ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

DEPARMENT OF INFORMATION SCIENCE

HIERARCHICAL AMHARIC NEWS TEXT CLASSIFICATION

BY

ALEMU KUMILACHEW TEGEGNIE

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

JULY, 2010

Addis Ababa, Ethiopia

Acknowledgment

I would like to thank my advisor Ato Wondwossen Mulugeta next to God for his continuous, constructive comments and encouragement up to the date this research was finalized.

No less thanks is given to Ethiopian News Agency (ENA) for letting me use its data, especially Ato Belsti, News Structuring and Organization Expert, for his valuable support during the data preprocessing stages of the research.

I am also grateful to my brothers Yassabe Desta and Tewodros Gera for everything they did for me. Thank you!

Alemu Kumilachew Tegegnie

Table of Contents

Contents **Pages**

Acknowledgment …….. ……..i
Table of Contents.……...…….……ii
List of Tables ...……..…….v
List of Figures…. ……..……vi
List of Appendices……..……..vii
List of acronyms……vii
List of synonymously used words and phrases ……vii
Abstract …….ix

Chapter One i
Introduction 1
1.0. Background 1
1.1. Statement of the Problem and Its Justification 5
1.2. Objective of the Study 7
1.2.1 General Objective 7
1.2.2. Specific Objectives 7
1.3. Methodology 8
1.3.1. Literature Review 8
1.3.2. Data Source and Data Collection Methods 8
1.3.3. Development Tools and Techniques 8
1.3.4. Experimental Procedure 9
1.4. Scope of the Study 10
1.5. Significance of the Study 10
1.6. Application of the Study 10
1.7. Organization of the Study 11

Chapter Two 12
Automatic Text Classification 12
2.0. Introduction 12
2.1. Text Classification: Definition 13
2.2. Approaches of Text Classification 14
2.2.1. Manual Classification 14
2.2.2. Automated Classification 14
2.3. Automatic Text Categorization: Basic Concepts 17
2.4. Application of Automatic Classification 19
2.5. Hierarchical Text Classification 20
2.5.1. Introduction 20
2.5.2. Hierarchical Text Classification Methods 23
2.5.3. Single Label Versus Multi Label Classification 24
2.5.4 Single-Parent Vs Multi-parent Hierarchical Text Classification 25
2.6. Text Classification Steps 26
2.6.1. Preprocessing-Document Indexing 27
1.7. Machine Learning Approach to Hierarchical Text Classification 34
2.7.1. Introduction 34
2.7.2. Support Vector Machine (SVM) 36
2.7.3. Basic SVM Kernels 43
2.7.4. SVM Multiclass Classification Methods 44
2.7.5. Training Vs Test Sets 46
2.7.6. Performance Measures 46
Chapter Three 48
The Amharic Language and Its Writing System 48
3.0. Introduction 48
3.1. Origin of Amharic Language and Its Script 48
3.2. Amharic Characters/Alphabets 50
3.3. Amharic Punctuation Marks 51
3.4. Amharic Number System 52
3.5. Problems in Amharic Writing System 52

3.5.1. Characters with Different Form 53
3.5.2. Compound Words Usage 54
3.5.3. Transliterations problem 55
3.6. Amharic Unicode Representation 56
Chapter Four 57
Experiment and Discussion of Results 57
4.0. Introduction 57
4.1. Data pre-processing 58
4.1.1. Data source 58
4.1.2. Data cleaning 59
4.2. Representing Documents and Classes 64
4.2.1. Normalization and Tokenization 64
4.2.2. Stop Word Removal 66
4.2.3. Stemming: Affix removal 67
4.2.4. Term Weighting 72
4.3. LibSVM: Experimentation Tool 72
4.3.1. Input File Preparation 73
4.3.2. Running LibSVM 74
4.3.3. Performance Measures 76
4.4. Experiments and Results 76
4.4.1. Experimental Setup 76
4.4.2. Effects of the Number of Classes and Documents on Flat Classification 77
4.4.3. Experiment using Hierarchical Classification 81
4.4.4. Comparison between Flat Classifier and Hierarchical Classifiers 85
Chapter Five 86
Conclusion and Recommendation 86
5.0. Conclusion 86
5.2. Recommendations 91
References 93
Appendix 99

List of Tables

Table1.1. Findings of previous researches using flat classification approach.……..6

Table2.1: Category- to-document matrix.……..13

Table3.1 a sample list of Amharic characters (Fidel).……..51

Table 3.2a &b: Sample lists of labialized consonants.……....51

Table3.3.Some examples of punctuation marks used in Amharic.……..52

Table3.4.Some lists of Amharic characters with the same sound.……53

Table3.5. Examples of the different word spellings ……..54

Table3.6. Inconsistencies caused by compound words.……..55

Table3.7.Word variations due to transliterations.……..55

Table4.1. Statistics of data collected from ENA (2007-2010).…..61

Table4.2. Document-similarity matrix for selected class "Health".……62

Table4.3.a, b & c. Cluster generation through document-similarity matrix.……..63

Table4.4: Example of hierarchical level for classes ''Tourism & Culture" and "Health".……..64

Table 4.4.Normalization, Stop word, stemming experiments for feature reduction.…..…71

Table 4.5.Accuracy of a normalizer, stop word removal and stemmer on 500 sample tokens.……72

Table4.6. Experiment with 8- classes.……..79

Table4.7. Experiment with 20- classes ….....79

Table4.8. Experiment with 69- classes.……..80

Table4.9: The Increasing performance of hierarchical classifiers.……..82

List of Figures

Figure2.1. Text classification approaches.……..16

Figure2.2: A sample category tree.……..21

Figure2.3: An example of multi-parent class ……..25

Figure 2.4: Document classification as a two step process.……..26

Figure 2.5: Components of document preprocessing and indexing.……....27

Figure2.6. Linearly separable data ……..37

Figure2.7. Two possible hyper planes and their associated margins

for the same training data ……..38

Figure2.8. Non-separable data set.……41

Figure3.1.Origins of Ethiopic, a family tree model.……49

Figure4.1. Architecture of the Hierarchical Amharic News Text Classifier.…….....…..58

Figure4.2. Effects of the number of top features on the performance of flat classification

…….81

Figure4.3: Effects of the number of top features on level-0 classification accuracy.…..…..83

Figure4.4: Effecst of the number of topfeatures on level-1 classification accuracy.……84

Figure4.5: Effects of the number of top features on level-2 classification accuracy.…...85

List of Appendices

Appendix1: Lists of Amharic characters **('Fidel')-ፊደል**

Appendix2: Lists of Amharic punctuation marks

Appendix3: Lists of Amharic Numbers

Appendix4: News items - major and sub categories

Appendix5: The hierarchical data associated with their unique codes

Appendix6: Lists of affixes removed from the token

Appendix 7: Lists of special words which should co-occur with another word

List of Acronyms

AI	Artificial Intelligence
ASCII	American Standard Code for Information Interchange
DF	Document Frequency
ENA	Ethiopian News Agency
ICT	Information and Communication Technology
IDF	Inverse Document Frequency
IR	Information Retrieval
KNN	K-Nearest Neighbor
LibSVM	Library for support Vector Machine
RBF	Radial Basis Function
SVM	Support Vector Machine
TC	Text Classification
TF	Term Frequency
TF*IDF	Term Frequency by Inverse Document Frequency

Words Used synonymously in This Paper

- **Text categorization** and **Text Classification**
- **Word, Feature,** and **Term**
- **Classes** and **Categories**
- **Classification Tree, Category Tree** and **Hierarchy**

Abstract

The advancement of the present day technology enables the production of huge amount of information. Retrieving useful information out of these huge collections necessitates proper organization and structuring. Automatic text classification is an inevitable solution in this regard. However, the present approach focuses on the flat classification, where each topic is treated as a separate class, which is inadequate in text classification where there are a large number of classes and a huge number of relevant features needed to distinguish between them.

This paper explores the use of hierarchical structure for classifying a large, heterogeneous collection of Amharic News Text. The 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.

An experiment had been conducted using a categorical data collected from Ethiopian News Agency (ENA) using SVM to see the performances of the hierarchical classifiers on Amharic News Text. The findings of the experiment show the accuracy of flat classification decreases as the number of classes and documents (features) increases. Moreover, the accuracy of the flat classifier decreases at an increasing number of top feature set. The peak accuracy of the flat classifier was 68.84 % when the top 3 features were used.

The findings of the experiment done using hierarchical classification show an increasing performance of the classifiers as we move down the hierarchy. The maximum accuracy achieved was 90.37% at level-3(last level) of the category tree. Moreover, the accuracy of the hierarchical classifiers increases at an increasing number of top feature set compared to the flat classifier. The peak accuracy was 89.06% using level three classifier when the top 15 features were used.

Furthermore, the performance between flat classifier and hierarchical classifiers are compared using the same test data. Thus, it shows that use of the hierarchical structure during classification has resulted in a significant improvement of 29.42 % in exact match precision when compared with a flat classifier.

Keywords: Automatic Text Classification, Flat Classification, Hierarchical Classification, Support Vector Machine

Chapter One

Introduction

1.0. Background

Humans use classification techniques to organize things in various activities of their life. People make their own judgment to classify things in their everyday life – they classify things based on similarities or likeliness of color, size, concept, ideas, subject and so on (Koller& Sahami, 1997).

The need to classify information resources has become an important issue as the production of such resources increase dramatically from time to time. More specifically, for the last 6 decades (Dumais & Chen, 2000), there is a great increase in the production of information. Manuscripts, newspapers, journals, magazines, thesis and dissertations are available electronically in different formats such as text, audio, video, and graphics. Several studies have shown that these collections are constantly growing from time to time due to the advancement of technology (Chien et.al. 2004). However, seeking information of one's need in these huge collections requires organization. Especially in a system where there is large collection of documents, retrieval of a given document or set of documents is possible if the collection is organized systematically. Many web sites offer a hierarchically organized view of the Web. E-mail clients offer a system for filtering e-mail. Academic communities often have a Web site that allows searching on papers and shows an organization of papers.

Nowadays, news items are produced every day in digital devices and organized in some order (Rennie, 2001). However, most of the time, text classification process is done manually which brings about enormous costs in terms of time and money. In other words, organizing documents by hand or creating rules for filtering is painstaking and labor-intensive

Therefore, automatic classification systems are very desirable since they minimize such problems (Neumann & Schmeier, 1999; Rennie, 2001).

Automatic text classification can be done using two approaches (Sebastiani, 2002; Rasmussen, 1992): clustering or classification. Text clustering is the automatic identification of a set of natural categories and the grouping of documents under each. Text classification, on the other hand, is the automatic assignment of documents to a predefined set of categories.

Many previous studies focus on *flat classificatio*n, in which the predefined categories are treated individually and equally so that no structures exist to define relationships among them (Yang & Liu, 1999; D'Alessio et al., 2000). A single huge classifier is trained which categorizes each new document as belonging to one of the possible predefined classes.

Limitations to the flat classification approach exists in the fact that, as the Internet grows, the number of possible categories increases and the borderlines between document classes are blurred. As we use a large corpus we may have hundreds of classes and thousands of features. The computational cost of training a classifier for a problem of this size is prohibitive. Furthermore, the variance of the resulting classifier is typically very large; since such a model will have many thousand parameters which need to be estimated and thus can easily lead to over fitting of the training data.

Even though, previous work (Koller & Sahami, 1997) has shown that feature selection can be a useful tool in dealing with these issues by eliminating many of the words that appear in the corpus as being unindicative of any topic so as to obtain a significant increase in accuracy, the computational cost and the robustness still pose significant limitations.

To resolve this issue, Koller and Sahami (1997) suggest the use of hierarchical structures in text classification. In hierarchical text classification (Sun & Lim, 2001) a large classification problem can be addressed using a divide-and-conquer approach. Koller (1997) proposed an approach that utilizes the hierarchical topic structure to decompose the classification task into a set of simpler problems, one at each node in the classification tree. At each level in the category hierarchy, a document can be first classified into one or more sub-categories using some flat classification methods. In such a hierarchical structure document types become more specific as we go down in the hierarchy. We can use features from both the current level as well as its children to train this classifier. Rather than ignoring the topical structure and building a single huge classifier for the entire task, we use the structure to break the problem into manageable size pieces.

There are various reasons why researchers are motivated for designing hierarchical document classification (D'Alessio et.al. 2000).

- First, rather than issue keyword-based queries from general-purpose search engines, many users prefer to look for information by browsing hierarchical catalogs and by issuing queries that are corresponding to specific topics. Experiments have shown that an interface that organizes on the fly the keyword-based queries into hierarchies improves usability, search success rate and user satisfaction.
- Second, hierarchical structures identify only parent-child and no complex relationship between the categories and provide a valuable information source for many problems. Since hierarchical structures enable the use of a divide-and-conquer approach, they result in higher efficiency and accuracy.
- Third, the flattened classifier loses the intuition that topics that are close to each other in hierarchy have more in common with each other, in general, than topics that are spatially far

apart. These classifiers are computationally simple, but they lose accuracy because the categories are treated independently and relationship among the categories is not exploited.

- Fourth, text categorization in hierarchical setting provides an effective solution for dealing with very large problems. By treating problems hierarchically, the problem can be decomposed into several problems each involving a smaller number of categories. Moreover, decomposing a problem can lead to more accurate specialized classifiers.
- Lastly, it is important to note that the key here is not merely the use of feature selection, but its integration with the hierarchical structure. A single flattened classifier would have to consider all of these features in order to do a reasonable job of classifying all of the documents. For any given document, however, most of these features are irrelevant, and serve only to confuse the classifier. In the hierarchical approach, a document percolating down the hierarchy of classifiers only encounters questions concerning a small fraction of the features throughout the process.

In general, the use of the hierarchical structure allows developers to focus on the relevant distinctions in the dependency model.

Hierarchical classifications of this type have long been used in special-purpose collections of documents such as MEDLINE or collections of patent documents (Koller & Sahami, 1997). Several researchers have studied the use of hierarchies for text classification and obtained promising results (D'Alessio et al., 2000; Sun & Lim, 2001; Ashwin & Susan, 2001). Their findings have shown that the use of the hierarchical structure during classification has resulted in a significant improvement of 45.4% in exact match precision when compared with a flat classifier. More recently, they have been used in several internet search engines, such as Yahoo (Yahoo! 1995) or Infoseek (Infoseek 1995) to categorize the contents of the World Wide Web.

1.1. Statement of the Problem and Its Justification

The automated classification (categorization) of texts has been flourishing in the last decade or so due to incredible increase in electronic documents on the Internet; this renewed the need for automated text classification (Klein, 2004). Even though, extensive studies have been already done for English language, other languages have also attracted increasing attention these days including Amharic, Arabic and Chinese texts (Chung & Noh, 2003; Solomon & Menzel, 2007).

Amharic is the native language of people living in the north central part of Ethiopia. The language is also spoken as a second language in many parts of the country. Significant number of immigrants in the Middle East, Asia, Western Europe and North America also speak Amharic (Solomon & Menzel, 2007). Amharic language has its own writing system that uses the Ge'ez alphabet.

Recently, there are numerous electronic documents produced and stored in Amharic by different organizations. More specifically, Ethiopian News Agency (ENA) produces and stores more than 100,000 news articles (Yohannes, 2007). Most of these data were destroyed due to damage and now has more than 16, 000 news articles. The agency has its own website that it uses to release news in Amharic and English. Now, the agency uses ENASoft software for the management of news. But the classification task is done manually. Currently, there are 110 categories available in ENA. Among these, 12 are major categories and 98 are sub categories. Using manual classification is challenging for these large number of classes. To cope up with the challenges of manual classification (see section 1.0 above), using automatic classification, four researches have been done by Zelalem (2001), Surafel (2003), Yohannes (2007) and Worku (2009).

Zelalem (2001) tried to design a flat news text classifier using a statistical analysis. The test result on three categories achieves 90.5% accuracy. However, to improve the performance of the classifier he recommended further work to investigate a hierarchical text classification approach, because of the fact that the classification of news items is hierarchical in nature.

As a continuation, different researchers attempt to design the flat news classification system using different machine learning approach, as shown in table1.1 below.

Name	**Numbers of documents used**	**Major categories Considered**	**Methods used**	**Accuracy achieved**
Surafel (2003)	1,024	16	KNN	85.05%
			Naïve Bayes	78.48%
Yohannes (2007)	Not clearly stated	15	LMT	79.72%
			LibSVM	80.15%
Worku (2009)	1,463	13	ANN	68.03%

Table1.1. Findings of previous researches using flat classification approach

According to researches (D'Alessio et al., 2000; Sun & Lim, 2001; Ashwin & Susan, 2000; Koller & Sahami, 1997) done on English text, as the number of classes increase due to huge collection of documents the accuracy decrease in a flat classification system. Furthermore, this can also be strengthened by experiments done on Amharic News text by Surafel (2003) and Yohannes (2007). The experiment is done using flat classification approach at an increasing number of categories and documents; which show that classification accuracy decreases as the number of categories and documents increase.

Since hierarchical text classification emphasize the relationships among classes and deal with this problem using a divide-and-conquer approach that decompose the classification task into a set of simpler problems, one at each node in the classification tree (Koller & Sahami, 1997; Sun & Lim, 2001), it can obtain a significant accuracy gains over the standard flat approach.

Hence, the aim of this research is to explore the use of hierarchical structure for classifying a large, heterogeneous collection of Amharic news items such that rather than building a single massive classifier, a hierarchy of classifiers will be constructed that increase accuracy and speed of classification.

1.2. Objective of the study

1.2.1. General Objective

The main objective of this research is to explore the possibility of designing and developing hierarchical news text classifier that is effective and efficient in classifying a large, heterogeneous collection of Amharic news text.

1.2.2. Specific Objectives

The specific objective of this research is to:

- Review literature on the concepts, techniques and tools of hierarchical text classification.
- Select classification technique to build a hierarchical Amharic news text classification system
- Develop a hierarchical Amharic text classifier using a selected technique so that the classification is applied on test data set.
- Evaluate the performance of hierarchical based Amharic news text classifiers compared to a flat classifier.
- Recommend further research areas for future work.

1.3. Methodology

The following methods were employed to achieve the above stated objectives.

1.3.1. Literature Review

Extensive study of available literature (books, journals, Internet, research works, etc) have been reviewed to understand the concepts and approaches of text classification in general and hierarchical text classification in particular. Automatic text classification, basic concepts, approaches, and techniques; hierarchical text classifier algorithms, basics of Amharic writing system and development tools and techniques in the area of text classification were the major once which have been reviewed extensively in this paper.

1.3.2. Data source and Data Collection Methods

The data source used for this study was Ethiopian News Agency (ENA). The Researcher used ENA as a source for two reasons. Firstly, there is no large collection of data available and easily accessed for research, as to the researcher's knowledge. Secondly, ENA uses manual document categorization; and the hierarchical nature of the existing classification system makes it interesting to undertake the study.

The collected data is then pre-processed and used after classified in two training (70%) and test data (remaining) sets for classification. Taking 70% of it as training data and the remaining for test data is recommended by the experimentation tool as most of classifier done using these much data has shown good performance, and is taken as a default value by the tool.

1.3.3. Development Tools and Techniques

There are many Machine learning algorithms such as Support Vector Machine (SVM), Decision Tree, Naïve Bayes and Artificial Neural Network used for hierarchical text classification (Ranganatan, 2001).

SVM is selected for this study due to its capability of providing the following benefits as compared to other algorithms (Joachims, 1998). SVMs:

- support high dimensional input space so that it can deal with large data sets
- tend to be less prone to over fitting since the learned classifier is characterized by the number of support vectors rather than the dimensionality of the data
- are capable of providing more accurate results

LIBSVM[multiclass] is selected as a tool for this experiment as it provides the following main features:

- Different SVM formulations
- Efficient multi-class classification
- Cross validation for model selection
- Automatic model selection which can generate contour of cross validation accuracy.
- Automatic parameter tuning to select the best parameter for training the classifier.

Python 3.0 is used as a programming tool for data preprocessing as it is a powerful too in text preprocessing, easy to use and familiarity with the researcher.

1.3.4. Experimental Procedure

In this study, the following procedures were followed to undertake the experiment.

- Data preprocessing including data cleaning, normalization, stop word removal, stemming & exception handling, and term weighting.
- Transform (prepare) the data to the format of an LibSVM package
- Conduct simple scaling on the data
- Systematically try a few kernels & parameters and select the one which performs best.
- Use the best parameter to train the whole training set
- Test the classifier.

1.4. Scope of the Study

The scope of this study is limited to investigating the possibility of designing & development of hierarchical Amharic news text classification system for ENA using Support Vector Machine (SVM) algorithm. The study is extended to evaluate the performance of hierarchical text classifier against the traditional text classifier (flat classifier). Moreover, the data considered in this study are the keyword, slug, the title and the first three hundred fifty characters of the news item.

