Learning Approach. MSc Thesis. Addis Ababa University, Addis Ababa, Ethiopia.

Tikk, D., Biro G. and Yang, J. (2001) A Hierarchical Text Categorization Approach and Its Application to FRT Expansion. Hungary: Elsevier.

Witten, I. and Frank, E. (2005) Data Mining: Practical Machine Learning Tools and Techniques. San Francisco, USA: Morgan Kaufmann.

Work Kelemework (2009) Automatic Amharic Text News Classification: A Neural Networks Approach. MSc Thesis. Addis Ababa University, Addis Ababa, Ethiopia.

Yang, Y. and Pedersen, J. P. (1997) A Comparative Study on Feature Selection in Text Categorization .USA: Morgan Kaufmann.

Yohannes Afework (2007) Automatic Classification of Amharic News Text. MSc Thesis. Addis Ababa University, Addis Ababa, Ethiopia.

Yong, U. and Mooney, J. R. (2002) Text Mining with Information Extraction. Stanford: Citeseerix

Zelalem Sintayehu (2001) Amharic News Classification. MSc Thesis. Addis Ababa University, Addis Ababa, Ethiopia.

Zhao, Y. (2002) Comparison of Agglomerative and Partitioning Document Clustering Algorithms. Washington DC: ACM Press.

APPENDIX

Appendix I: Tigrigna Alphabets

1st	2nd	3rd	4th	5th	6th	7th
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
ቐ	ቑ	ቒ	ቓ	ቔ	ቕ	ቖ
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ
ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
ኸ	ኹ	ኺ	ካ	ኼ	ኽ	ኾ
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ
ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ

Appendix II: Tigrigna Numbers

Tigrigna Numbers	Arabic
፩	1
፪	2
፫	3
፬	4
፭	5
፮	6
፯	7
፰	8
፱	9
፲	10
፳	20
፴	30
፵	40
፶	50
፷	60
፸	70
፹	80
፺	90
፻	100
፼	10000

Appendix III: Tigrigna Script Translation in to Latin Script for Preprocessing

ሀ he	ሁ hu	ሂ hi	ሃ ha	ሄ hie	ህ h	ሆ ho
ለ le	ሉ lu	ሊ li	ላ la	ሌ lie	ል l	ሎ lo
ሐ He	ሑ Hu	ሒ Hi	ሓ Ha	ሔ Hie	ሕ H	ሖ Ho
መ me	ሙ mu	ሚ mi	ማ ma	ሜ mie	ም m	ሞ mo
ሠ se	ሡ su	ሢ si	ሣ sa	ሤ sie	ሥ s	ሦ so
ረ re	ሩ ru	ሪ ri	ራ ra	ሬ rie	ር r	ሮ ro
ሰ se	ሱ su	ሲ si	ሳ sa	ሴ sie	ስ s	ሶ so
ሸ xe	ሹ xu	ሺ xi	ሻ xa	ሼ xie	ሽ x	ሾ xo
ቀ qe	ቁ qu	ቂ qi	ቃ qa	ቄ qie	ቅ q	ቆ qo
ቐ Qe	ቑ Qu	ቒ Qi	ቓ Qa	ቔ Qie	ቕ Q	ቖ Qo
በ be	ቡ bu	ቢ bi	ባ ba	ቤ bie	ብ b	ቦ bo
ቨ ve	ቩ vu	ቪ vi	ቫ va	ቬ vie	ቭ v	ቮ vo
ተ te	ቱ tu	ቲ ti	ታ ta	ቴ tie	ት t	ቶ to
ቸ ce	ቹ cu	ቺ ci	ቻ ca	ቼ cie	ች c	ቾ co
ኀ he	ኁ hu	ኂ hi	ኃ ha	ኄ hie	ኅ h	ኆ ho
ነ ne	ኑ nu	ኒ ni	ና na	ኔ nie	ን n	ኖ no
ኘ Ne	ኙ Nu	ኚ Ni	ኛ Na	ኜ Nie	ኝ N	ኞ No
አ ae	ኡ au	ኢ ai	ኣ aa	ኤ aie	እ a	ኦ ao
ከ ke	ኩ ku	ኪ ki	ካ ka	ኬ kie	ክ k	ኮ ko
ኸ Ke	ኹ Ku	ኺ Ki	ኻ Ka	ኼ Kie	ኽ K	ኾ Ko
ወ we	ዉ wu	ዊ wi	ዋ wa	ዌ wie	ው w	ዎ wo
ዐ eee	ዑ eeu	ዒ eei	ዓ eea	ዔ eeie	ዕ ee	ዖ eeo
ዘ ze	ዙ zu	ዚ zi	ዛ za	ዜ zie	ዝ z	ዞ zo
ዠ Ze	ዡ Zu	ዢ Zi	ዣ Za	ዤ Zie	ዥ Z	ዦ Zo
የ ye	ዩ yu	ዪ yi	ያ ya	ዬ yie	ይ y	ዮ yo
ደ de	ዱ du	ዲ di	ዳ da	ዴ die	ድ d	ዶ do
ጀ je	ጁ ju	ጂ ji	ጃ ja	ጄ jie	ጅ j	ጆ jo
ገ ge	ጉ gu	ጊ gi	ጋ ga	ጌ gie	ግ g	ጎ go
ጠ Te	ጡ Tu	ጢ Ti	ጣ Ta	ጤ Tie	ጥ T	ጦ To
ጨ Ce	ጩ Cu	ጪ Ci	ጫ Ca	ጬ Cie	ጭ C	ጮ Co
ጰ Pe	ጱ Pu	ጲ Pi	ጳ Pa	ጴ Pie	ጵ P	ጶ Po
ጸ Se	ጹ Su	ጺ Si	ጻ Sa	ጼ Sie	ጽ S	ጾ So
ፀ Se	ፁ Su	ፂ Si	ፃ Sa	ፄ Sie	ፅ S	ፆ So
ፈ fe	ፉ fu	ፊ fi	ፋ fa	ፌ fie	ፍ f	ፎ fo
ፐ pe	ፑ pu	ፒ pi	ፓ pa	ፔ pie	ፕ p	ፖ po