1.5. Significance of the Study

There are many organizations today which use hierarchical nature of manual categorization system such as Yahoo!, etc and special purpose collections such as MEDLINE including ENA. The out put of this research could be used as an input to the development of a general hierarchical Amharic news text classifier for ENA. In addition to this, it can be used as initiative for further study in the area of hierarchical based Amharic text classification with a different approach (such as ANN, DT, Naïve Bayes and KNN).

1.6. Application of the Study

Recent advances from IR and AI have made document classification a hot issue. Document classification may appear in many applications: automatic indexing for Boolean information retrieval systems, document organization, text filtering, news monitoring, hierarchical categorization of web pages and word sense disambiguation (Sebastiani, 2002). The findings of this study can be further extended to do similar researches in the aforementioned domains.

1.7. Organization of the Study

The study is organized in to five chapters. The first chapter introduces background for text classification, the statement of the problem and its justifications, objective of the study, methodology, scope and application of the study.

Chapter two describes basic concept and approaches of text classification, methods and steps involved in text classification, machine learning algorithms used for text classification in general and hierarchical classification in particular. Support Vector Machine (SVM) is also discussed in greater detail.

Chapter three elaborates the domain of the study, i.e. the Amharic language and its writing system. It also discussed the origins of the language, characters (alphabets) and the problems associated to the writing system of the language.

Chapter four discusses the experimentation. It presents document pre-processing, LibSVM basics, input file preparation, experimentation details, results, analysis, and finding of the study.

The last chapter, Chapter Five, contains concluding remarks and recommends further work**.**

Chapter Two

Automatic Text Classification

2.0. Introduction

With the recent advances in computer technology and systems, the process of collecting information regarding transactions, customers, competitors, and other environmental factors has become progressively easier. However, the emergence of the current digital age has brought both new opportunities and unprecedented challenges to organizations. One of the major challenges is managing the overwhelming volume of information, as a result of the continuous expansion of the Internet, inventions and advances in information technology (IT). Searching through a large amount of collections is a challenge and can bring great loss in the form of productivity waste (workers spend about 65 percent of their time searching for information needed to complete their work (Eiring, 2002)); missed opportunities (failure to discover patterns and trends); and mismanaging or lack of managing knowledge (W. Zaghloul et al. 2009).

The rapid growth in stored and transient data led to a great deal of interest in developing useful and efficient tools and software to assist users in finding relevant information. In this respect, text categorization has become one of the key techniques for handling and organizing text data. It has been proved to be a powerful technique for automating assignment of documents to categories, in turn helps to organize and search text information on text data sources (Y. Kwok**,** 1999**;** Leopold & Kindermann, 2002, 1999, Sebastiani, 2002, etc). Automatic classification schemes can greatly facilitate the process of categorization (Joachims 1998; Han et al., 1999). In this chapter the concept of text classification, approaches of text classification, machine learning algorithms to hierarchical text classification are discussed.

2.1. Text Classification: Definition

The concept of text classification is defined by different authors as the task of automatically assigning a set of documents into categories (or classes, or topics) from a predefined set (Murtagh & Anderson 1999; Giorgino 2004; Klein 2004, Blumberg and Atre, 2003, Leopold & Kindermann, 2002, etc). More specifically, text classification describes the process of matching/mapping a document with the best possible concept(s) from a predefined set of concepts (categories).

In other words, If C= $\{c_1, c_2 \ldots c_m\}$ is a set of categories (classes) and D= $\{d_1, d_2, \ldots, d_n\}$ is a set of documents, the purpose of text classification is assigning c_i to d_j ($1 \leq i \leq m$ and $1 \leq j \leq n$) a value of 0 if the document d_j does not belong to c_i; otherwise a value of 1. The mapping is sometimes referred to as the decision matrix (Klein, 2004) and it is depicted in table2.1 below.

	d_1	...	d_j	...	d_n
c_1	A_{11}	...	a_{1j}	...	a_{1n}
...	...	...	...	...	...
c_i	a_{i1}	...	a_{ij}	...	a_{in}
...	...	...	...	...	...
c_m	a_{m1}	...	a_{mj}	...	a_{mn}

Table2.1: Category- to-document matrix

Building a text classifier is a two step process: *training* and *classification*. In the first step, training, the system is given a set of pre-classified documents. It uses these to learn the features that represent each of the concepts. In the classification phase, the classifier uses the knowledge that it has already gained in the training phase to assign a new document to one or more of the categories. Schutze et.al (1995) has underlined the role of feature selection in document classification to improve the performance of the classification. They have shown improvement by using latent semantic indexing and optimal term selection to reduce the number of features.

2.2. Approaches of Text Classification

Robert Blumberg and Shaku Atre (2003) identify the following four approaches in text classification.

2.2.1. Manual Classification

Manual classification requires individuals to assign each document to one or more categories. These individuals are usually domain experts who are thoroughly versed in the category structure or taxonomy being used. It is often used in library and technical collections as well as in call centers and form-processing environments. Manual classification can achieve a high degree of accuracy-although even domain experts will occasionally disagree on how to categorize document. However, manual classification is more labor-intensive and therefore most costly than automated techniques.

2.2.2. Automated Classification

The second major approach for text classification is automated classification. It comprises of the following three variants (R. Blumberg & S. Atre 2003; Witten & Frank 2005, etc).

2.2.2.1. Rule-Based Classification

In this form of classification, keywords and Boolean expressions are used to categorize a document. This is typically used when a few words can adequately describe a category. For example, if a collection of medical papers is to be classified according to a disease, then a medical thesaurus that lists each disease together with its scientific, common and alternative names can be used to define the keywords for each category.

While rule-based approaches are effective for carefully tuning a limited number of categories, the expense of defining and maintaining categories is generally prohibitive for large-scale classification systems.

2.2.2.2. Supervised Learning

Most approaches to automated text classification require a human subject expert to initiate the learning process by manually classifying or assigning a number of training documents to each category. This classification system first analyses the statistical occurrences of each concept in the example documents and then constructs a model or classifier for each category that is used to classify documents automatically. The system refines its model, in a sense learning the categories as new documents are processed.

In supervised learning, the classification is seen as supervised learning from training examples. The supervision took place when the data (observations, measurements, etc) are labeled with pre-defined classes. It is like a “teacher” gives the classes (supervision) such that the test data are classified into these classes too.

2.2.2.3. Unsupervised Learning

These systems identify a group, or clusters of related documents as well as the relationship between these documents. Commonly referred to as *clustering*, this approach eliminates the need for training sets because it does not require a pre-existing category structure. However, clustering algorithms are not always good at selecting categories that are intuitive to human users. For this reason, clustering generally works hand-in-hand with the supervised learning techniques.

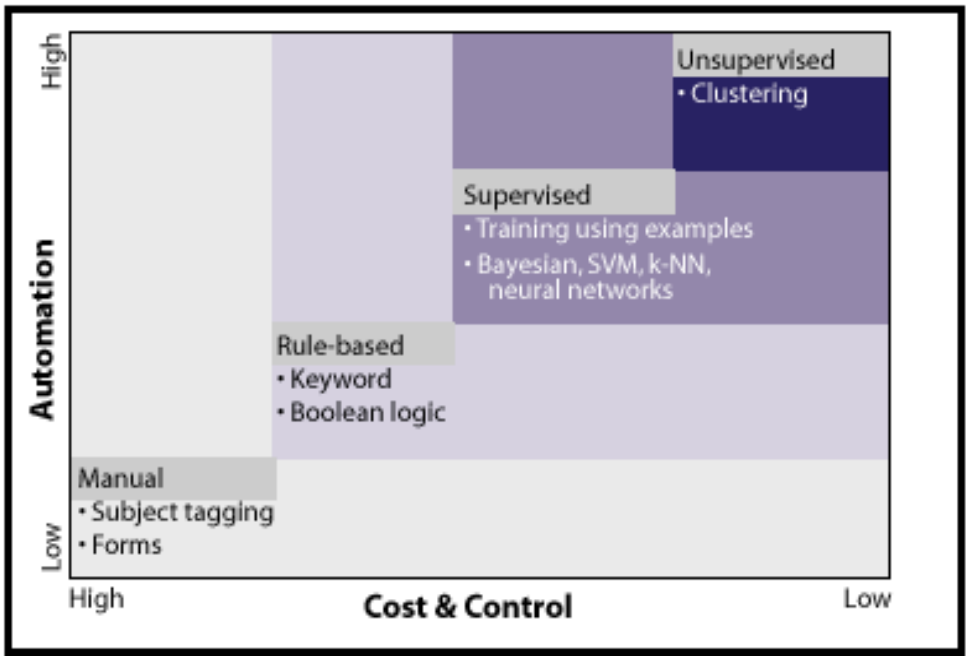

Figure2.1. Text classification approaches (adapted from: R. Blumberg & S. Atre 2003)

Figure2.1 shows the cost and control of the classification task always decreases, as the automation of the classification system moves from manual to unsupervised classification approach.

As discussed in the above sections**, e**ach of the four approaches is optimal for different situation (R. Blumberg and S. Atre, 2003). They explained that no single technique outperforms the others in all situations since categories vary in how precisely or diffusely they can be described. Therefore, we need to provide different approaches that can be used simultaneously and an appropriate way to combine the results so that users can achieve optimum results.

2.3. Automatic Text Categorization: Basic Concepts

With the explosive growth of the number of electronic documents (e.g. e-books or Web pages) for general reference or specific search purposes, is quite difficult, if not totally impossible, for library and information professionals to manually categorize and index documents. This is the problem of information overload (Levy, 1999). In fact, it is very time-consuming to classify a rich mixture of electronic documents solely based on the manual methods. To alleviate this problem some initial ideas for applying automatic document classification methods to categorization of electronic documents in an experimental setting have been explored (Chander, 97; Chung & Noh, 2003).

According to Sebastiani (2002), automatic text classification (categorization) is the task of building soft-ware tools to automatically assign some labels (from a set of pre-defined class labels) to a document based on some selected features of that document. Until the late 1980s, the most popular automatic text categorization method was based on the knowledge engineering approach, where a set of manually defined rules is applied to classify documents. However, the main problem of such an approach is the knowledge acquisition bottle-neck; domain experts must be available and heavily consulted in designing the classification rules. In fact, it is very time-consuming to elicit document classification knowledge even if domain experts are abundantly available, which is unlikely in the real world. In recent years, machine learning techniques have been applied to develop automated text classification systems (Joachims, 1997; Sebastiani, 2002; Koller, 1997; Yang & Liu, 1999). The advantage of applying machine learning approaches to automated document classification is that classification knowledge can be induced automatically based on a set of training documents.

The well-known machine learning algorithms for automatic text classification are the K-Nearest Neighbors (KNN) algorithm, Support Vector Machine (SVM), Decision Tree, Neural Network, and the Naïve Bayes (NB) algorithm. A comparison of these algorithms is presented in Yang &Liu (1999) and Sebastiani (2002). An outline of each method is as follows:

- **K-Nearest Neighbor (KNN):** Documents and concepts are represented as vectors in a vector space whose dimensions represent the various keywords in the vocabulary. The categories for a new document are determined by calculating the angle between the document vector and the category vector to identify the k-nearest neighbors. The weights of the dimensions are, for text, calculated using *tfidf*. *Tfidf* is a statistical measure (weight) used to evaluate how important a word is to a document in a collection or corpus.
- **Naïve Bayesian:** This approach uses the joint probabilities of words co-occurring in the category training set and the document to be classified to calculate the probability that the document belongs to each category. The document is assigned to the most probable category (ies). The naïve assumption in this method is the independence of all the joint probabilities.
- **Support Vector Machines:** This method represents every document as a vector and tries to find a boundary that achieves the best separation between the groups of vectors. The system is trained using positive and negative examples of each category and the boundaries between the categories are calculated. A new document is categorized by calculating its vector and determining the partition of the space to which the vector belongs.
- **Decision trees:** This kind of classifier builds a decision tree from the documents in the training set. Each branch defines a test on some attribute of the document. New documents are walked down the decision tree until the matching category is found.

- **Neural networks:** NN analysis is basically a prediction tool modeled on how human brain works. NNs are trained to recognize certain patterns or behavior when fed with a large data set and then they can determine predictors of a dependent variable. Thus, NN can be defined as a distributed processor that can create knowledge based on experience and make that knowledge available for future use.

2.4. Application of Automatic Classification

Recent advances from Information IR and AI have made text classification a hot issue. Its use appears in a wide variety of applications (Sebastiani, 2000)

- **Email filtering**

  Systems for filtering a person's incoming emails to weed out scam or spam or to categorize them into different classes are just now available.

- **Mail routing**

  Large enterprises are currently automating their document processing by means of work-flow management system, allowing an image of the document to circulate through the company rather than the original. In particular, they aim for a uniform treatment of incoming mail, whether it is electronic or in paper form. A bottleneck in this approach is the entering of documents into the right work flow. This process involves a superficial interpretation of the contents of the document, which is time consuming and error prone.

- **News monitoring**

  In knowledge-based companies like the stock exchanges, numbers of people are concerned with the scanning of newspapers and other information sources for items which are concerned with the national or international economy, or with individual companies on the stock market. The results are sent to the person who should be informed.

- **Narrowcasting**

  Press agencies strive to give more and more individual service, where each client obtains out of the large stream of outgoing news items only those that are relevant to him, according to his profile.

- **Content classification**

  Large information brokers have traditionally used pre-classification of documents as an aid in document disclosure. Documents are manually given a place within a large semantical hierarchy, or index terms according to a given thesaurus. This process is costly and error prone and changes in the thesaurus are hard to accommodate. Modern search machines on the web use an automatic pre-classification of web pages.

2.5. Hierarchical Text Classification

2.5.1. Introduction

Nowadays, text classification has become a wide research area in information science that develops methods for assigning text documents to a pre-defined set of categories. When the given categories are defined independently of one another, this is known as *flat classificatio*n. Whereas, when the structural relationship among a given categories are considered, this is known as *hierarchical classification.*

Most of the studies in text classification have focused on flat classification (W. Zaghloul et. al. 2009; Yi and Beheshti**,** 2008**;** Martinez**,** 2002**;** W. Zhang et.al, 2007; Leopold & Kindermann, 2002; Zelalem, 2001, Surafel, 2003; Yohannes, 2007; and Worku, 2009). They focus on categorizing a document into one or more class labels leaving the structural relationships among classes; treating each topic as a separate class (flat classification).

However, as the available information increases the inability of people to assimilate and profitably utilize such large amounts of information becomes more and more apparent without recognizing the structural relationship. There are two reasons where this difficulty is evident. In one case, the most successful paradigm for organizing this mass of information making it comprehensible to people is by categorizing the different documents according to their topic where topics are organized in a hierarchy of increasing specificity.

In the other case, hierarchical organization of documents have long been used for most real world organizations and in special purpose collections of documents (Koller & Sahami, 1997). More recently, they have been used in several internet search engines such as Yahoo (Yahoo! 1995) or Infoseek (Infoseek 1995), to categorize the contents of the World Wide Web.

In addition, limitations to the flat classification approach exists in the fact that, as the Internet grows, the number of possible categories increases and the borderlines between document classes are blurred. To resolve these issues, Koller and Sahami (1997) suggest the use of hierarchical structures for the first time in 1997. In such a hierarchical structure document types become more specific as we go down in the hierarchy

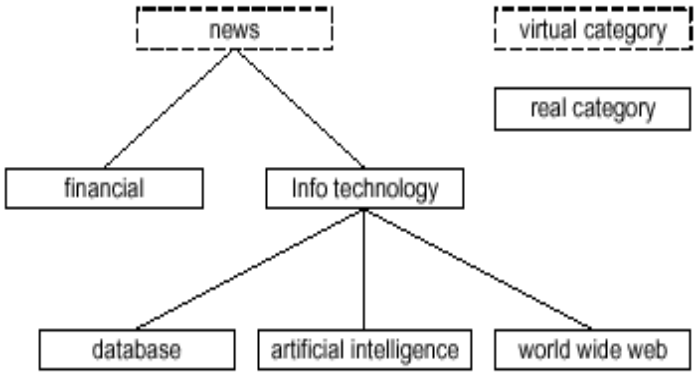

Figure2.2: A sample category tree (adapted from: Sun & Lim, 2001)

More recently, increasing attention has been given to *hierarchical classification* where the pre-defined categories are organized in a tree-like structure. A category tree example is shown in Figure2.2 above. In a category tree, there are parent-child relationships between categories. These parent-child relationships may suggest *strong* or *weak subsumption constraint* between categories. A parent and child category pair with strong subsumption constraint suggests all documents belonging to the child also belong to the parent (the study focuses on this as it fits with our data). Weak subsumption, on the other hand, allows a child category to have documents not belonging to its parent category. By organizing a large number of categories in a tree, hierarchical classification allows us to address a large classification problem using a divide-and-conquer approach, also known as the top-down approach (Sun & Lim, 2001). At the root level, a text document can be first classified into one or more child categories. The document can then be further classified at each child category to determine if it belongs to categories at the next lower level. The classification step can be repeated until the document cannot be further classified into any lower-level categories. In flat classification a given document is assigned to a category based on the outcome of one or one set of classifiers; whereas the assignment of document to the category can be the outcome of multiple sets of classifiers in hierarchical classification. These classifiers are associated to different levels of the category tree to filter away documents that do not belong to the lower level categories.

Hierarchical classifications might have taken different categorical structures (Sun et.al. 2003). Hence, category structures for hierarchical classification can be classified into four as explained in the following points:

a) *Virtual category tree*: In this category structure, categories are organized as a tree. Each category can belong to at most one parent category and documents can be assigned to leaf categories only.

b) *Category tree*: This is an extension of virtual category tree that allows documents to be assigned into both internal categories and leaf categories.

c) *Virtual directed acyclic category graph*: In this category structure, categories are organized as a Directed Acyclic Graph (DAG). DAG is a graph structure where one node can reach another by following a path of edges along the direction of arrows. It is a graph with directed edges and no cycles. Similar to a virtual category tree, documents can only be assigned to leaf categories. This paper employs this structure as it matches with the data collected for the study.

d) *Directed acyclic category graph*: This is perhaps the most commonly-used structure in the popular web directory services such as Yahoo! [Yahoo] and Open Directory Project [ODP]. Documents can be assigned to both internal and leaf categories.

2.5.2. Hierarchical Text Classification Methods

There are two basic approaches to hierarchical classification, namely, *big-bang* approach and the *top-down level-based* approach (Sun et.al, 2003). In the big-bang approach, only a single classifier is used in the classification process. Given a document, the classifier assigns it to one or more categories where a document is most appropriate by traversing the category tree from the root to the leaf node in one single step. The assigned categories can be internal or leaf categories depending on the category structure supported by the methods.

Whereas in top-down level-based approach, the classification is accomplished with the cooperation of classifiers built at each level of the tree. The test document starts at the root of the tree and is compared to categories at the first level. The document is assigned to the best matching first level category and is then compared to all sub-categories of that category. This process continues until the document reaches a leaf or an internal category below which the

document cannot be further classified. One of the obvious problems with top-down approach is that a misclassification at a parent class may force a document to be misrouted before it can be classified into child class (Wang, 2001).

Compared to the top-down level-based approach, the big-bang approach can only use the information carried by the category structure during the training phase but not the classification phase. As discriminative features (e.g. terms) at a parent category may not be discriminative at child categories (because of the above defined reasons), it is usually very difficult for a classification method using big-bang approach to exploit different sets of features at different category levels.

In this paper a top-down approach will be adopted as it helps to construct a better classification system since it utilizes the different features of classes at different category levels during both in training and classification phase. Moreover, the concepts of the approach fits with the data collected for this study.

2.5.3. Single Label Versus Multi Label Classification

Depending on the application, different constraints may be enforced on the text classification task. For instance, we might need that, for a given integer k, exactly *k* (or ≤k, or ≥k) elements of category, *C* be assigned to each document, *dj*∈D, where D is a document collection. The case in which exactly one category must be assigned to each $d_j \in D$ is often called the *single-label* (also known as *non- overlapping categories)* case, while the case in which any number of categories from 0 to |*C*| may be assigned to the same $d_j \in D$ refers to the *multi-label* (also known as *overlapping categorie*s) case. For instance, in hierarchical setting, a document can take the class labels from the root to the leaf along a category tree.

A special case of single-label TC is *binary* TC, in which each $d_j \in CD$ must be assigned either to category C_i or to its complement C_i^-.

In a top-down level-based approach hierarchical text classification, a single document could only take at most one of the top level classes but could take two or more class labels as it routes from the root level down to the last level class in the category tree.

In spite of the fact that the above statement is taken into consideration, this research employs multi label classification.

2.5.4. Single-Parent Vs Multi-Parent Hierarchical Text Classification

In the real world application of hierarchical text classification, a child class might have one or more parent class (es) depending on the relationship of concepts. When a class has a single parent class in a hierarchy, it is referred to as single-parent class. Whereas, a child class has one or more parent classes (as shown in the figure2.3) is referred to as multi-parent class. In a hierarchy, child classes may have single-parent or multi-parent.

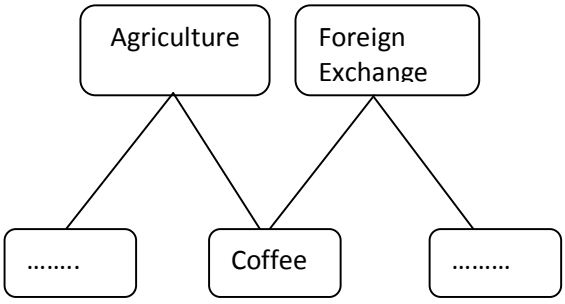

Figure2.3. an Example of multi parent class

Whatever is the type of relationship in the category tree, the selected top-down based text classification approach is independent of these cases. Because the top down approach starts from the root, selects the best category and then the features of child classes of the selected category is

compared at next level of the category tree and it continues in the same fashion (Wang 2001). Furthermore, the top-down approach leaves out the problem of single label or multi label problems described in section (2.5.4) with regard to only the upper top most class labels.

2.6. Text Classification Steps

The aim of text classification (whether flat or hierarchical) is to approximate an unknown target function through construction of a classifier on a given training data set (learning). Afterward, new, unseen documents are assigned to classes using the approximation function f (classification).

The classification task, *learning* and *classification* can be divided into the following two steps:

1. **Preprocessing/Indexing**: is the mapping of document contents into a *logical view* (e.g. a vector representation of documents) which can be used by a classification algorithm. Text operations and statistical operations are used to extract important content from a document.
2. **Learning/Classification**: based on the logical view of the document learning or classification takes place. It is important that for classification and learning the same preprocessing/indexing methods are used.

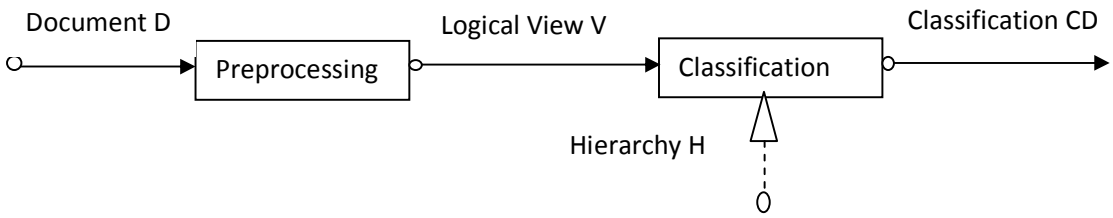

Figure 2.4: Document classification as a two step process: Preprocessing and classification.

2.6.1. Preprocessing-Document Indexing

As stated before, preprocessing is the step of mapping the *textual content* of a document into a *logical view* which can be processed by classification algorithms. A general approach in obtaining the logical view is to extract meaningful units *(lexical semantic*s) of a text and rules for the combination of these units *(lexical compositio*n) with respect to language. The lexical composition is actually based on linguistic and morphological analysis and is a rather complex approach for preprocessing. Therefore, the problem of lexical composition is usually disregarded in text classification.

Document processing (indexing) involves the main activities of text classification task such as term extraction, term weighting and dimensionality reduction. Figure2.5 shows the steps used for document preprocessing and their dependencies.

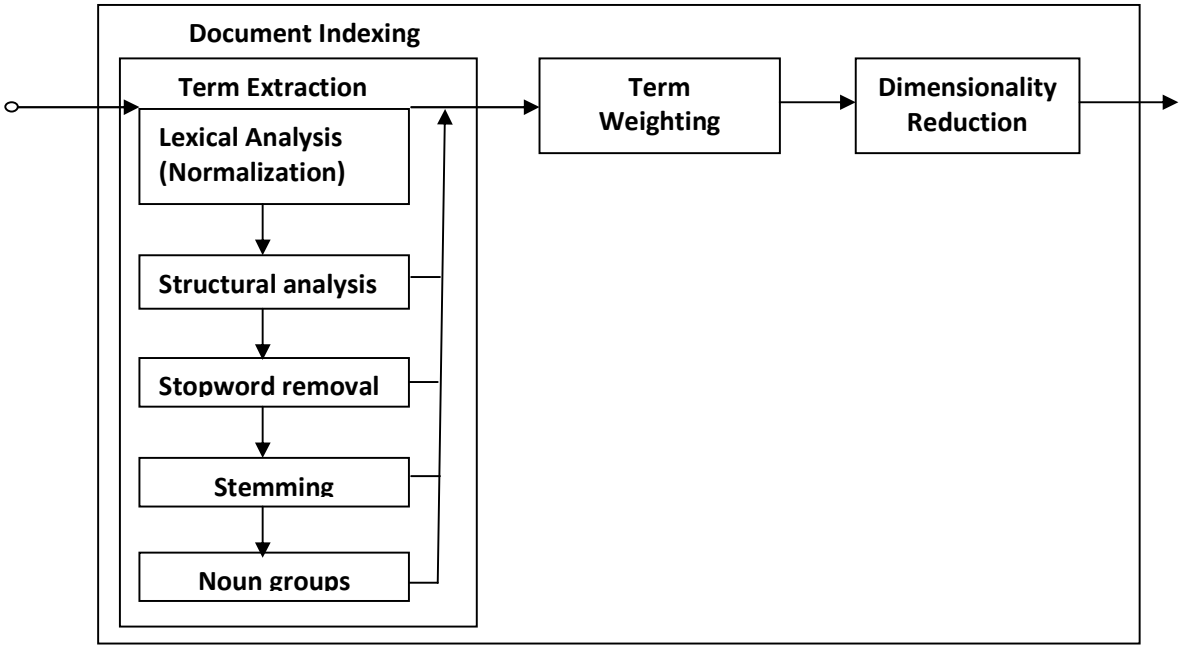

Figure 2.5: Components of document preprocessing and indexing (adapted from: Grantizer, 2000)

The logical view of a document D_j can be obtained by extracting all meaningful units (terms) from all documents D and assigning weights to each term in a document reflecting the importance of a term within the document. More formally, each document is assigned an n-dimensional vector $d_j=<w_1, w_2, w_3,\ldots,w_n>$, whereby each dimension represents a term from a term set *T*. The resulting *n*-dimensional space is often referred to as *Term Space* of a document corpus. Each document is a point within this Term Space. So preprocessing can be viewed as transforming character sequences into an *n*-dimensional vector space.

Obtaining the vector representation is called *Document Indexing* and involves two major steps:

1. *Term Extractio*n: Techniques for defining meaningful terms of a document corpus (e.g. lexical analysis, stemming, word grouping etc.)
2. *Term Weighting:* Techniques for defining the importance of a term within a document (e.g. Term Frequency (*TF*), Document Frequency (*DF*), Term Frequency Inverse Document Frequency (*TFIDF*))

2.6.1.1. Term Extraction

Term extraction, often referred to as feature extraction, is the process of breaking down the text of a document into smaller parts or terms. Term extraction results in a set of terms T which are used for the weighting and dimensionality reduction steps of preprocessing.

In general the first step is a *lexical analysis* where non-letter characters like sentence punctuation and styling information (e.g. HTML Tags) are removed. This reduces the document to a list of words separated by white space.

Beneath the lexical analysis of a document, information about the document structure like sections, subsections, paragraphs etc. can be used to improve the classification performance,

especially for long documents. Incorporating structural information of documents has been done in various studies (K. Summers, 1995; Hearts & Plaunt, 1993). Doing a *document structure analysis* may lead to a more complex representation of documents make the term space definition hard to accomplish. Most experiments in this area have shown that performance over larger documents can be increased by extracting structures and subtopics from documents.

Stop words are topic neutral words such as articles or prepositions contain no valuable or critical information. These words can be safely removed, if the language of a document is known. Removing stop words reduces the dimensionality of term space.

One problem in considering single words as terms is the semantic ambiguity (e.g. river bank, financial bank) which can be roughly categorized in:

- *Synonyms*: is a word which means the same as another word (e.g. Movie & Film).
- *Homonym*: refers to a word which can have two or more meanings (e.g. lie).

Since only the context of the word within a sentence or document can dissolve this ambiguity, sophisticated methods like morphological and linguistic analysis are needed to diminish this problem. In (Leopold & Kindermann 2002) morphological methods are compared to traditional indexing and weighting techniques. It was stated that morphological methods slightly increase classification accuracy for the cost of higher computational preprocessing.

Beside synonymous and homonymous words, different syntactical forms may describe the same word (e.g. go, went, walk, and walking). Methods for extracting the syntactical meaning of a word are *suffix stripping* or *stemming* algorithms. Stemming is the notation for reducing words to their word stems. Most words in majority of Western languages can be "stemmed" by deleting (stripping) language dependent suffixes from the word (e.g. CONNECTED, CONNECTING→CONNECT). The performance of stripping and stemming algorithms depends

strongly on the simplicity of the used language. For English a lot of stripping and stemming algorithms exist, the Porters Algorithm being the most popular one. Affix (prefix or suffix) removal is another form of stemming algorithm especially for languages which morphologically extensive (Leopold & Kindermann, 2002).

Taking *noun group*s, which consist of more than one word as term, seems to capture more information. In a number of experiments single word terms were replaced by word grams or phrases. However, some studies have shown that (Grantizer, 2000; Martinez, 2002; Joachims, 1998) this did not give a significantly better performance. Thus, it is not considered in the study.

2.6.1.2. Term Weighting

After extracting the term space from a document corpus the influence of each term within a document has to be determined. Therefore each term t_i within a document is assigned a weight w_i leading to the vector representation, $d_j = <w_1, w_2, w_3, \ldots, w_n>$ of a document. The simplest approach is to assign binary values as weights indicating the presence or absence of a term. A more general approach for weighting is counting the occurrences of terms within a document normalized by the amount of words within a document, the so called *term frequency.*

$$freq(tk, \mathrm{Di}) = \frac{\mathrm{occ}(tk, Di)}{\mathrm{N}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

Where, N is the number of terms in D_i and occ(*tk*, *Di*)is the number of occurrences of term *tk* in *Di.*

The term frequency approach seems to be very intuitive, but has a major drawback. For example function words occur often within a document and they have a high frequency, but since these words occur in nearly all documents they carry no information about the semantics of a document. These circumstances correspond to the well known Zipf-law (1932) which states, that the frequency of terms in texts is extremely uneven. Some terms occur very often, whereas as a

rule of thumb, half of the terms occur only once. Similar to term frequencies, logarithmic frequencies as

$$freqlog(tk, Di) = \log(1 + \text{freq}(tk, Di)) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.2)}$$

may be taken, which is a more common measure in quantitative linguistics (Leopold & Kindermann, 2002). Again, logarithmic frequency suffers from the drawback that function words may occur very often in the text. To overcome this drawback, weighting schemes are applied for transforming these frequencies into more meaningful units. One standard approach is the *inverse document frequency (idf)* weighting function which has been introduced by (Salton & Buckley 1988)

$$wk, i = freq(tk, Di) * log \frac{|D|}{|\{Di \epsilon D | tk \epsilon Di|} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \textit{(2.3)}$$

and is know as *Term Frequency Inverse Document Frequency* (*TFIDF*) weighting scheme. TFIDF weighting is the standard weighting scheme within text classification and information retrieval (Frakes & Baeza-Yates, 2002). Thereby *freq(tk, Di) d*enotes the term frequency of term *tk* within document *Di.* D denotes the set of available documents and {*Di* ϵ D | *tk* ϵ *Di*} denotes the set of documents containing term *tk.*

In other words a term is relevant for a document if it:

(i) occurs frequently within a document and

(ii) discriminates between documents by occurring only in few of the documents

For reducing the effects of large differences between frequencies of terms, a logarithmic can be applied to the term frequency leading to

$$wk, i = freqlog(tk, i) * log \frac{|D|}{|\{Di \epsilon D | tk \epsilon Di|} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \textit{(2.4)}$$

2.6.1.3. Dimensionality Reduction

Document indexing by using the above methods leads to a high dimensional term space whereby only a few terms contain important information for the classification task. The dimensionality depends on the number of documents in a corpus, for example the 20,000 documents of the Reuters 21578 data set (Grantizer 2003) have about 15,000 different terms.

Dimensionality reduction is done for the following two reasons:

1. *Computational Complexity:* higher computational costs for classification and training. The learning time of more sophisticated classification algorithms increases with growing dimensionality and the volume of document corpora.
2. *Over fitting*: Over fitting occurs when algorithms classify all examples of the training corpus rather perfect, but fail to approximate the unknown target concept. This leads to poor effectiveness on new, unseen documents. Most classifiers (except Support Vector Machines (T. Joachims, 1998) tend to overfit in high dimensional space, due to the lack of training examples.

To deal with these problems, the following common techniques are suggested (Grantizer, 2003):

i. Dimensionality reduction is performed by keeping only terms with valuable information. Thus, the problem of identifying irrelevant terms has to be solved for obtaining a reduced term space using techniques like term selection and term extraction.
ii. Increasing the amount of training examples.

i. Dimensionality Reduction by Term Selection

A simple dimensionality reduction function is based on the document frequency of a term t_k. According to the Zipf-law, the highest and lowest frequencies are discarded. In this case, predetermined threshold is used to remove words which have document frequency less than or greater than the threshold value.

More technically, we apply dimensionality reduction to reduce the size of |T| to |T'| $\subseteq$ |T|, where |T| is the original term space and |T'| is the reduced term space. This process can speed up the categorization; increase the performance of the classifier with a few percent and time efficiency is more significant. In general, we reduced |T| by disregarding terms that either occurs less than $min_{occurrence}$ times in the entire train document set, or occur more often than a certain threshold in the training set, i.e. if $t_k / |D_{Train}| \geq |max_{freq}|$. By the former process we disregard words that are not significant in the classification, while by the later process we ignore words that are not discriminative enough between categories. The typical values are $min_{occurrence} \in |1 \ldots 10|$ and $max_{freq} \in |0.05 \ldots . 1.0|$ (Wibovo 2002).

ii. Dimensionality Reduction by Term Extraction

Term extraction methods create a new term space T ' by generating new synthetic terms from the original set T. Term extraction methods try to perform a dimensionality reduction by replacing words with their concept. The most common method used in various experiments, is Term Clustering**:** Grouping together terms with a high degree of pair wise semantic relatedness, so that these groups are represented in the Term Space instead of their single terms. Thus, a similarity measure between words must be defined and clustering techniques like k-means or agglomerative clustering is applied (Dhillon et.al. 2002).

2.7. Machine Learning Approach to Hierarchical Text Classification

2.7.1. Introduction

Machine learning is a scientific discipline that is concerned with the design and development of algorithms that allow computers to evolve behaviors based on empirical data such as databases (Witten & Frank, 2005). Machine learning develops computational methods that would implement various forms of learning, in particular mechanisms capable of inducing knowledge from examples or data (Mitchell, 1997). A major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based on data.

Machine learning has a wide variety of application such as computer vision , search engines, medical diagnosis, bioinformatics, detecting credit card fraud, stock market analysis, classifying DNA sequences, speech and handwriting recognition, etc.

Text classification is one of the first applications of machine learning that applies to the general problem of supervised inductive learning. A machine learning algorithm learns the characteristics of the training instances and it uses this knowledge to categorize new instances There are different machine learning algorithms for text classification that help us to automatically categorize texts through learning the structure of the problem domain.. Some of common machine learning algorithms used for automatic categorization that support hierarchical text categorization are explained in section 2.4. This section tries to explain the selected machine learning algorithm and the reasons behind.

A growing number of statistical classification methods have been applied to text categorization, such as Naïve Bayesian, Bayesian Network, Decision Tree, Neural network, Linear Regression, K-Nearest Neighbor, and Boosting. However, most machine learning methods overfit the

training data when many features (high dimension vectors) are given. Therefore, we need to select features carefully. Support Vector Machine (SVM) is robust even when the number of features is large. In this regard, most researches indicate that SVM's have shown good performance for text categorization (Joachim, 1998; Sebastiani, 2002; Vapnik 1992).

The following are good reasons that show SVMs work well for text categorization (Joachims, 1998).

- ⇨ **High dimensional input space**: When learning text classifiers, one has to deal with very many (more than 10000) features using SVM. Because SVM measure the complexity of hypotheses based on the margin with which they separate the data, not the number of features. Hence, SVM is independent of the dimensionality of the feature space, dimensionality reduction techniques are not used in this study.
- ⇨ **Few relevant features**: One way to avoid these high dimensional input spaces is to assume that most of the features are irrelevant. Feature selection tries to determine these irrelevant features.
- ⇨ **Document vectors are sparse**: For each document, the corresponding document vector contains only few entries which are not zero
- ⇨ **Most text categorization problems are linearly separable:** Most data such as all Ohsumed categories, Reuters, etc, are linearly separable. The idea of SVMs is to find such linear or (polynomial, Radial Basis Function) separators.

These arguments give theoretical evidence that SVMs should perform well for text categorization and is selected for this study.

2.7.2. Support Vector Machine (SVM)

SVM is a method for supervised learning, applicable to both classification and regression problems. SVM classifiers creates a maximum margin hyperplane that lies in a transformed input space and splits the example classes, while maximizing the distance to the nearest cleanly split examples (P. Erasto, 2001).

SVM uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane (a decision boundary separating the instances of one class from another). Data from two classes can always be separated by a hyperplane, with an appropriate nonlinear mapping to a sufficiently high dimension. The SVM finds this hyperplane using essential instances from the training set called **support vectors** (Han & Kamber, 2006).

Vladimir Vapnik (1992) and colleagues presented the first paper on support vector machines in 1992. Although the training time for SVMs can be long (since it is computationally expensive and requires extensive memory space), they are highly accurate, owing to their ability to model complex nonlinear decision boundaries. Their use of support vectors for identifying decision boundaries makes them much less prone to overfitting than the other methods. Moreover, since they usually are subsets of the training, the support vectors provide a compact description of the learned model.

SVMs can be used for prediction as well as classification. They have been applied for handwritten digit recognition, object recognition, and speaker identification.

The following two classification problems provide insight on how SVM works: the case when the data are linearly separable and otherwise.

2.7.2.1. Linearly Separable Data

Considering the simplest case of a two-class problem where the classes are linearly separable,

Let the dataset D be given as,

(X_1,Y_1) , (X_2,Y_2),..., , $(X_{|d|}, Y_{|d|})$, where X_i is the set of training instances with associated class labels, Yi.

Each Y_i can take one of two values either +1 or –1, corresponding to the two classes: class-1 and class-2 respectively.

i.e. $Y_i \in \{+1,-1\}$.

Consider an example based on two inputs attributes, *A1 and A2* as shown in Figure2.6.

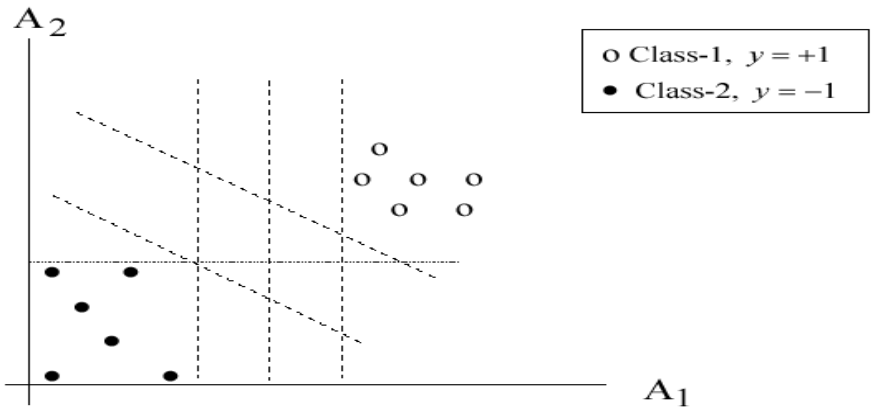

Figure2.6. linearly separable data

From the figure2.6, it can be seen that the 2-D data are linearly separable because a straight line can be drawn to separate all instances of class-1 from all instances of class-2. There are an infinite number of separating lines that could be drawn. The problem is to find the best line that will have the minimum classification error on previously unseen instances. Note that for data with three attributes (3-D data) the problem would be finding the best separating *plan*e. Therefore, in general for n-dimensions the problem would be to find the best *hyperplane.*

A SVM approaches this problem by searching for the maximum marginal hyperplane.

Consider Figure2.7, which shows two possible separating hyperplanes and their associated margins.