Appendix IV: Tigrigna prefix lists

መ	ዘይተ	ከምዘይ	ክነ	ስለዘ	እንድሕር
ይ	ዝተተ	ከም	ክተ	ስለዝተ	እንተዘይ
ክ	ዘተ	ከምእን	ከምዝተ	ስለተ	እንዳ
ዝ	ዝም	ከምት	ከምዘ	ኣይ	እንዳተ
ዘ	ዘይም	ከምዘይተ	ከነ	ኣይን	እና
ተ	ዝተመ	ክን	ክነ	ኣይምተ	እናተ
ም	ከምዝ	ክንዲ	ከየ	ኣይን	እንተተ
ን	ከከም	ክሳብ	ክት	ኣይተ	እንተዝ
ብ	ከምተ	ክሳብዝ	ክተተ	ኣይት	እን
ኣ	ከከምዝ	ክንድት	ክተት	እንተይ	
ዘዝ	ከይ	ክንድዝ	ስለዝ	እንተ	
ዘየ	ክት	እን	ስለ	እንከይ	
ዘይ	ክተ	ክን	ስለዘይ	እንተይተ	

Appendix V: Tigrigna suffix lists

ስ	ዎ	ታይ	ይነትን	ኣውን	ካለንን
ና	ካ	ቶም	ያውን	ኣም	ሎም
ን	ኪ	ተኛ	ኣዊ	ኣምን	ለን
ቲ	ኩ	ኩም	ኣም	ኩም	ለይ
ኦ	ት	ነት	ኦት	ክን	ዎም
ኣ	እ	ናሎም	ዊነት	ኩምን	ወታይ
ም	ታ	ናዮም	እቲ	ካን	
ኛ	ይ	ነታዊ	ኡ	ክንን	
ዊ	ኡ	ነቱ	ኣም	ኩን	
ዋ	ታት	ናለን	ኣም	ካሎምን	

Appendix VI: Tigrigna stop word lists

ብ	እዙይ	እንተድኣ	ከምዛ	ከመይ	ከማና
ሕዚ	ኣሎ	እወ	ከምቲ	ካሊእ	ከማካ
በቢ	ኢና	እንኮ	ከምተን	ካልኦት	ከማኪ
ዝኾነ	ኣብዚ	እውን	ከምቶም	ከምዚ	ከምዙይ
ኾነ	እዚ	ኣበይ	ከምታ	ከምኡ	ከምዚኣ
ናብ	እያ	ኣብኡ	ከምቲኦም	ከማይ	ከምዚኣቶ ም
ኣብ	እያተን	ኣብዛ	ከምቲኣን	ከምኣ	
ኣነ	እያቶም	ኣብዘሓ	ከቶ	ከማና	ከምኡ
ኣበይ	እቶም	ካብ	ክሳብ	ከማኩም	ኩልክን
ኣየን	እየ	ካሊእ	ክንዲ	ከማክን	ከዓ
እቲ	የምፃኣ	ክንዲ	ክስይ	ከምኦም	ኮነ
ኮይኑ	ከምዘለዎ	ከም	ኩሎም	ከማካ	ከኣ
እታ	ተሓቢሩ	ከከም	ኩልና	ከማኪ	ክንደይ
እዩ	ሓደ	ክብል	ኩልክን	ከምኣቶም	ኩሎም
እናተ	ዓመት	ኹሉ	ኩላትና	ከምኣተን	ክንድቲ
እና	ክሳብ	ከምቲ	ኩላቶም	ክስቶ	ናይቲ
እዮም	እየን	ከምናቶም	ኩሉኩም	ክስታይ	ነቲ
እቶም	እውን	ከምናታ	ኩለካትክን	ከምዚኦም	ናይ
እሞ	እዙይ	ከዓ	ኩላተን	ከምዚኣን	ነቶም
ኣላ	እምበር	ካብዚ	ኩሉ	ከምኣን	ነቱይ

Appendix VII: List of Tigrigna special words which co-occur with another word to have meaning in a concept

ቤት
ስነ
መራሕቲ
ኣብያተ
መንበሪ
ወሃቢ
ሰበ
መምርሒ

ክነ