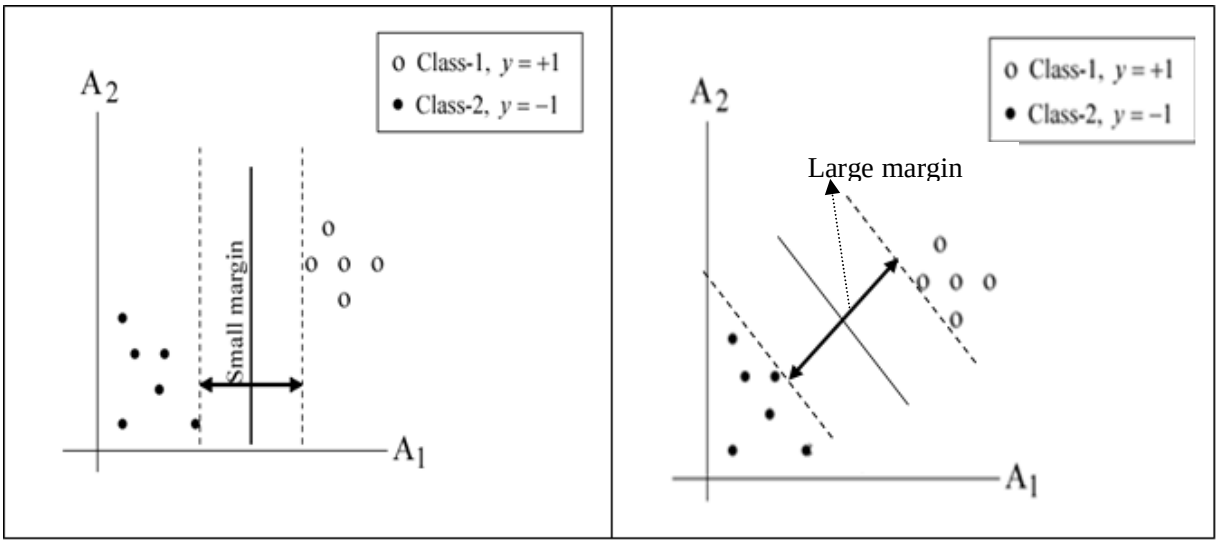

Figure2.7. Two possible hyper planes and their associated margins for the same training data

Figure2.7 reveals that both hyperplanes can correctly classify all the given data instances. The hyperplane with the large margin is however expected to be more accurate at classifying future data instances than the hyperplane with the smaller margin. This is why, during the learning phase, the SVM searches for a hyperplane with the largest margin - the maximum marginal hyperplane (MMH).

A separating hyperplane can be written as

$$W.X + b=0 \quad \text{(2.11)}$$

Where W is a weight vector, namely, $W = \{w_1, w_2, w_3 \ldots w_n\}$ where n is number of attributes, and b is a scalar referred as a bias.

Consider two input attributes, A_1 and A_2, as in Figure2.7. The training instances are 2-D, like $X=(x_1,x_2)$ where x_1 *and* x_2 are the values of attributes ,A_1 and A_2 respectively, for X.

Taking *b* as an additional weight,w_0 , the separating hyperplane in Equation (2.11) can be rewritten as

$$w_0 + w_1x_1 + w_2x_2 = 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.12)$$

Any point that lies above the separating hyperplane thus satisfies the equation

$$w_0 + w_1x_1 + w_2x_2 > 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.13)$$

Similarly any point that lies below the separating hyperplane satisfies

$$w_0 + w_1x_1 + w_2x_2 < 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2,14)$$

The weights can be adjusted so that the hyperplanes defining the two sides of the margin can be written as

$$H_1 = w_0 + w_1x_1 + w_2x_2 \geq +1 \textbf{ for } Yi = +1 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.15)$$

$$H_2 = w_0 + w_1x_1 + w_2x_2 \leq \textbf{-1 for } Yi = -1 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.16)$$

This means any instance that falls on or above H_1 belongs to class-1 and any instance that falls on or below H_2 belongs to class-2

Combining Equation (2.15) and Equation (2.16) one can write

$$Yi(w_0 + w_1x_1 + w_2x_2) \geq +1, \forall \text{ i} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.17)$$

Any training instances that fall on hyperplanes H_1 or H_2 satisfy equation (2.17) and are called **support vector**s. They are equally close to the separating MMH (Han & Kamber, 2006).

Using a Lagrangian[1] formulation and solving for the solution, Equation (2.17) can be rewritten as a constrained convex quadratic optimization problem.

Solving the constrained convex quadratic problem is required to find the support vectors and MMH and thus train the support vector machine. Such trained SVM, are called *linear* SVMs, since the MMH is a linear class. Thus the MMH can be written as a decision boundary, based on the Lagrangian formulation

[1] In mathematical optimization, the method of **Lagrange multipliers** (named after Joseph Louis Lagrange) provides a strategy for finding the maxima and minima of a function subject to constraints.

$$d(x^T) = \sum_{i=1}^{\ell} y_i \alpha_i x_i x^T + bo \quad \text{…….. (2.18)}$$

Where y_i is the class label of support vector x_i

x^T is test instance

b_o are numeric parameters determined automatically by the SVM algorithm

α_i are Lagrangian multipliers and

ℓ is the number of support vectors.

Using the test instances X^T in equation (2.18) is how classification is done by SVMs. If the sign of the result is positive, then X^T falls on or above the MMH, and SVM predicts that X^T belongs to class-1. If the sign is negative, then X^T falls on or below the MMH and the prediction is for class-2 (Han & Kamber, 2006).

The compact prediction model of SVM comes from the fact that the learned classifier is characterized by the number of support vectors rather than the dimensionality of the data. Hence SVMs tend to be less prone to over fitting than some other methods. An SVM with a small number of support vectors can have good generalization, even for a high dimensional data.

2.7.2.2. Linearly Inseparable Data

When the data classes are not linearly separable the approach used for linear SVM can be extended to create *nonlinear* SVMs for the classification of linearly inseparable data. Such SVMs are capable of finding nonlinear decision boundaries (i.e. non linear hypersurfaces) in input space.

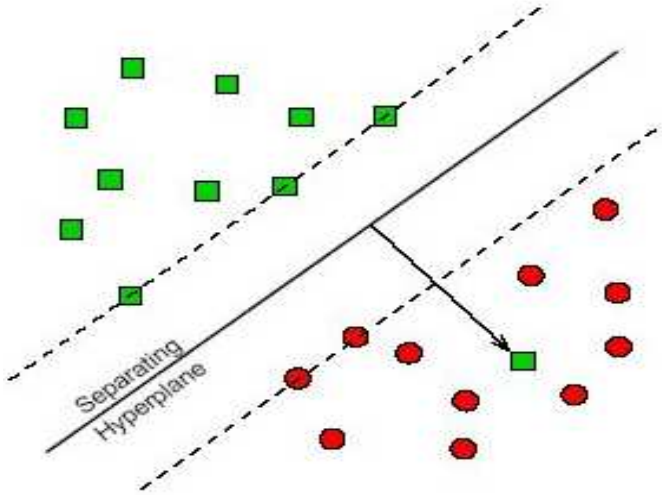

Figure2.8. Non-separable data set

Nonlinear SVM extends the approach for linear SVM using two main steps

a) Transforming the original input data into higher dimensional space using a nonlinear mapping and then
b) Searching for a linear separating hyperplane in the new space. Thus getting a quadratic optimization problem that can be solved using the linear SVM formulation

To solve linearly inseparable problems like the one shown in Figure2.8. SVM allows some flexibility in separating the categories. SVM models have a cost parameter, *C*, that controls the tradeoff between allowing training errors and forcing rigid margins. It creates a *soft margin* that permits some misclassifications. Increasing the value of *C* increases the cost of misclassifying points and forces the creation of a more accurate model that may not generalize well.

The maximal marginal hyperplane found in the new space corresponds to a nonlinear separating hyper surface in the original space.

Considering the following example of transformation of an input data into a higher dimensional space, a 3-D input vector is mapped to a 6-D space

$$\Phi_1(X) = x_1,\ \Phi_2(X) = x_2,\ \Phi_3(X) = x_3,\ \Phi_4(X) = (x_1)^2,$$

$$\Phi_5(X) = x_1x_2,\ \Phi_6(X) = x_1x_3$$

The decision hyperplane in the new space is linear and given as $d(Z)=WZ + b$, Where Z are vectors

Solving the above equation involves choosing a nonlinear mapping to a higher dimensional space and a subsequent costly calculation for the classification of test instant X^T (refer to Equation 2.18). However there is a way of avoiding both.

When searching for linear SVM in the new higher dimensional space, the training instances appear only in the form of dot products (Han & Kamber, 2006),

$$\Phi(Xi).\Phi(Xj),$$

where $\Phi(X)$ is the nonlinear mapping function applied to transform the training instances.

Moreover, applying a *kernel function* $K(X_i, X_j)$ is found to be equivalent to computing the dot product on the transformed data instances, i.e.

$$K(X_i, X_j) = \Phi(X_i).\Phi(X_j) \quad \text{.....(2.19)}$$

Equation (2.19) shows how both nonlinear mapping and calculation on transformed data can be avoided. Afterwards the maximal separating hyperplane can be found in a process similar to linear SVM, though the non-linear SVM involves placing a user-specified upper bound, *C*, on the Lagrange multipliers α_i. This upper bound is best determined experimentally.

Some of the kernel functions that can be used to replace the dot product (See Equation 2.19) are discussed in the next section.

2.7.3. Basic SVM Kernels

SVM is largely characterized by the choice of its kernel, and SVMs thus link the problems they are designed for with a large body of existing work on kernel-based methods. Now the main forms of the kernel function are (Wei-Feng Cao, 2007) liner kernel function, polynomial kernel function, radial basis function and sigmoid function.

i. Linear Kernel

Linear kernel transforms a high dimensional problem into a linear separable problem. This method makes the samples small and linear separable, so that the classifier based on linear kernel has the highest accuracy, as well as the training time is relatively short. Furthermore the linear kernel is easy to perform. Linear kernel is given as

$$k(X_i, X_j) = X_i . X_j \quad \text{.....(2.20)}$$

ii. Polynomial Kernel

When the training set is small, there are many opportunities for polynomial kernel to converge in condition of both high dimension and low dimension. However, the opportunities to converge for polynomial kernel are much fewer than RBF and Sigmoid kernel when the sample is large. It is given by the following function.

$$k(X_i, X_j) = (X_i .. X_j + 1)^h \quad \text{.....(2.21)}$$

where h is the degree of the polynomial

iii. Radical Basic Function (RBF)

For all kinds of samples, both high dimension and low dimension, moreover, both large sample and small sample, there are many opportunities for RBF to converge with a wide convergent area. It is given as:

$$k(X_i, X_j) = e^{-\left\|X_i - X_j\right\|^2 / 2\delta^2} \quad \text{.....(2.22)}$$

where δ^2 is a tuning parameter.

iv. Sigmoid Function

When the feature space is low dimension, both large sample and small sample, the chances of sigmoid kernel to converge are good. Otherwise, there are few chances to converge in condition of high dimension. Sigmoid function is given by:

$$k(X_i, X_j) = \tanh(kX_i.X_j - \delta) \quad \text{.....(2.23)}$$

where k and δ are the parameters.

Since the use of kernels is characterized by the amount of training data taken, the dimensionality of the feature space and the duration of the training time, we used systematic selection of the kernels during our experiment.

2.7.4. SVM Multiclass Classification Methods

One of the disadvantages of the SVM is that, in its original formulation, it is targeted as binary (i.e. two-class) classification problems only. Various approaches have been considered for extending the SVM into the domain of multi-class problems, often at a considerable additional cost of the training. For example, Weston & Watkins (99) proposed an extension of the original optimization problem in which k models are sought simultaneously, where k is the number of classes. This approach does not scale well to problems with many classes.

Most of the other approaches are based on translating the original k-class classification problem into several two-class problems. These approaches are usually not SVM-specific but could use any learning algorithm to train the models for the individual problems. When classifying a new instance, it is shown to the models for these two-class problems and the predictions of these models are then combined into an assignment of the instance to one of the k classes of the original multi-class problem.

The individual two-class problems can be defined in two approaches (Weston & Watkins, 1999). These are;

"One Vs rest" approach: an approach in which there is one two-class problem for each of the *k* classes of the original problem; in the two-class problem corresponding to class i, instances from class *i* are treated as positive and those from other classes are treated as negative. Thus each model tries to predict whether a given instance belongs to that particular class or not.

More specifically, "one vs. rest" approach constructs *k* classifiers. The *i*th classifier output function ***f*** *is* trained taking the examples from class *i* as positive and the examples from all other classes as negative. For new examples ***x,*** this strategy assigns it to the class with larges value of ***f.***

"One Vs one" approach: An alternative is the "one vs. one" approach, in which there is one two-class problem for each pair of classes. Thus for a pair of classes (i, j), where $1 \leq i \leq j \leq k$, the corresponding binary classifier $\boldsymbol{C_{ij}}$ is trained taking the examples from class *i* as positive, those of class *j* as negative, and the rest of the training instances (examples) is not used at all for this particular two-class problem. Thus it constructs k(k-1)/2 classifiers for each distinct pair of classes. For a new example ***x,*** if classifier $\boldsymbol{C_{ij}}$ ***says x*** is in class *i*, then the vote for class *i* will be increased by one. Otherwise the vote for class *j* will be increased by one. After each of the k(k-1)/2 binary classifiers makes its vote, this method assigns ***x*** to the class with the largest number of votes.

The individual problems in one vs one approach are simpler than in the one-vs-rest approach, but the number of models is much larger, k $(k - 1)/2$ rather than just k. For a large number of classes, this approach is infeasible (since it takes time and memory space). This study employs one vs. one method as it is supported by the experimentation tool used in this study (LibSVM).

2.7.5. Training Vs Test Sets

The supervised machine learning approach relies on the availability of an initial corpus of documents pre-classified under categories. Therefore, prior to classifier construction the initial corpus is split in two (training set and test set), not necessarily of equal size. Most researchers use 20% (Han and Kamber, 2006), 30% (Koller & Sahami, 1997) or 33 % (Joachims, n.d) of data as test set and the remaining for training respectively.

The training set is inductively built by observing the characteristics of the documents. In most research settings, once a classifier has been built it is desirable to evaluate its effectiveness. Each document from the test set is fed to the classifier, and the classifier decisions are compared with the expert decisions. The documents in test set cannot participate in any way in the inductive construction of the classifiers; otherwise, the experimental results obtained would likely be unrealistically good, and the evaluation is considered not scientific (Sebastiani, 2002).

2.7.6. Performance Measures

A measure of classification effectiveness is based on how often the classifier decisions match the expert decisions. This, usually, is measured in terms of the classical IR notions of precision and recall, adapted to the case of TC (Sebastiani, 2002). Recall (R) is the percentage of the documents for a given category that are classified correctly. Precision (P) is the percentage of the predicted documents for a given category that are classified correctly. These can be formalized as

R = NCP/NC, and

P = NCP/NP respectively, where NC is the number of testing documents for a given category c; NP is the number of documents that are predicted as category c by the classifier; and NCP is the number of documents that are classified correctly.

Classification accuracy is also the other method of measure of performance represented by *c/n* where *n* is the total number of test instances and *c* is the number of test instances correctly classified by the system (Sebastiani, 2002). Accuracy is the rate of correct predictions made by the model over a data set. This method is used as evaluation measure in this study.

Chapter Three

The Amharic Language and Its Writing System

3.0. Introduction

The recent development of technologies support to produce, store and disseminate information using different language such as English, French, German, Arabic, Chinese, etc. Most organizations store the information in different ways that they can later retrieve. Document categorization is one of the techniques which most of these organizations are using for classifying the huge collection of information into categories based on the similarity or 'likeliness' of the documents.

Amharic is one of the languages through which information can be produced and disseminated in Ethiopia. Ethiopian News Agency (ENA) is an organization that text categorization is implemented in Amharic news items. Though, it is manual, it experiences the use of news classification system to store news into categories such as 'economy', 'politics', 'sport', etc and later retrieve it.

This chapter discusses origins of Amharic script, the Amharic alphabets/characters, numerals and punctuation marks being used; Amharic representations and Amharic writing system problems with the context of building automatic Amharic text classification system.

3.1. Origin of Amharic Language and Its Script

The Ethiopic writing system has its origins in the same ancestral writing systems as those of European alphabets, namely the Semitic scripts that proliferated in the Middle East more than three thousand years ago (Coulmas 1989). As to Coulmas (1989), little is known about the precise timing and location of the emergence of the earliest Semitic phonetic writing system

though speculations abound. All that seems reasonably certain is that a consonantal script developed among Semitic people on the Eastern shore of the Mediterranean sometime between 1800-1300 BC. Coulmas (1989) developed a family tree model of the writing systems that shows two main branches descending from Proto-West Semitic: North Semitic and South Semitic. Among the descendants in the North Semitic branch are Hebrew, Arabic and Greek (and hence Roman and Cyrillic). The South Semitic side is usually held to have produced Ethiopic via the Sabean system, which is speculatively dated as emerging in the 11th and 10th centuries BC, but there are dissenting voices. Bernal (1990), rejecting the family tree model, dates the origins of Ethiopic script earlier, relating it to Thamudic, an older script. The '?' (Question mark) in the family tree model shows controversies between Coulmas and Bernal as to origins of Ethiopic script via Sabean or Thamudic.

The Ethiopic system is used on a large scale in the representation of three Semitic languages, all confined to Ethiopia and Eritrea (the latter being formerly part of Ethiopia but now an independent state). These three languages are Ge'ez, Amharic and Tigrinya. Figure3.1 shows the origins of the language Ethiopic as indicated in Coulmas, 1989

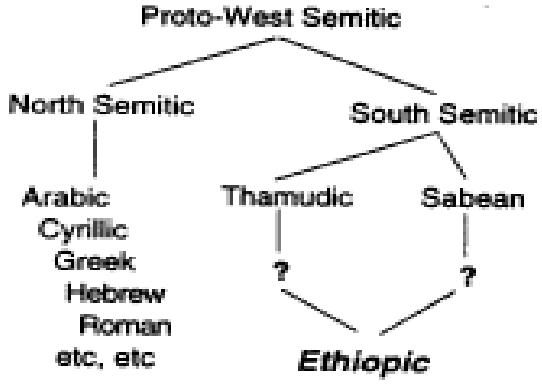

Figure3.1, *ORIGINS OF ETHIOPIC*, a family tree model (adapted from: Coulmas, 1989)

Another hypothesis which supports Coulmas (1989) is given by Bender et al., (1976); this script had developed from the script of Ethiopia's classical language, Ge'ez, which was derived from the Sabean script.

Amharic is written with a version of the Ge'ez script known as **ፊደል** (Fidel). There is no standard way to translate Amharic into the Latin alphabet. Amharic is named after the district of Amhara, which is thought to be the historic centre of the language (Omniglot, 1998). There are other languages which are a member of a family of Semitic languages like Arabic and Hebrew. The writing system difference of Amharic from these languages is that it is written from left to right.

3.2. Amharic Characters/Alphabets

A character or a symbol, Fidel (**ፊደል**) in Amharic, is used to represent a phoneme, which is a combination of a vowel and a consonant (Tewodros, 2003). The Amharic writing system consists of a core of thirty three characters each of which occur in one basic form and in six other forms called orders. The seven orders (the 1st basic form and rest six orders) of the Amharic script represent the different sounds of a consonant-vowel combination known as syllabic. As Worku (2009) discussed, since each character designates a consonant together with its vowel, the vocalic symbol cannot be detached from the consonant element. Thus, Amharic does not use independent symbols for vowels (Bender et.al., 1976). Bender et.al added that the non-basic forms are derived from the basic forms by more or less regular modifications except the last two orders.

Table3.1 shows a sample of a list of Amharic characters (Fidel) showing the seven orders of the alphabet. The whole character set is listed in appendix1.

1st	2nd	3rd	4th	5th	6th	7th
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
hä	hu	hi	ha	he	h	ho
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
lä	lu	li	la	le	l	lo
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
mä	mu	mi	ma	me	m	mo

Table3.1: An example of lists of Amharic alphabets

In addition to these, the Amharic script contains five-so called labio-vellars (table3.2a) which have five orders and 18 additional labialized consonants (table3.2b).

ቈ	ቊ	ቋ	ቌ	ቍ
ኈ	ኊ	ኋ	ኌ	ኍ
ጐ	ጒ	ጓ	ጔ	ጕ
ኰ	ኲ	ኳ	ኴ	ኵ
ጐ	ጒ	ጓ	ጔ	ጕ

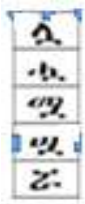

Table 3.2a: Letter variants (labio-vellars) Table 3.2b: Sample lists of labialized consonants

Zelalem (2001) indicates that the Amharic alphabet has around 290 letters and the alphabet/character has no any capital letter and small letter distinction.

3.3. Amharic Punctuation Marks

Pronunciations are usually used to create a sound gap between words or statements in a language. Amharic has its own punctuation marks: some of them are unique to the language and others are borrowed from foreign languages. Amharic has about 17 punctuation marks (Atelach, 2002). Table3.3 shows an example list of punctuation marks used in the language. The complete is listed in an appendix2.

	Mark	Amharic meaning	English meaning	Uses
Punctuations marks unique to Amharic language	:	ሁለት ነጥብ	space	To separate words
	።	አራት ነጥብ	Full stop	To separate single phrases
	፣	ነጠላ ሰረዝ	Comma	To separate single phrases
	፤	ድርብ ሰረዝ	Semicolon	To separate more than single phrases
Punctuation marks borrowed from Foreign languages	?	ጥያቄ ምልክት	question mark	To emphasize a sentence spoken
	!	ቃል አጋኖ	Exclamation	To give to a spoken word, phrase or sentence,
	“ ”	ትምህርተ ጥቅስ	Double quotation	To emphasis ones speech, says, etc

Table3.3 Examples of punctuation marks used in Amharic

3.4. Amharic Number System

Numbers in Amharic consist of single characters for one to ten, for multiples of ten (twenty to ninety), hundred, and thousand (see appendix3). According to Bender et al. (1976), these characters are derived from Greek letters, and some were modified to look like Amharic ‘Fidel’. Each of the symbols has a horizontal stroke above and below. There is no symbol for zero in the Amharic script. Thus, arithmetical computations using the symbols are very difficult, if ever done. As a result, people generally use the Hindu-Arabic numerals. Ethiopic numbers are used mostly in writing dates and page numbers in text (Bender et al., 1976).

3.5. Problems in Amharic Writing System

There are numerous problems observed in the writing system of the Amharic language. Most researches show that these problems are attributed to the very nature of the language (Zelalem, 2001; Atelach, 2002; Surafel, 2003; Yohannes, 2007). Some of these are:

- Amharic is morphologically rich language

- Amharic borrowed most of its scripts from Ge'ez without selecting those symbols that are only necessary for its consonants. As a result there are phonemes with different symbols, where they have meaning in Ge'ez, but their meaning is not known in Amharic (Zelalem 2001)
- The proper use of symbols in Amharic is not studied exhaustively and there is no standard dictionary to refer to.

These and other reasons cause the following problems to appear in Amharic information retrieval tasks in general and classification in particular.

3.5.1. Characters with Different Form

Amharic took the whole Ge'ez alphabet (all seven orders of the 26 symbols of Ge'ez) without considering whether all the 26 characters have meaning in the Amharic writing system. This results in redundancy of characters where more than one symbol is used for a given sound. The following table shows an example for this problem.

Consonants	Other symbols with the same sound
ሀ (hä)	ሃ ሐ ሓ ኻ ኀ
ሰ (sä)	ሠ
አ (ä)	ኣ ዐ and ዓ
ጸ (tsä)	ፀ

Table3.4. *some lists of Amharic characters with the same sound* (taken from: Yohannes 2007)

Zelalem (2001) recognized these problems creating spelling variations of a word, which would unnecessarily increase the number of words representing a document which could reduce the efficiency and accuracy of the classifiers. Amharic document processing for feature selection

should therefore normalize word variants (spelling differences) caused by use of different characters which have the same sound. This study employs this problem by changing word variations into one common form (see algorithm 4.1 in section 4.2.1.1).

The Word in English	The Word in Amharic	Spelling Variants of the Word
Work	ሥራ	ስራ
Sun	ፀሐይ	ጸሐይ፣ ፀሀይ፣ ጸሀይ
World	አለም	ዐለም፣ ዓለም፣ ኣለም
Power	ሀይል	ሃይል፣ ኃይል

Table3.5. Examples of the different word spellings (taken from: Zelalem 2001).

3.5.2. Compound Words Usage

In the Amharic writing system, inconsistency is often observed regarding the representation of compound words. Some compound words are used as a single word in some instances (either by fusing the two words or by inserting a hyphen between them) and as two separate words at other instances. According to Yohannes (2007) this problem would cause high dimensional spaces; i.e. compound words could result in redundant word features by creating more words when a compound word (example **አዲስ-አበባ**) is treated as two separate words **አዲስ** and **አበባ**. In the other hand, it may also result in a semantic loss by confusing a document about the city Addis Ababa (**አዲስ-አበባ**) with the one talking about the floral industry. Such kind of inconsistencies might happen in Amharic documents (for instance in ENA data). Thus, this problem should be handled in the document preprocessing (see algorithm 4.3 in section 4.2.1.3).

Table3.6 shows some examples of the problems caused by the use of compound words in Amharic writing system.

Compound words used as single word	Literal English meaning	Compound words used as two separate word	Literal English meaning
ባሕርዳር	Bahirdar	ባሕር ዳር	Sea shore
ቤተሙከራ (ቤተ-ሙከራ)	Laboratory	ቤተ ሙከራ	House of Experimentation
ፀረኤድስ (ፀረ-ኤድስ)	Anti-AIDS	ፀረ ኤድስ	Anti AIDS
ቀዶጥገና (ቀዶ-ጥገና)	Surgery	ቀዶ ጥገና	Cutting and fixing

Table3.6. Inconsistencies caused by compound words

3.5.3. Transliterations Problem

Transliteration from foreign words to Amharic words is also another problem. The problems resulted from use of loan words that are borrowed from other languages and that do not possess their own translation in Amharic. The word “Oxford” may be transliterated as **ኦክስፈርድ** (oxferd) or **ኦክስፎርድ** (oxford) (Bender et al., 1976). Table3.7 shows translation of the word ‘meteorology’ in to different Amharic spelling (adapted from: Yohannes 2007).

Foreign Word	Equivalent Words in Amharic usage
Meteorology	ሚትሪዮሎጂ፣ ሚትዎሮሎጂ፣ ሚትዮሮሎጂ፣ ሚቲዎሮሎጂ፣ ሚቲዎሮሎጅ፣ ሚቲዎሮሊጂ፣ ሜትዎሮሎጂ፣ ሜትሮዎሎጂ፣ ሜትሮዎሎጅ፣ ሜትሪዎሎጂ፣ ሜትሮሎጂ፣ ሜትርዎሎጂ፣ ሜቲዎሮሎጂ፣ ሜቴዎሮሎጂ

Table3.7.Word variations due to transliterations

Another form transliteration problem is abbreviation translation. It can result in different variation of the word. For example A.D can be translated in to **ዓ.ም**, **ዓ.ም**., and **ዓም**.

In this research, such kinds of problems are resolved by finding common word for different variation of words.

3.6. Amharic Unicode Representation

The Unicode Standard is the universal character encoding scheme for written characters and text (Unicode Consortium v.3.0). It defines a consistent way of encoding multilingual text that enables the exchange of text data internationally and creates the foundation for global software.

The Unicode Standard specifies a numeric value and a name for each of its characters. In this respect, it is similar to other character encoding standards from ASCII onward. In addition to character codes and names, other information is crucial to ensure legible text: a character's case, directionality, and alphabetic properties must be well defined.

Unicode supports two encoding forms: UTF-16 and UTF-8. UTF-8 has been designed for ease of use with existing ASCII-based systems. UTF-16 used to encode more than 65, 000 characters (semantic and surrogate information in addition to UTF-coded characters)

The current standard, Unicode Standard, Version 3.0, contains 49,194 characters from the world's scripts. These characters are more than sufficient not only for modern communication, but also for the classical forms of many languages of the world.

Many new scripts and characters have been added in Version 3.0, including Ethiopic, Canadian Aboriginal Syllabics, Cherokee, Sinhala, Syriac, Myanmar, Khmer, Mongolian, Braille, and additional ideographs.

Since, the Ethiopic script was not included in ASCII in earlier times, there were a significant problem representing and manipulating Amharic characters. However, now Unicode includes the Ethiopic scripts (Amharic and others), it makes easier to manipulate and process Amharic documents.

Chapter Four

Experiment and Discussion of Results

4.0. Introduction

This chapter presents the process of experimentation, results and discussions. The pre-processing and the experiments are performed based on the concepts discussed in the previous chapters. More specifically, the efforts made while undertaking the whole experiment up to building the model/classifier is explained in the following architecture (Figure4.1).

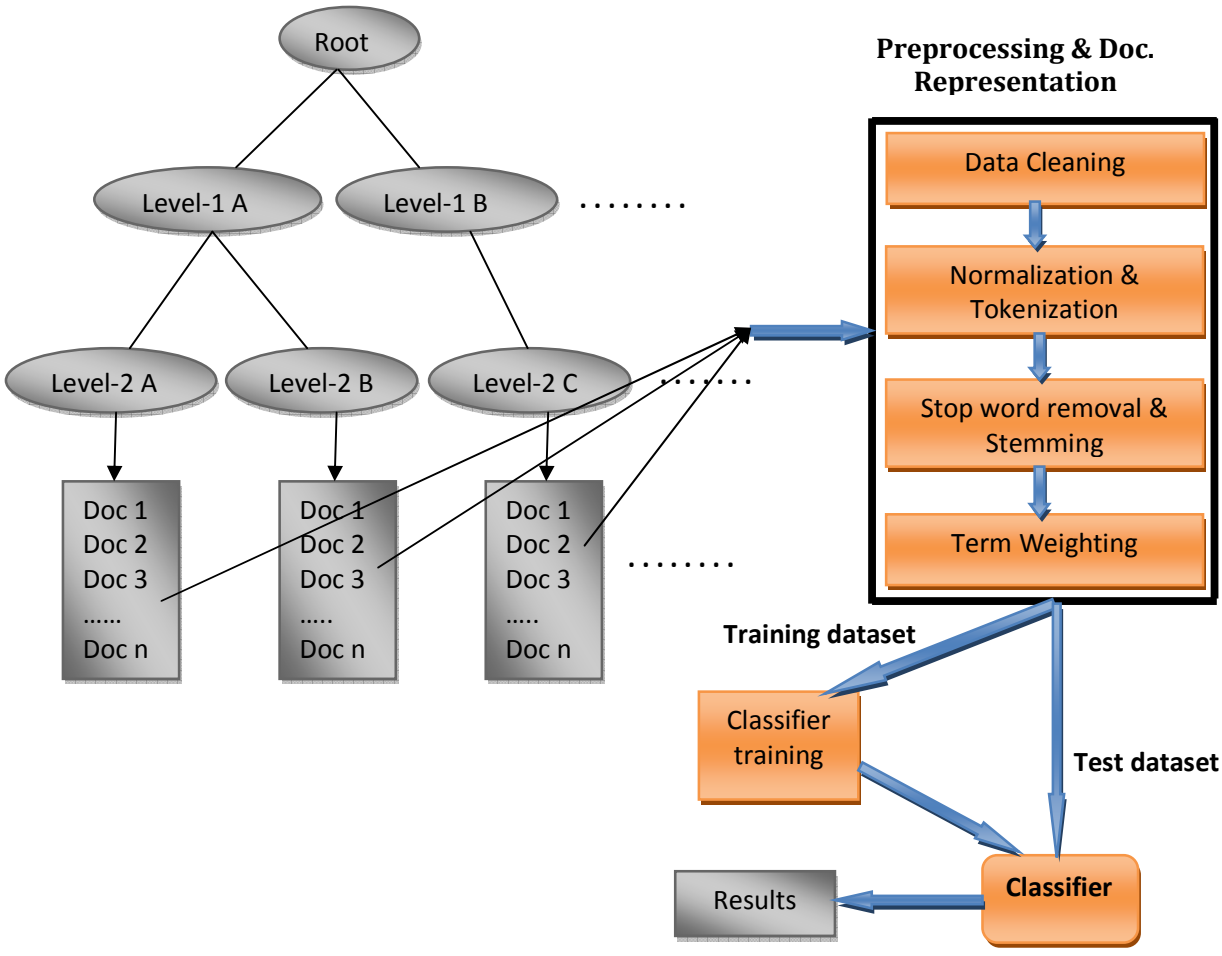

Figure4.1. Architecture of the Hierarchical Amharic News Text Classifier

As shown in figure4.1, the classifier has three elements: preprocessing & document representation, classifier building, and Performance evaluation through testing.

Preprocessing is the process of making the document ready for modeling building and testing. It includes text processing methods such as normalization and tokenization, stop word removal & stemming and term weighting. Training and test data is then prepared according to the format required by the tool, LibSVM.

After the data is made ready for experimentation, the training data set has been fed to the learning algorithm implemented in LibSVM so that the classifier (model) would be built. Finally, the classifier performance evaluation is made using accuracy test. The remaining sections discuss each steps of the experiment in detail.

4.1. Data Pre-processing

4.1.1. Data Source

The data source for the study was the Ethiopian News Agency (ENA). ENA produces news in two languages: Amharic and English. The News prepared in Amharic was the data used for this study. ENA uses software named ENASoft to categorize news items into categories. The title of the document, slug, keyword and place where the news happened are the attributes thorough which a document can be categorized.

The total number of Amharic news items collected was 16075 from 2006-2010. A three year data from 2007-2010 is considered in this study, because it is the only data stored in ENA in a hierarchical manner. Moreover, since not all the contents of the document are important for representing the document (Sebastiani, 2000), keyword, slug, title, and the first 350 characters of the news were taken.

The main problem that appears in data collection was the format that the data was published with. Each news item was published in html format. Since LibSVM (the experimentation tool) deal with data in text format, the data should be first changed into text format. However, there is no any software that changes the data from html format to text format for Amharic language. Thus, the data were converted into text format manually, which was a bit troublesome during conducting the study.

4.1.2. Data Cleaning

Real world data are usually incomplete, noisy, redundant or inconsistent. Data cleaning is an attempt to fill *the missing values*, smooth out *noise* remove, *redundancies* and correct inconsistencies (Witten & Frank, 2005). Text classification is also suffers from these problems. Hence, it necessitates data cleaning prior to further processing. Since the data was converted manually, it makes easier to identify and remove the following types of documents:

- Documents redundant in the same category
- Documents which were written in both Amharic and English
- Documents which have no title and keyword.
- News items with missing category labels

ENA uses 12 major categories and 98 sub categories to categorize news items. The categories, sub categories and number of documents are shown in table4.1.

No.	Major class	No. of Subclass	No. of documents (2007-2010)
1	Culture and Tourism (**ባህልና ቱሪዝም**)	9	791
2	Economy (**ኢኮኖሚ**)	11	1134
3	Education (**ትምህርት**)	14	1117
4	Health (**ጤና**)	11	1067
5	Law and Justice (**ህግናፍትህ**)	6	704
6	Politics (**ፖለቲካ**)	9	1136
7	Social (**ማህበራዊ**)	11	1064
8	Sport (**ስፖርት**)	7	523
9	Accident (**አደጋ**)	3	2102
10	Weather and Environmental Protection (**የአካባቢ ጠበቃና የአየር ሁኔታ**)	5	2203
11	Relations, Defense, and Security (**ግንኙነት፣ መከላከያና ደህንነት**)	8	2017
12	Science and Technology (**ሳይንስና ቴክኖሎጂ**)	4	1150
	Total	98	15008

Table4.1. Statistics of data collected from ENA (2007-2010)

As shown from table4.1, the data was a two level category, where documents were assigned in sub categories. Thus, based on the concept in DAG (section 2.4.2), the sub categories in ENA were used as leaf nodes (last category) in the hierarchy where the actual news items are assigned. However, this study needs to have more than two levels of categorical data to achieve its defined purpose. Thus, it is necessary to bring up these sub categories to have parent class. In this case, these categories are further categorized to obtain more than two levels of a categorical data. The parent categories where one or more sub categories are assigned to were given a descriptive name by the researcher in consultation with domain expert.

The following techniques are used to do the above process:

2. **Expert judgment**: experts from ENA were asked to give their own judgment on how to further categorize sub-categories in to one or more parent classes.
3. **Document-similarity matrix**: documents in each subcategory are represented using vectors. Term weight is calculated for each term. Then the cosine similarity method is used to calculate the similarity among document in each category. Cosine similarity produces a value between [-1, 1] and 0.5 was used as a threshold document similarity value between those sub-categories. The following table shows an example of how sub categories were categorized up into one category to obtain more levels of categorical data.

	CMH	DPT	OD	MTH	HCD	HP	HS	MD	DH	TT
CMH	1	0.691	0.53	0.64	0.05	0.35	0.14	0.13	0.191	0.34
DPT[2]	0.779	1	0.98	0.73	0.196	0.096	0.45	0.44	0.08	0.06
OD	0.822	0.7	1	0.907	0.02	0.153	0.067	0.105	0.11	0.117
MTH	0.54	0.95	0.564	1	0.1	0.55	0.261	0.232	0.5	0.088
HCD	0.254	0.321	0.262	0.096	1	0.863	0.942	0.683	0.48	0.262
HP	0.122	0.06	0.21	0.08	0.85	1	0.8	0.98	0.14	0.147
HS	0.08	0.392	0.1	0.519	0.602	0.935	1	0.57	0.16	0.08
MD	0.042	0.09	0.08	0.04	0.3	0.795	0.544	1	0.155	0.088
DH	0.151	0.33	0.236	0.11	0.047	0.233	0.121	0.3	1	0.501
TT	0.013	0.022	0.01	0.01	0.003	0.017	0.01	0.021	0.823	1

Table4.2. Document-similarity matrix for class “Health”

Collecting nodes which have similarity value greater than 0.5 and the values that lie in shaded one in table 4.2, we can get the following clusters,

1. **Category one**

   Given category name: “Disease Prevention and Treatment”

   Category code: 4.1

   Level: level 2

[2] DPT (Disease Prevention & Treatment) indicates a combination of two subcategories (‘Disease Prevention ‘and ‘Disease Treatment’) which were counted as two separate class in ENA

	CMH	DPT	OD	MTH
CMH	1.0	0.691	0.530	0.64
DPT	0.779	1.0	0.98	0.73
OD	0.822	0.7	1.0	0.907
MTH	0.54	0.95	0.564	1.0

Key:
- CMH: Children's' & maternity Health
- DPT: Disease Prevention & Treatment
- MTH: Malaria, TB & HIV-AIDS
- OD: Other Diseases

Table4.3.a The first cluster containing 4 classes

2. **Category two**

   Given Name: "Drugs and Pharmatituicals"

   Category code: 4.2

   Level: level 2

	DH	TT
DH	1.0	0.501
TT	0.823	1.0

Key:
- DH: Drugs and Heroines
- DT: Traditional Treatment

Table4.3.b. the second cluster containing 2 classes

3. **Category three**

   Given Category name: "Health Development"

   Category code: 4.3

   Level: level 2

	HCD	HP	HS	MD
HCD	1.0	0.863	0.942	0.683
HP	0.850	1.0	0.80	0.98
HS	0.602	0.935	1.0	0.570
MD	0.3	0.795	0.544	1.0

Key:
- HCD: Health Center Development
- HP: Health Professionals
- HS: Health Services
- MD: Medical Devices and materials

Table4.3.c.The third cluster containing 4 classes

Therefore, generation of more than two levels categorical data were done with the help of experts' judgment and cosine similarity value as shown in table4.2 among documents between categories. Those categories which have below 0.5 similarity values and decided as they should be put as a separate class on its own by the expert were removed.

Document similarity matrix was calculated after all preprocessing of documents were done. 8-class items with 5100 documents are generated as the result of this process. Table4.4 shows an example of hierarchical data generated using the above process. The complete table is sown in appendix5.

Level-0	**Level-1**	**Code**	**Level-2**	**code**	**Level-3(Leaf node)**	**Code**	**No. of dcts**
R O O T	Tourism and Culture	1	Religion and Holiday	1.1	Religious & national holidays	1.1.1	109
					Religious conferences and forums	1.1.2	105
			Tourism Attractions and Development	1.2	Arts	1.2.1	79
					Heritages	1.2.2	110
					History	1.2.3	79
					Nations and Nationalities of Peoples	1.2.4	84
					Tourism Development	1.2.5	91
					Tourists	1.2.6	92
	Health	4	Disease Prevention & Treatment	4.1	Children's' & Maternity Health	4.1.1	34
					Disease Prevention and Treatment	4.1.2	135
					Malaria, TB &HIV-AIDS	4.1.3	64
					Other Diseases	4.1.4	18
			Drugs & Pharmatituicals	4.2	Drugs & Heroines	4.2.1	30
					Traditional Treatment	4.2.2	17
			Health Development	4.3	Health Center Development	4.3.1	150
					Health Professionals	4.3.2	17
					Health Services	4.3.3	90
					Medical Devices & Materials	4.3.4	14

Table4.4: Example of hierarchical level for classes ‘’Tourism & Culture” and “Health”

4.2. Representing Documents and Classes

Document representation is a process of identifying terms, also called index words which can distinguish one document (class) from the other. The purpose of this section is identifying terms from training documents such that each class can be represented with the appropriate terms (class representatives) of that class. The following sections discuss the processing methods.

4.2.1. Normalization and Tokenization

4.2.1.1. Changing spelling variation of the same sound word to one common form

Spelling variation of a word in a document is one of the problems in text classification. However, these different symbols must be considered as equivalent as they do not cause changes in meaning. As a result, to resolve this problem, a replacement approach has been followed. For example, if the character is one of '**ሀ**', '**ሐ**', '**ኀ**', '**ኘ**', '**ኻ**', or '**ኸ**' (all of them with similar sound h), then it is converted to one common form '**ሀ**'. The same rule applies to all orders of '**ሠ**', '**ዐ**' and '**ፀ**' are converted to the corresponding orders of '**ሰ**', '**አ**' and '**ጸ**' respectively.

Read a character
    If the character is one of **ሐ** or **ኘ** or **ኸ** replace them with **ሀ**
    (The same applies for the orders of will be replaced by the corresponding orders of **ሀ**)
        Return the replaced character
    If the character is **ሠ** replace them with **ሰ**
    (The same applies for the orders)
        Return the replaced character
    If the character is **ፀ** replace them with **ጸ**
    (The same applies for the orders)
        Return the replaced character
    If the character is **ዐ** replace them with **አ**
    (The same applies for the orders)
        Return the replaced character
    If the character is **ጐ** replace them with **ጎ**
        Return the replaced character
    If the character is **ኩ** replace them with **ኰ**
        Return the replaced character
    If the character is **ዉ** replace them with **ው**
        Return the replaced character

Algorithm4.1. Character replacement algorithm (adapted from: Tessema, 2007)

4.2.1.2. Removal of unnecessary Variables

The numbers, unnecessary spaces, garbage values created when a document are read and control characters in the text of each file are not considered for classification as they do not have contribution to discriminate the classes. Words containing numbers like (2nd i.e. 2ኛ or ENA103862) and unnecessary spaces that exists between words and garbage values created (like '\r\n', '\ufeff') are excluded at the first phases of preprocessing.

Moreover, punctuation marks are also common in Amharic documents. Punctuation marks are used to separate words, phrases or used to indicate the end of a sentence. They are found in a sentence attached with the word or found with space in between. As explained in section3.3, their presence or absence discriminate the similarity of the word in a sentence or the entire document. For example if the words with punctuation mark '**ግብርና፣**' to mean (agriculture) is not the same as the word with no punctuation mark '**ግብርና**' in calculating the ***tf*** value for the word. Hence, removing punctuation marks help to obtain more similar words. Hence, the standard control character; Amharic punctuation marks (፣,፤,፥,፦) ; and symbols borrowed from English language (?,!,",", ',', ',|,/) are removed. The following algorithm is used to remove extraneous variables from a document(s) as part of the normalization process.

```
Read file
For each character in file
        If character in unnecessaryC harchterList then
        Remove the character from file
        End if
End for
```

Algorithm4.2. Extraneous variable removal algorithm

4.2.1.3. Combination of Co-occurring Words

There are words in documents which should appear together like ‘’**አውዱ- ርዕይ**’’, “**ኪነ-ጥበብ**”, etc. These words should not be taken separately one from another; otherwise it will lose the concept. To overcome the problem, such words are concatenated to form a single word. The following algorithm is used to combine such kinds of words with each other. The complete list is shown in appendix7 (taken from Ellele Amharic-English-Affan Oromo dictionary).

```
Read file
For each word in an unsorted file
        If word is in SpecialWordList then
                Combine word with next word
        Else
                Continue
        End if
End for
```

Algorithm4.3. Word combination algorithm

4.2.2. Stop Word Removal

Stop words are non-content bearing words which are common in many documents. Thus, stop word removal is a necessity in text classification purposes. The assumption is that words which occur most frequently in almost all documents are non-informative. Hence, 2100 Amharic stop words are identified. In this research, two kinds of stop words are prepared; one which are common to Amharic language text and the other which are related to the domain under study (Amharic News Items).

Like English language, some words in Amharic are used very frequently in the normal usage of the language such as ‘**ነው**’, ‘**ሆነ**’, ‘**ነበር**’, ‘**ጋር**’, ‘**ሆኖም ግን**’, etc. Such words are identified for the removal process.

Moreover, it is usual to see news text that is full of some common words that occur frequently and used in almost all items. For instance, the words '**አካሂዱ**', '**አስታወቀ**', '**ተገለፀ**', '**ገለፁ**' ,etc, frequently occur in most Amharic news texts. Hence, news specific common words of this type are used as stop word list.

Such stop words are saved as file, and the filename is provided to the developed program which is capable of reading the file and removing the stop words from each document.

4.2.3. Stemming: Affix Removal

Stemming is used to reduce variants of same word to common stem. A single Amharic word has different forms from context to context. For example, words like '**ሰዎች**', '**ሰዎቹን**', '**ሰዎቻቸውን**', '**የሰዎች**', etc should be stemmed into a root word '**ሰው**' using stemming.

Nega (2000) has developed a stemming algorithm for Amharic language. Despite the effort made, the algorithm could not be found. Hence, a stemming algorithm was developed in collaboration with research group members[3] which considers only prefix and suffix removal.

4.2.3.1. Prefixes Considered

Prefixes such as '**ከ**', '**በ**',' **የ**',' **ስለ**', '**እንደ**', '**እንደየ**', '**ለ**', etc (Zelalem, 2001) are considered in this study . Such kinds of prefixes are removed from words except where these characters are not used as prefix such as '**የመን**', '**ከብማ**', '**በጀት**', etc. Such kinds of words are automatically detected from the corpus (news item) and manually inspected and inserted into the exception list.

[3] The stemming algorithm was done in collaboration with Zeleke Abebaw & Alemu Kumilachew.

4.2.3.2. Suffix Considered

Suffixes are characters attached at the end of the word. Like English, Amharic is also exposed to a variety of suffixes depending on the context of the sentence. There are three kinds of suffixes identified in this study. The first once are suffixes which pluralize a noun or a phrase in a statement such as '**ዎች**' and '**ኦች**'; the second once are those suffixes which comes at the end of a verb, noun and phrase such as '**ና**', '**ን**', '**ም**', '**ው**', '**ንና**', '**ንም**', etc; the third once are suffixes which shows plural possession such as, **ዎቻቸው**, **ዎቻችን**, etc. Two ways are used to remove these suffixes. First, the last one, the last two, the last three, the last four, and the last five most characters of the word are assigned to a different variable respectively. The basic assumption[4] here is the maximum length of suffixes may not greater than 5. Then if the word is checked whether it ends with one of the contents of the five variables, then they are removed form the word provided that the word is not in the exceptional list such as '**ዘመን**', '**ግብርና**', '**ስላም**', etc, where the contents of these variables are not actually a suffix.

Secondly, if the variable contains the suffix '**ዎች**', the word '**ዎች**' is directly removed from the word to be stemmed provided that the last character of the stemmed word is different from 6th order ('Sades') of the character. For example, if the suffix word is removed form the word '**ኮማሪዎች**', it resulted in the word '**ኮማሪ**', since the last character of the stemmed word (**ሪ**) which is different from 6th order of the character '**ረ**', i.e. '**ር**'. In this case '**ዎች**' is directly removed and the word '**ኮማሪ**' is used as a stem word. However, direct removal of the suffix has exceptions to words which end with '**ዉ**' (for example '**ሰው**', '**ሰዎች**'). When the suffix '**ዎች**' is removed, it resulted with character '**ሰ**'. In this case, '**ሰ**' is combined with the character '**ው**' and used as a stem word.

In the other hand, if the word ends with '**ኦች**', it is directly removed and the 6th symbol of the first character of '**ኦች**' is added to the stemmed word. For example, in the word '**ዛፎች**', when

[4] An experiment was tried in lab to know the average length of suffixes for sample Amharic Words by taking their plural and plural possession of a word. The average length was 4. This helps us to assume a maximum length could not be greater than one step higher than the average length, 5.

‘ፎችኑ’ is removed, the first 6th symbol of the first character of ፎ, which is, ‘ፍ’ is added to the stemmed word/symbol (e.g. ‘ዛ’), which will be ‘ዛፍ’. The following is the algorithm used for prefix and suffix removal.

```
Read tokens
        For each token in token list
                If token starts with prefix
                        If token not in exceptional list then
                                Remove prefix
                        End If
                End If
        End for
        For each token in token list
                Assign the last one, two, three, four, and five most character of the token to a
                variable.
                If token ends with either one of contents of a variable
                        If first char of suffix different from ‘Sades’
                                If token not in exceptional list then
                                        Remove suffix
                        Else
                                If token not in exceptional list then
                                        Normalize a stem word
                                        Remove suffix
                        End If
                End if
        End for
Update token list
```

Algorithm4.4. Affix removal algorithm

The result of previous experiments is a set of representative terms used to calculate term weights for the classification task. Hence, in a tokenization process, words which have less contribution in representing and discriminating news have been eliminated. Words including numbers which match to the stop words list are removed. Moreover, as a method of enhancing representation, stemming is also used to bring words with similar concept but varying characters due to grammatical usage of Amharic; that would otherwise be treated differently. . As a result, the features are reduced from 297981 to 189877. The following table shows the results of features generated through text preprocessing.

Category	No. of tokens	Normalization & Stop word removal			Stemming			All preprocessing		
		FN	RN	R%	FN	RN	R%	FN	RN	R%
Culture& Tourism	24325	18176	6149	25.28	20075	4250	17.47	15834	8491	34.91
Economy	27242	20753	6489	23.82	21534	5708	20.95	17352	9890	36.3
Educatio n	51223	41271	9952	19.43	43531	7692	15.02	31770	19453	37.98
Health	42277	33062	9215	21.8	39531	2746	6.5	26348	15929	37.68
Law & Justice	22307	16185	6122	27.44	18937	3370	15.11	14961	7346	32.93
Politics	61459	51321	10138	16.5	51778	9681	15.75	38690	22769	37.05
Social	50463	41829	8634	17.11	42524	7939	15.73	32234	18229	36.12
Sport	18685	13648	5037	26.96	15499	3186	17.05	12688	5997	32.1
Total	297981	236245	61736	20.72	253409	44572	14.96	189877	108104	36.28

Table 4.4.Normalization, Stop word, stemming experiments for feature reduction

Table4.4 shows the number of features resulted after each processing, FN is the number of features after each preprocessing, RN is the number of reduction of features from the total tokens generated during tokenization, R% is the reduction of features in percentage.

As it can be observed from the table that normalization and stop word removal reduces features better than other preprocessing methods, which is 20.72% for all categories. Stemming reduces features by 14.96%. When all the preprocessing is done together, 36.28% of features are reduced for all categories.

More over, a sample of 500 tokens is randomly selected from a set of training documents in each class to test the efficiency of the normalizer, the stop word removal, and the stemmer. Thus, these tokens are manually counted and put into three groups; tokens which need *normalization*, *stop word removal*, and *stemming*. Therefore, a total of 289 tokens which need to be stemmed; a

total of 101 tokens which stop word removal process should be applied; and a total of 307 tokens which need to be stemmed are identified. The manual inspection of the result after each of the algorithms is applied on each of the corresponding tokens is shown in the following table.

Algorithm	No of tokens given to (a)	No. of correctly preprocessed tokens (b)	Accuracy (b/a)
Normalizer	289	271	93.77%
Stop word removal	101	81	80.2%
Stemmer	307	194	63.2%

Table4.5: Accuracy of a normalizer, stop word removal and stemmer on 500 sample tokens

As it can be seen from table4.5, the efficiency of the algorithms looks like the result shown in the fourth column. Most of the incorrectly preprocessed tokens come from a writing font difference between comparing characters during normalization; unnecessary existence of bad characters with the token during stop word removal; and problem of pluralization during stemming.

4.2.4. Term Weighting

The next step is term weight preparation using the features generated as the result of the previous experiment (as shown in table4.4). Thus, term weights are prepared based on the concepts discussed in section2.6.1.2 according to the requirements of the tool selected for experiment.

4.3. LibSVM: Experimentation Tool

LibSVM (a Library for support Vector Machine) is integrated software for support vector classification, regression and distribution estimation. LibSVM [multiclass] is an implementation of the multi-class Support Vector Machine (SVM). For a training set (x_1,y_1) ... (x_n,y_n) with labels y_i in [1..k], it finds the solution of the following optimization problem during training.

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned}$$

For all y in [1…k]……. (4.1)

Where C is the usual regularization parameter that trades off margin size and training error (avoids misclassification error) and ξ_i is misclassification error in linearly non-separable case. Misclassification takes place when $\xi_i > 1$.

The above optimization defines the optimal hyperplane to be the hyperplane that maximizes the geometric margin and minimizes the functional misclassification error (ξ_i)

Here the training vectors x_i are mapped into a higher dimensional space by a function . Then SVM finds a linear separating hyperplane with the maximal in the higher dimensional space. C>0 is the penalty parameter of the error term. Furthermore, k(*xi*,*xj*) (*xi*)T is called the kernel function. SVM used four basic kinds of kernels for mapping training instances into high dimensional space as their distinctions are discussed in section2.7.3.

We used linear kernel and c=1 in all our experments as it gives best results during our experments

4.3.1. Input File Preparation

The term weight prepared in section 4.2.4 is used to prepare the input file using LibSVM. The input file contains the training examples and testing examples with the same file format. LibSVM requires the following format to prepare the input file:

[line].=. [label] [index1]:[value1] [index2]:[value2] ...

[line] .=. [label] [index1]:[value1] [index2]:[value2] ...

…….

One record per line, as:

+1 1:0.708 2:1 3:1 4:-0.320 5:-0.105 6:-1

Where,

Label (target value): Sometimes referred to as 'class', the class (or set) of classification and its values are integers.

Index (feature number): Ordered indexes. Usually continuous integers

Value (feature value): The data for training which contains term weights.

The target value and each of the feature/value pairs are separated by a space character. Feature/value pairs must be ordered by increasing feature number. Because LibSVM supports only numerical data, the words were changed to numerical.

Each line represents one training example in a structure described above. It means, we have an array (vector) of data (numbers): value1, value2, ….. valueN (and the order of the values are specified by the respective index), and the class (or the result) of this array is label.

Value1, value2,….valueN is usually the input data to the problem under study which involves lots of 'features', or 'attributes'; so the input will be a set (or say vector/array). If we take two points in the X and Y plane, it is assumed that each point has coordinates X and Y so it has two

attributes (*x* and *y*). For example, to describe two points (0,3) and (5,8) as having labels(classes) 1 and 2, we will write them as:

1 1:0 2:3
2 1:5 2:8

And 3-dimensional points will have 3 attributes and so on.

LibSVM uses the so called "sparse" format where zero values do not need to be stored. Hence, this kind of file format (representation) has the advantage that we can specify a sparse matrix. For example, if we have data with attributes

1 0 2 0

is represented as

1:1 3:2

Therefore, all the files (documents) in each category are done automatically in a similar way.

4.3.2. Running LibSVM

SVM^{multiclass} consists of a learning module (svm_multiclass_learn) and a classification module (svm_multiclass_classify). The learning module takes the files to be learned. It learns the characteristics of the data and develops the model or classifier. It has the following format:

svm_multiclass_learn [options] training_example_file model_file

Where,

- *svm_multiclass_learn* is the learning module.
- *options* are the kernel functions and their parameters given to learning module to train the example file. Linear kernel and C=0.01 are some of the default values in LibSVM
- *training_example_file* is a file containing training instances (a file to be learned)
- *model_file* is the learned rule generated by the classification module using the selected parameters

In the other hand, the classification module can be used to apply the learned model to new examples. It has the following format:

svm_multiclass_classify [options] test_example_file model_file output_file

where,

- *svm_multiclass_classify* is the classification module
- *options* are functions and parameters
- *test_example_file* a file containing test instances (a file used to test a learned model)
- *model_file* the learned rule generated by the classification module on which the *test_example_file* is tested
- *output_file is a* standard output of a classification/prediction result

For all test examples in test_example_file the predicted classes (and the values of $x \bullet w_i$ for each class) are written to output_file. There is one line per test example in output_file in the same order as in test_example_file. The first value in each line is the predicted class, and each of the following numbers is the discriminant values for each of the k classes. For example, given a testing file,

2 1:1.08889 2:2.1978 3:0.9634

the *output_file* results the following output

Class		1	2	3
Prediction	2	-0.067107	0.112766	-0.045659

SVM compares the prediction of each class and then the class with the maximum value is assigned to the test file. Thus, the above example shows that the test example is correctly predicted as class 2 among 3 classes.

4.3.3. Performance Measures

After models are trained by solving the above optimization problems, users can apply LibSVM to predict labels (target values) of testing data. Let $x1,\ldots,xn$ be testing data and $f(x1),\ldots, f(xn)$ be LibSVM's predicted decision values (target values for classification). If the true labels (target values) of testing data are known and denoted as $y1,\ldots, yn$, the predictions are evaluated by the following measures:

$$\text{Accuracy} = \frac{\#\ correctly\ pridicted\ data}{\#\ total\ test\ data} \times 100\% \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.2)$$

4.4. Experiments and Results

4.4.1. Experimental Setup

Training data

The number of news documents used in this experiment was 5100. Since hierarchical classification emphasizes the relationship among classes, rather than building single huge classifier, a classification is accomplished with the cooperation of classifiers built at each level of the tree. The training data is organized into 3 levels: from level-0(root level) to level-2. Each level represents classes or subclasses in a classification tree. Thus, there were 8 classes at level-0, 20 classes at level-1, and 69 classes at level-2 with at least 14 documents in them.

The classifiers at each level were trained using the associated documents of all subclasses of that class. Thus, the level-0 classifier was trained using documents of all subclasses of that class from level-1 through 2. In contrast, each level-1 classifier was trained with documents from the appropriate level-1 subclasses up level-2.

Testing data

The accuracy of the classifier was tested using the test data selected from each level-3 documents. These documents were excluded from the training process and were selected from different level-3 classes. Since the class from which the test documents were selected is known, the accuracy of the classifier is evaluated how often the classifier assign the test documents to the classes from which they originally came. Moreover, we used the accuracy of classification (see equation4.2) as an evaluation measure.

4.4.2. Effects of the Number of Classes and Documents on Flat Classification

We created a single classification system by training a flat classifier for all classes in the top 3 levels of the classification tree, ignoring structure. In other words, each of the 97 classes was trained using 70% of the documents from each class. We had broken the classification process into pieces of classes taken separately at a time to see the performance of the classifier while increasing number of classes and documents (features). Thus, classes in level-0, level-1, and level-2 were separately considered for the first, second, and third experiment respectively. Since each document is assigned in the leaf node of the classification tree; level-0 classes will have the same number of documents as that of level-1 and level-2 when used separately for the next corresponding experiment. Hence, I selected documents using 50% of the collection to experiment on level-1 classes, and 70% of collection for the second experiment and 90% of the document collection for the third experiment.

Moreover, the effect of top features on the classifier performance of a flat classification system and the reasons behind is addressed in this section of the experiment.

Experiments labeled as I, II and III shows the result and discussion of the experimentation when an increasing number of classes and documents were considered where as Experiment IV shows the effect of top features on the classifier performance of a flat classification system.

I. Classification with 8-Classes

In this level, 8- Classes and 50% of the total number of documents in the collection were considered. Training and testing data shares the 70% and 30% of these data. Hence, 1785 documents were used for training and 765 documents were used for testing. Thus, 80.34% accuracy was found as result of this experiment.

Category	Tourism & Culture	Economy	Education	Health	Law & Justice	Politics	Social	Sport	Tot.
Training	525	394	639	399	353	596	486	182	1785
Testing	224	168	273	170	151	255	207	78	765
Accuracy	80.34%								2550

Table4.6. Experiment on 8-(level-0) classes

II. Classification with 20-Classes

The number of classes which are considered in this experiment was 20; and then 70 % of the total documents in the collection (from which 70% of it for training and the remaining for testing) were used. Thus, 66.09% accuracy was found as the result of the experimentation. Table4.7 shows the result of the experimentation when more number of classes and documents were used.

Training/testing data sets	No. of documents	Total
Training	2499	3570
Testing	1071	
Accuracy	66.09%	

Table4.7. Experiment on 20(level-1) classes

III. Classifications with 69-Classes

To make the experiment sound, again more number of classes and documents were used in this experiment. Hence, 69 classes and 90% of the total document in the collection were used; and the training and testing data receives 70% to 30% respectively. An accuracy of 50.32% accuracy was found as the result of this experiment.

Training/testing data sets	No. of documents	Total
Training	3213	4590
Testing	1377	
Accuracy	50.32%	

Table4.8. Experiment on 69(level-2) classes

From the above three experiments, it was found that the accuracy decreases when the number of classes increased from 8 to 20 and then to 69; and the number of documents increased from 2550 to 3570 and then to 4590. Moreover, the average accuracy obtained from the above three experiments was 65.58%, which were decreased at each steps of the experiment. This shows that as the number of classes and documents increase, the performance of a flat classifier decreases. This is because, as the number of documents is increasing, the number of support vectors increases. Since the classification is done in a multidimensional plane where we can draw a number of hyperplanes, the increasing number of support vectors causes to narrow the margin between these hyperplanes. The smaller the marginal hyperplane then causes maximum classification error on unseen test instances apart from the difficulty to get the maximum marginal hyperplane (MMH).

IV. Effects of Number of Top Features on Flat Classification

All the 97 classes and total documents in the collection were used to see the performance of the flat classifier at increasing number of top features which were extracted from the test documents. Thus, the top features up to 20 words were considered where the features were selected based on their *tfidf* weights. The peak accuracy in this experiment was 68.84 % when the top 3 features[5] were used. This means that least number of features has high discriminating power among classes than more number of features, which are found across many classes. Figure4.2 shows effects of top features on flat classifier.

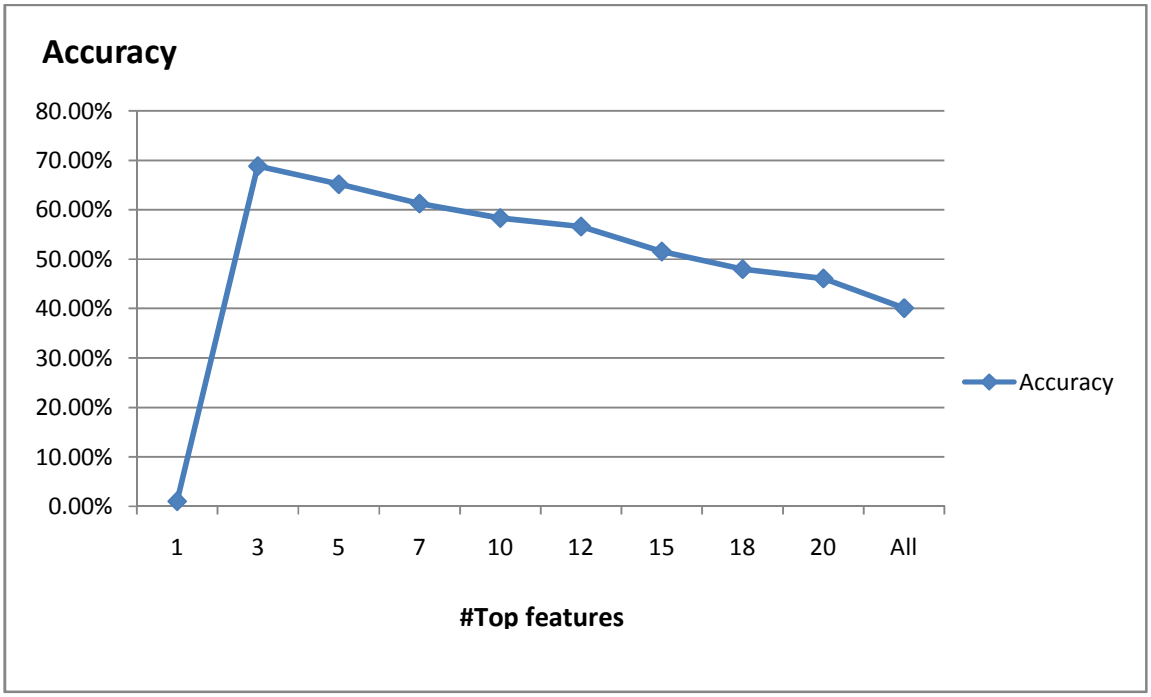

Figure4.2. Effects of the number of top features on the performance of flat classification

[5] Top 3 features are selected based on their term weight. Hence, the first three maximum weighted features are used.

4.4.3. Experiment Using Hierarchical Classification

For the hierarchical classifier, we constructed a set of classifiers, one at each level of the classification tree, using training method described in section 4.4.1. Thus, there was one classifier at level-0 (trained on the 8 level-1 classes), 8 classifiers for level-1 (one for each level-1 class), and 20 classifiers for level-2 (one for each level-2 class). Since each classifier has to deal with a more easily separable problem, and can use an independently optimized feature set, it leads to slight improvements in accuracy apart from the gain in training (learning) and testing speed.Table4.9 shows the performance of hierarchical classifier as it improves down the hierarchy for randomly selected classes (Education(code 2), Health (code 4) & Politics (code 6)).

Classifier	*Level-0 Classifier*
Training set Testing set	3570 1530
Accuracy	63.03%

Classifier	*Level-1 Classifier (2)*	*Level-1 Classifier(4)*	*Level-1 Classifier(6)*
Training set	394	399	595
Testing set	168	170	256
Accuracy	78.76%	81.15%	79.76%

Level-2 Classifier	2.1	2.2	2.3	4.1	4.2	4.3	6.1	6.2
Training set	163	161	70	176	33	190	475	120
Testing set	70	68	30	75	14	81	204	52
Accuracy	87.93 %	90.37 %	88.23 %	85.71 %	89.37 %	86.01%	82.62%	85.56%

Table4.9: The Increasing performance of hierarchical classifiers

Table4.9 shows the improved performance of the hierarchical classifiers at each levels of the classification tree. This is because each classifier deals with the documents associated to only that class or subclasses of that class and it concentrates on a smaller set of documents, those relevant to the task at hand. As it is shown above, exploiting the relationship among classes and utilizing the hierarchical topic structure results in a considerable increase in the classifier accuracy.

The testing data in the table4.9 shows those instances which participate in testing the level-0 classifier, extracted from the corresponding class in level1 through 2. This is because the LibSVM only evaluates whether the classifier assigns the test documents to the classes from which they originally came; from which the accuracy is calculated using the equation described in equation 4.1. Thus, in this study level-1 and level-2 classifiers are tested with documents selected from that classes and subclasses of that class; others left out as it always degrade the accuracy value of the classifier.

Effects of Number of Top Features in Hierarchical Classifiers

The documents were initially classified at level-0 using a varying number of features per document where the features were selected based on their *tfidf* weights. The first run used only the highest weighted feature for classifying the documents and number of features was increased in each subsequent run until a maximum of 20 features. The level-0 classifier had a peak accuracy of 81.50% when the top 5 features were used. Figure4.3 shows the result of the experiment.

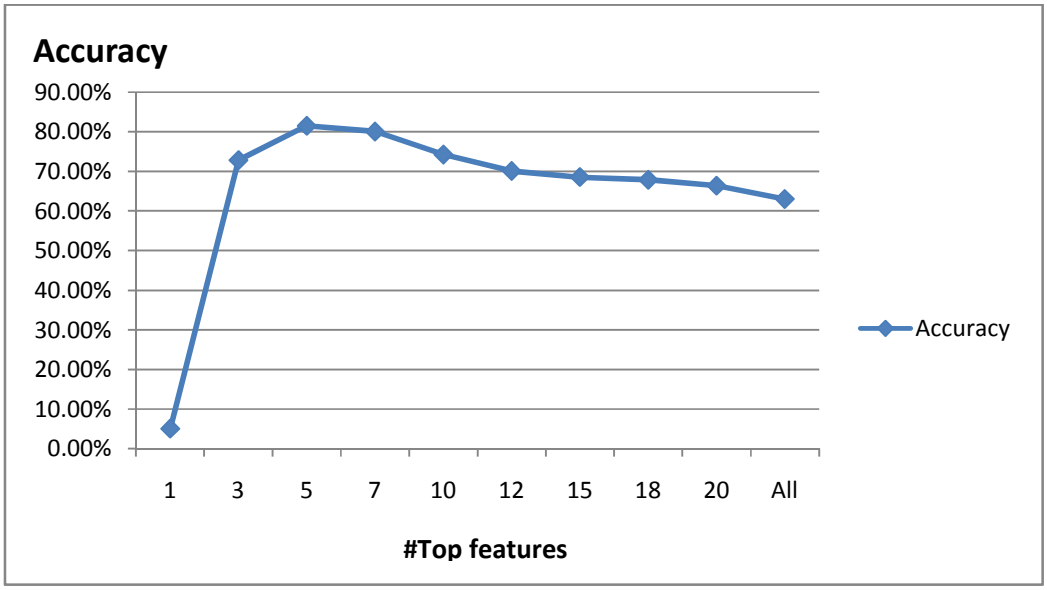

Figure4.3: The effect of the number of top features selected from test documents on level-0 classification accuracy

The test documents were then classified at level-1 while again varying the number of top features from 1 to 20. At level-1, the classification process is same as above, but it is constrained to consider only the subclasses of the best matching class at level-0. As shown in figure4.4 below, the level-1 classifier had a peak accuracy of 85.07% when the top 10 features were used.

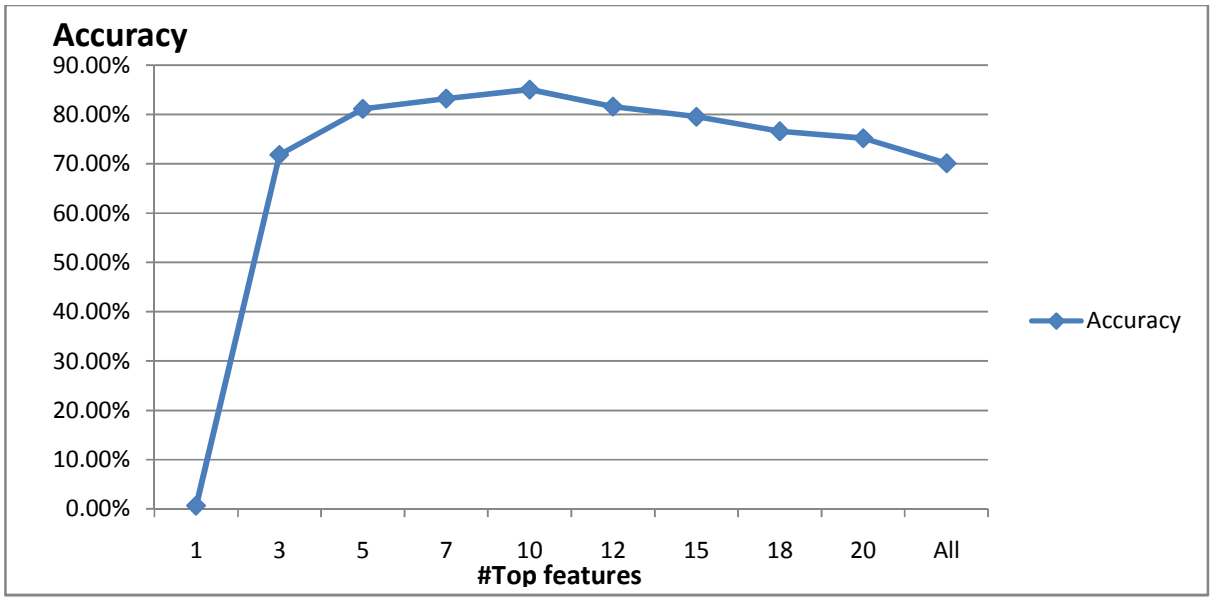

Figure4.4: The effect of the number of top features selected from test documents on level-1 classification accuracy

Finally, the test documents were classified at level-2 with the classification process now constrained to consider only the subclasses of the best matching class at level-1. Since all the test documents originally came from level-2 classes, the accuracy of the classifier overall is best judged by the accuracy at level-2. The level-2 classifier had an exact match precision of 89.06% when the top 15 features were used. This means that, from a set of 97 classes, the hierarchical classifier correctly classified 89.06% of documents to their original class.

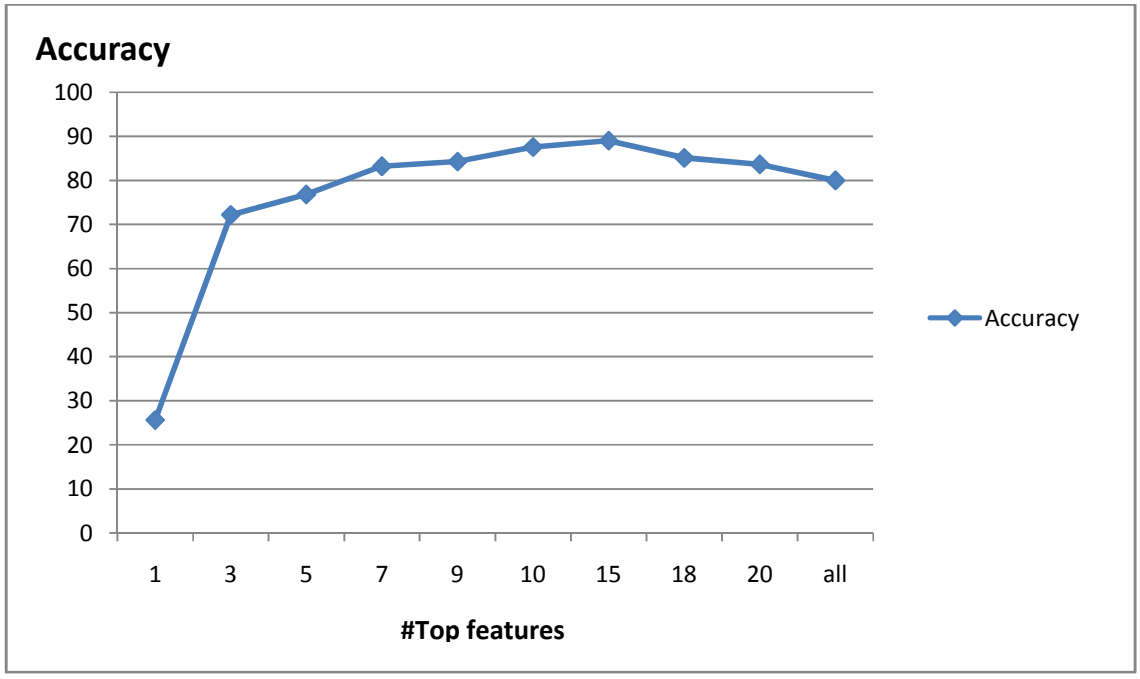

Figure4.5: The effect of the number of top features selected from test documents on level-2 classification accuracy

It is interesting to note that, as we move down the hierarchy, the classifiers perform better with more features extracted from test documents. This is because they need more information in order to make finer-grained distinctions between the classes.

4.4.4. Comparison between Flat Classifier and Hierarchical Classifiers

Analyzing the result of the above experiments, as the number of class and documents to be considered increases, the performance of flat classifier decreases but the performance of hierarchical classifiers increases as we move down the hierarchy. This shows that flat classification depends on the number of classes and documents to be considered compared to hierarchical classifiers.

To compare the relative performance of the flat classifier and the hierarchical classifiers, the same set of test documents was used. As shown in figure4.2 and figure4.5, the flat classifier produced an exact match accuracy of only 68.84 % when the top 3 features were used, where as with the hierarchical classifiers 89.09 % of the documents classified had exact match accuracy when the top 15 features were taken. This implies that least number of features are needed to discriminate among a large number of classes where there is no relationship among classes; whereas more number of features are needed to discriminate among classes where there are more closer similarity between classes or documents of a class (in hierarchical classification). Thus, we can conclude that the use of hierarchy for text classification results in a significant improvement of 29.42 % in exact match accuracy over the flat classifier.

Chapter Five

Conclusion and Recommendations

5.0. Conclusion

Along with the advancement of Information Technology, we are flooded by huge amount of information. This problem has caused obstacles to human beings to get useful information out of these huge collections. Hence, this necessitates a proper way of data organization and management in such a manner that it can be easily accessible to those who seek it. One way is the use of manual document classification. Manual classification requires individuals to assign each document to one or more categories. However, as the amount of data and information increases, this approach became tedious and consumes times to organize documents using human hand.

An alternative way is to organize data and information using automated systems, which are often referred to as automatic document classification. It uses automatic solutions to classify an electronic document into one or more categories based on the characteristics of its contents.

Several researches have been done on automatic document classification with the help of different machine learning approaches; and good results were found. However, most of them focus on flat classification system, i.e. each topic (category) is considered independent of others where there is no any relationship among them.

Even though, flat classification has become a well-established research area for the last 30 to 40 decades and many good classifiers have been developed, the approach hadn’t yet a feasible solution where most real world application need structures that define relationships among them are necessary. As the technology such as internet grows, the number of possible categories

increases and the borderlines between document classes are blurred. As we use a large corpus we may have hundreds of classes and thousands of features. The computational cost of training a classifier for a problem of this size is prohibitive

To solve these problems, approach that utilizes the hierarchical topic structure to decompose the classification task into a set of simpler problems is proposed. It is often referred to as hierarchical classification approach. Hierarchical text classification uses a divide-and-conquer approach (also known-as top-down approach) to deal the large classification problem into a set of simpler sub problems, one at each node in the classification tree.

In such a hierarchical structure document types become more specific as we go down in the hierarchy. Thus, hierarchical classification of documents is very much important to access a specific document or group of documents from the hierarchical organized document collections as compared to flat classification.

This paper also introduces support vector machines (SVM) for hierarchical text categorization. It provides both theoretical and empirical evidence that SVMs are very well suited for text categorization in general and hierarchal classification in particular. The theoretical analysis concludes that SVMs acknowledge the particular properties of text: (a) high dimensional feature spaces, (b) few irrelevant features (dense concept vector), and (c) sparse instance vectors.

The experimental results show that SVMs consistently achieve good performance on hierarchical text categorization tasks, outperforming existing methods substantially and significantly. With their ability to generalize well in high dimensional feature spaces, SVMs eliminate the need for feature selection, making the application of text categorization considerably easier. Another advantage of SVMs over the conventional methods is their robustness. Furthermore, SVMs do

not require any parameter tuning, since they can find good parameter settings automatically. All this makes SVMs a very promising and easy-to-use method for learning text classifiers from examples.

LibSVM was used as an experimentation tool due to its ability for efficient multiclass classification, automatic model selection and contains different SVM formulations.

The data collected for this study a one-level data. However, it is preprocessed in a manner that hierarchical data (categorically leveled) data is obtained to fit with the purpose of the study. Document-similarity matrix and experts' judgment were used to generate a categorical level data. Therefore, three level hierarchical data is obtained with 8 level-0 classes, 20 level-1 classes and 69 level-2 classes.

The experiment was done following three approaches.

1. Assure whether the traditional classification method (flat classification system) is dependent on the number of classes and features
2. Constructing hierarchical classifiers at each levels of the classification tree and see whether the performance of the classifiers were improved as we move down the hierarchy
3. Evaluating the classification performance between existing traditional system (flat classifier) and the hierarchical classifiers with the same test data.

Accordingly, the following result was obtained based on the experiments done using the above three approaches.

- Based on the first approach, the 97 classes were divided into 8, 20 and 69 separate classes with an increasing number of documents in them. Thus, it was found that the accuracy was decreasing from 80.34% to 66.09% and then to 52.32% as the number of classes

increased from 8 to 20 and then to 69; and the number of documents increased from 2550 to 3570 and then to 4590 respectively. This shows that as the number of classes and documents increase, the performance of a flat classifier decreases. This is because, as the number of support vectors increases with the increasing number of documents. Since the classification is done in a multidimensional plane where we can draw a number of hyperplanes, the increasing number of support vectors causes to narrow the margin between these hyperplanes. The smaller the marginal hyperplane then causes maximum misclassification error on the later unseen test instances apart from a difficulty to get the maximum marginal hyperplane (MMH).

- According to the second approach, an experiment was done on randomly selected class levels as shown in table4.9. Thus, it shows an improved performance as we move down the classification tree. For example, the maximum accuracy achieved is in level-2(the last level in the category level), which is 90.37% in the economy class as designated using code 2.2 in table4.9.

  The improved performance of the hierarchical classifiers at each levels of the classification tree in the experiment is because each classifier deals with the documents associated to only that class or subclasses of that class, it concentrates on a smaller set of documents, those relevant to the task at hand. Hence, the maximum marginal hyperplane can be easily generated in one hand linear SVM classifier can be easily applied in the other hand. Moreover, we can deduce that a considerable increase in the classifier accuracy is as a result of exploiting the relationship among classes and utilizing the hierarchical topic structure in it.

- Again based on the third point, an experiment was done using the top number of features selected from the same test document using one flat classifier and classifiers at each

levels of the category tree. Accordingly, a flat classifier shows a maximum exact match accuracy when 3 top features were used whereas the hierarchical classifiers shows an improved exact match accuracy at increasing number of top features at each level of the category tree. As a result, the last level (level-2) classifier accuracy result was taken and compared with the accuracy obtained in flat classifier. In such a way, the flat classifier produced an exact match accuracy of only 68.84% when the top 3 word were used, where as with the hierarchical classifiers 89.09% of the documents classified had exact match accuracy when the top 15 features were taken. This means that the use of hierarchy for text classification results in a significant improvement of 29.42 % in exact match accuracy over the flat classifier.

From this experiment, we can understand that more words are needed to discriminate classes (topics) that are close to each other in hierarchy as they have more in common with each other than classes (topics) that are spatially far apart.

Apart from the increased classification performance, classification speed can be taken as an advantage in hierarchical classification approach using Support Vector Machine. The only limitation to SVM is the longer learning/training time. It might take more than a day. This learning time could increase with an increase number of training data.

Finally, the findings of this research could be much significant to content-based information retrieval in addition to the different applications explained in the paper.

5.2. Recommendations

The aim of this research is to explore the use of hierarchical structure for classifying a large, heterogeneous collection of Amharic news items. Accordingly, the result of this research showed that hierarchical classification is a potential solution in classifying a large, heterogeneous collection of Amharic news text efficiently and effectively. However, there are also additional tasks recommended to see the full capacity of using a hierarchical approach in text classification. Thus, two kinds of recommendations are forwarded.

i. Regarding the Domain Under Study (Amharic News Text)

Many preliminary works in Amharic language have to be done as prerequisites for any research on Amharic text processing. Otherwise, it would require a researcher do these tasks from scratch and rather than concentrate on a specific problem he/she interested to do. In this regard, at least the following systems should be done:

a. **Amharic spell checker**: as can be seen in many documents, Amharic documents suffer from spelling errors. In reality spelling errors will degrade the performance of text processing systems.
b. **Stop word list**: the stop lists used in this research are mainly news specific terms as explained in section4.3.2. As a result, they may not be helpful in other areas or domains. Therefore, an exhaustive stop word list for the language should be developed.
c. **Thesaurus**: as there are many variants of words in Amharic, a thesaurus would help in reducing the features identified by bringing variants of the same word into a one common word. This will increase the discrimination power of terms. Thus, Amharic thesaurus should be developed.

d. **Corpus Preparation:** in other languages, it is very common to prepare corpus for research purpose; unfortunately, there are no standard corpus for Amharic text classification, as to the researcher's knowledge. Researchers can devote much time on their work if standard corpus is prepared for Amharic classification experiments like 'Reuters-21578' for English.

ii. Regarding the Approach Used

- The approach used in this study is solely a supervised one. However, there are some concepts which did not match with any of the classes. Such kinds of cases in supervised approach are forced to be assigned one of the classes despite forming its own cluster (class). In such cases, a combination of supervised learning and clustering is important to solve these kinds of problems. So this method should be tested.

- In our hierarchical classification approach, an error made at the parent category is not recoverable further down the tree. Some mechanism to take care of this error cascading can be developed so that child classifiers are able to recover from the errors made at their parents.

References

1. A. Sun and E.-P. Lim. Hierarchical text classification and evaluation. In *Proc. of the 1st IEEE Int. Conf. on Data Minin*g, pages 521–528, California, USA, Nov 2001.
2. A. Sun, E. Lim, and W. Ng. Performance Measurement Framework for Hierarchical Text Classification. *Journal of the American Society for Information Science and Technology, 54(11), 200*3. Pages 1014-1028.
3. Ana Fuentes Martinez. Document classification for computer Science Related Articles. Journal of Information Science May 15, 2002.
4. Ashwin & Susan. Hierarchical text classification. Electrical and Computer Science Department, University of Kansas, Masters Thesis, 2001.
5. Atelach, A. Automatic Sentence Parsing for Amharic Text an Experiment Using Probabilistic Context Free Grammars. Addis Ababa University, Masters Thesis, 2002
6. B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory*, pages 144.152, 1992.
7. Bender, M.; Bowen, J.; Cooper, R. and Ferguson, C. the Ethiopian writing system. Oxford University Press: London. 1976
8. Bernal M. Cadmean letters winona Lake: Eisenbrauns, 1990
9. Bi, Y.; Murtagh, S. and Anderson, T. Text Passage Classification Using Supervised Learning. Oxford University Press, 1999.
10. Chien, L.-F, Huang, C.-C., & Chuang, S.-L. Creating hierarchical text classifiers through web corpora. WWW '04: Proc. of the 13th Int. Conf. on World Wide Web (pages 184-192). New York, NY, USA: ACM Press, 2004.
11. Coulmas & Guar . The writing system of the world: Basil Blackwell, 1989. Available at Http://.spellingsociety.org/journals/j19/ethiopic.php. . Accessed March 17, 2010

12. D. Levy, Users and interaction track: memex and hypertext: to grow in wisdom: Vannevar Bush, information overload, and the life of leisure. In: *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital libraries.* 1999 pages 281–286.

13. D'Alessio, S., Murray, K., Schiaf_no, R., & Kershenbaum, A. The effect of using hierarchical classifiers in text categorization. *Proceeding of RIAO-00, 6 International Conferences*, 2000.

14. Dumais, S., & Chen, H. Hierarchical classification of web document. Graz University of Technology, Austria, Masters Thesis, 2000.

15. Edda Leopold & Jorg Kindermann, Text Categorization with Support Vector Machines: How to Represent Texts in Input Space?, *German National Research Center for Information Technology, Institute for Autonomous intelligent Systems, Germany,* 2002

16. Eiring, H.L. The evolving information overload, information managmnet formal journal vol. 36 No. 2002 pages 20-24.

17. F. Sebastiani. Machine learning in Automated Text Categorization-in ACM Computing surveys 34(1), 2002, pages 1-47.

18. G. Salton and C. Buckley. Term-weighting approaches in Automatic text retrieval. *Information Processing and Management: an International Journa*l, 24(5): pages 513-523, 1988.

19. G. Zipf, Selective Studies and the principle of Relative Frequency in Language. Cambridge, Mass, 1932.

20. Giorgino, T. An Introduction to Text Classification. 2004. Available at www.cs.cornell.edu/people/tj/publications/joachims_98a.pdf.

21. H. Schutze, D. Hull, and J. Pederson. A comparison of document representations and classifiers for routing problem. In proceeding of SIGIR 1995.pages 229-237.

22. Han, Eui-Hong et al. Text Categorization using Weight K-Nearest Neighbor classification. Minnesota: University of Minnesota. Available at http://www.cs umn.edu/~han, 1999.

23. I. Dhillon, S. Mallela, and R. Kumar. Enhanced word clustering for hierarchical text classification. Ben Allison Department of Computer Science University of Sheffield UK, Masters Thesis, 2002.

24. Ian H. Witten & Eibe Frank. Data mining: practical machine learning tools and techniques, department of computer science, university of WAIKATO, second edition, 2005. ISBN 0-12-088407-0.

25. Infoseek. 1995. Internet directory and query service. http//www.infoseek.com/.

26. J. Han and M. Kamber. *Data Mining: Concepts and techniques* (2^{nd} ed.). Morgan Kaufmann Publishers, 2006.

27. J. Weston and C. Watkins. Multi-class support vector machines, 1999.

28. James Tin Yau Kwok. Automated Text Categorization Using Support Vector Machine, Department of Computer Science, Hong Kong Baptist University, Hong Kong, 1999.

29. K. Summers. Near-wordless document structure classification. In proceedings of the international conference on document analysis and recognition ICDAR, 1995, pages 462-465.

30. K. Wang, S. Zhou, and Y. He. Hierarchical classification of real life documents. In *Proceedings of the 1st SIAM International Conference on Data Mining*, Chicago 2001.

31. Klein, B. Text Classification Using Machine Learning. Journal of Theoretical and Applied Information Technology. 2004.

32. Koller & Sahami. Hierarchically classifying documents using very few words. The 14th national conference on machine learning. Computer Science department, Stanford University , 1997

33. Kwan Yi and Jamshid Beheshti, A hidden Markov model-based text classification of medical documents, *Journal of Information Science*. Oct 23, 2008;

34. M.A Hearts and c. Plaunt. Subtopic Structuring for full-length document access. In research Development in Information Retrieval, pages 59-68,1993

35. Michael Grantizer, Hierarchical document classification using methods from machine learning, Graz University of Technology, Masters Thesis , October 2003.

36. Neumann, Günter & Sven Schmeier. Combining Shallow Text Processing and Machine Learning in Real World Applications. 1999. Available at http://www.dfki.de/~neumann/publications/newps/ijcai99-ws.pdf.

37. Omniglot. Writing systems and languages of the world. 1998.

38. P.G. Chander, R. Shinghal, B.C. Desai and T. Radhakrishnan, An expert system to aid cataloging and searching electronic documents on digital libraries, *Expert Systems with Applications* 12(4), 1997. Pages 405–416.

39. Panu Erasto. Support Vector Machine-Background and practice, Academic dissertation for the degree of Licentiate of Philosophy, Rolf Nevanlinna Institute, Helsinki. 2001.

40. Rasmussen, E. Clustering Algorithms. *In Data Structures and Algorithm*. Prentice Hall PTR. 1992

41. Rennie, Jason D. M. Improving Multi-Class Text Classification with Naive Bayes. Massachusetts Institute of Technology, Masters Thesis, 2001.

42. Robert Blumberg and Shaku Atre, 2003. Automatic classification: moving to the mainstream. Available at www.soquelgroup.com/articles/dmreview0403 classificatio.pdf. Reprinted from DM Review April 2003.

43. Shankar Ranganatan. Text classification combining clustering and hierarchical approaches. Computer Science and Engineering, University of Madras, Chennai, India, Masters thesis, 2001

44. Solomon, T. and Menzel, W. Syllable-Based Speech Recognition for Amharic. Proceedings of the 5th Workshop on Important Unresolved Matters, 2007. Pages. 33–40.

45. Surafel Teklu. Automatic categorization of Amharic news text: A machine learning approach. Department of Information Science, Addis Ababa University, Masters Thesis, 2003.

46. T. Joachims, Text categorization with support vector machines: learning with many relevant features, *Proceedings of ECML 98, 10th European Conference on Machine Learning* (Chemnitz, Germany, 1998). Pages 137–142.

47. T. Mitchell. Machine Learning. McGraw-Hill International, 1997.

48. Tessema Mindaye & Dr. Solomon Atnafu. Search Engine for Amharic Web documents. ISBN: 978-3-639-19632-0. VDM Verlag co. Germany, 2007.

49. Tewodros, H. Amharic Text Retrieval: an Experiment Using Latent Semantic Indexing (LSI) with Singular value Decomposition (SVD). Addis Ababa University Department of Information Science. Masters Thesis. 2003.

50. Unicode Consortium v.3.0. Unicode, Inc.

51. W. B. Frakes and R. Baeza-Yates. Information Retrieval: Data Structures and Algorithms. Prentice Hall: N. J., 2002.

52. W. Wibovo and H. E. Williams. Simple and accurate feature selection for hierarchical categorization. In *Proc. of the 2002 ACM symposium on Document engineerin*g, pages 111–118, McLean, Virginia, USA, 2002.

53. W. Zaghloul et.al. Text classification: neural networks vs support vector machines, journal of Industrial Management & Data Systems , 2009. Vol. 109, No. 5 pages: 708-717,ISSN: 0263-5577, Emerald Group Publishing Limited

54. Wei-Feng Cao, Lei Li, Xiao-Lilv. Kernel function characteristic analysis based on support vector machine in face recognition. School of Telecommunication and Information, Nanjing University of Posts and Telecommunications, China. 2007

55. Wen Zhang, Xijin Tang , and Taketoshi Yoshida, Text Classification with Support Vector Machine and Back Propagation Neural Network, ICCS 2007, Part IV, LNCS 4490, 2007. Pages. 150–157,

56. Worku K. Automatic Amharic News Text Classification: A Neural network approach. Department of Information Science, Addis Ababa University, Master's Thesis, 2009.

57. Y. Yang and X. Liu, A re-examination of text categorization methods, *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval* (Berkeley, CA, 1999) pages 42–49.

58. Y.M. Chung and Y.-H. Noh. Developing a specialized directory system by automatically classifying web documents, *Journal of Information Science* 29(2) (2003) pages 117–126.

59. Yahoo! 1995. On-line guide for the internet. http://www.yahoo.com/.

60. Yohannes A. Automatic Amharic news text classification using Support Vector Machine approach. Department of Information Science, Addis Ababa University, Master's Thesis, 2007.

61. Zelalem Sintayehu. Automatic Classification of Amharic News Items: The Case of Ethiopian News Agency. School of Information Studies for Africa, Addis Ababa University, Addis Ababa, 2001.

Appendix

Appendix1: Amharic characters (‘Fidel’)-ፊደል- adapted: from Zelalem, 2001

Order							Labialized				
1st	2nd	3rd	4th	5th	6th	7th					
ሀ ha	ሁ hu	ሂ hi	ሃ ha	ሄ he	ህ h	ሆ ho					
ለ lä	ሉ lu	ሊ li	ላ la	ሌ le	ል l	ሎ lo			ሏ l^{w}a		
ሐ ha	ሑ hu	ሒ hi	ሓ ha	ሔ he	ሕ h	ሖ ho					
መ mä	ሙ mu	ሚ mi	ማ ma	ሜ me	ም m	ሞ mo			ሟ m^{w}a		
ሠ sä	ሡ su	ሢ si	ሣ sa	ሤ se	ሥ s	ሦ so					
ረ rä	ሩ ru	ሪ ri	ራ ra	ሬ re	ር r	ሮ ro			ሯ r^{w}a		
ሰ sä	ሱ su	ሲ si	ሳ sa	ሴ se	ስ s	ሶ so			ሷ s^{w}a		
ሸ šä	ሹ šu	ሺ ši	ሻ ša	ሼ še	ሽ š	ሾ šo			ሿ š^{w}a		
ቀ qä	ቁ qu	ቂ qi	ቃ qa	ቄ qe	ቅ q	ቆ qo	ቈ q^{w}ä	ቍ q^{w}i	ቋ q^{w}a	ቌ q^{w}e	ቍ q^{w}∂
በ bä	ቡ bu	ቢ bi	ባ ba	ቤ be	ብ b	ቦ bo			ቧ b^{w}a		
ተ tä	ቱ tu	ቲ ti	ታ ta	ቴ te	ት t	ቶ to			ቷ t^{w}a		
ቸ čä	ቹču	ቺ či	ቻ ča	ቼče	ች č	ቾčo			ቿ č^{w}a		
ኀ hä	ኁ hu	ኂ hi	ኃ ha	ኄ he	ኅ h	ኆ ho	ኈ h^{w}ä	ኊ h^{w}i	ኋ h^{w}a	ኌ h^{w}e	ኍ h^{w}∂
ነ nä	ኑ nu	ኒ ni	ና na	ኔ ne	ን n	ኖ no			ኗ n^{w}a		
ኘ ñä	ኙ ñu	ኚ ñi	ኛ ña	ኜ ñe	ኝ ñ	ኞ ño			ኟ ñ^{w}a		
አ a	ኡ u	ኢ i	ኣ a	ኤ e	እ ∂	ኦ o					
ወ wä	ዉ wu	ዊ wi	ዋ wa	ዌ we	ው w	ዎ wo					
ዐ a	ዑ u	ዒ i	ዓ a	ዔ e	ዕ ∂	ዖ o					
ከ kä	ኩ ku	ኪ ki	ካ ka	ኬ ke	ክ k	ኮ ko	ኰ k^{w}ä	ኲ k^{w}i	ኳ k^{w}a	ኴ k^{w}e	ኵ k^{w}∂
ኸ hä	ኹ hu	ኺ hi	ኻ ha	ኼ he	ኽ h	ኾ ho					
ዘ zä	ዙ zu	ዚ zi	ዛ za	ዜ ze	ዝ z	ዞ zo			ዟ z^{w}a		
ዠ žä	ዡ žu	ዢ ži	ዣ ža	ዤ že	ዥ ž	ዦ žo					
የ yä	ዩ yu	ዪ yi	ያ ya	ዬ ye	ይ y	ዮ yo					
ገ gä	ጉ gu	ጊ gi	ጋ ga	ጌ ge	ግ g	ጎ go	ጐ g^{w}ä	ጒ g^{w}i	ጓ g^{w}a	ጔ g^{w}e	ጕ g^{w} ∂
ደ dä	ዱ du	ዲ di	ዳ da	ዴ de	ድ d	ዶ do			ዷ d^{w}a		
ጀ ǧä	ጁ ǧu	ጂ ǧi	ጃ ǧa	ጄ ǧe	ጅ ǧ	ጆ ǧo					
ጠ ṭä	ጡ ṭu	ጢ ṭi	ጣ ṭa	ጤ ṭe	ጥ ṭ	ጦ ṭo			ጧ ṭ^{w}a		
ጨ ćä	ጩ ću	ጪ ći	ጫ ća	ጬ će	ጭ ć	ጮ ćo			ጯ ć^{w}a		
ጸ ṣä	ጹ ṣu	ጺ ṣi	ጻ ṣa	ጼ ṣe	ጽ ṣ	ጾ ṣo			ጿ ṣ^{w}a		
ፀ ṣä	ፁ ṣu	ፂ ṣi	ፃ ṣa	ፄ ṣe	ፅ ṣ	ፆ ṣo					
ፐ pä	ፑ pu	ፒ pi	ፓ pa	ፔ pe	ፕ p	ፖ po					
ፈ fä	ፉ fu	ፊ fi	ፋ fa	ፌ fe	ፍ f	ፎ fo			ፏ f^{w}a		
𝟉 pä	𝟊 pu	𝟋 pi	𝟌 pa	𝟍 pe	𝟎 p	𝟏 po					
𝟐 vä	𝟑 vu	𝟒 vi	𝟓 va	𝟔 ve	𝟕 v	𝟖 vo					

Appendix2: Amharic punctuation marks (Adapted from: Atelach, 2002)

No.	Punctuation mark	Symbol	Purpose
1	The four dots or double colon	::	Mark end of a sentence
2	Colon	:	Separate words in a sentence: not common
3	White space		Separate words in a sentence: current practice
4	Question mark	?	Placed at the end of questions
5	Exclamation mark	!	Used at the end of sentences that show exclamation
6	Comma	፣	Used like comma
7	Semi-colon	፤	Used like semi-column
8	Three dots	…	For deliberate omission of words, phrases, or sentences
9	Quotation marks	‹‹ ››	Used at the beginning and at the end of quoted word, phrase, etc.
10	Parenthesis	( )	To enclose elaboration
11	Stroke	/	Separate date, month, etc.
12	Mocking mark	፧	Placed at the end of mocking sentence

Appendix3: Amharic Numbers (adapted from: Zelalem, 2001)

1	፩	6	፮	20	፳	70	፸
2	፪	7	፯	30	፴	80	፹
3	፫	8	፰	40	፵	90	፺
4	፬	9	፱	50	፶	100	፻
5	፭	10	፲	60	፷	1000	፼

Appendix4: News items (major and sub categories) collected from ENA

No.	Major class	Sub no	Sub class
1	**ማህበራዊ** (Social)	1	**የሴቶች ጉዳይ** (Women Affairs)
		2	**ስራ አጥነት** (Unemployment)
		3	**ሰብአዊ እርዳታ** (Social aid)
		4	**ስርዓተ ፆታ** (Sex)
		5	**ጋብቻና ፍች** (Marriage & Divorce)
		6	**እድሮች** (Idir)
		7	**አሰሪና ሰራተኛ** (Employer and Worker)
		8	**አረጋዊን** (old persons)
		9	**ሕፃናትና ወጣቶች ጉዳይ** (kids & youths affairs)
		10	**የሙያና ህዝባዊ ማህበራት** (professional & public corporations)
		11	**አካል ጉዳተኞች** (physical disabled persons)
2	**ህግና ፍትህ** (Justice and law)	12	**ህገ መንግስታዊ ጉዳዮች** (Constitutional affairs)
		13	**ሙስና** (corruption)
		14	**የፍትህ ብሔር ብሔርሰቦች** (nations and nationalities & justice)
		15	**የፍትሕ አካላት** (Justical and legal bodies)
		16	**የወንጀል ጉዳዮች** (crime affairs)
		17	**ዘር ማጥፋት** (genocide)
3	**ትምህርት** (Education)	18	**ሁለተኛ ደረጃ ት/ቤት** (secondary school)
		19	**ከፍተኛ ደረጃ ት/ቤት** (higher institutions)
		20	**መደበኛ ያልሆነ ት/ቤት** (informal school)
		21	**መዋዕለ ህፃናት** (kindergartens)
		22	**ሴቶችና ት/ቤት** (women's education)
		23	**ተከታታይና የርቀት ት/ት** (distance & continuing education)
		24	**አንደኛ ደረጃ ት/ቤት** (primary level education)
		25	**የመምህርና የተማሪዎች ጉባዬ** (teachers & students forum)
		26	**የነፃ ት/ት ዕድል** (free scholarship)
		27	**የቴክኒክና ሙያ ት/ት** (technical & vocational education)
		28	**የት/ት ሽፋን** (education coverage)
		29	**የት/ት መገናኛ አዴዎች** (educational communication systems)
		30	**የት/ት መሳሪያዎች** (educational materials)
		31	**የት/ት ተቋማት ግንባታ** (educational institution development)
	ባህልና ቱሪዝም	32	**ሀይማኖታዊ ጉባየዎች** (religious conferences)
		33	**ሀይማኖታዊና ብሄራዊ ባዕል** (religious & national holidays)
		34	**ጎጅና ልማዳዊ ድርጊቶች** (Taboos)

4	(Culture and Tourism)	35	**ጎብኚዎች** (visitors)
		36	**ታሪክ** (history)
		37	**ኪነ-ጥበብ** (art)
		38	**ቅርሶች** (heritages)
		39	**የብሔር ብሔረሰቦችና ሕዝቦች** (NNP)
		40	**የቱሪዝም ልማት** (tourism development)
5	**ስፖርት** (Sport)	41	**ባህላዊ ስፖርት** (traditional sport)
		42	**ቦክስ** (boxing)
		43	**ሌሎች ዘመናዊ የስፖርት አይነቶች** (modern sports)
		44	**የፌደሬሽን አካላት** (federation bodies)
		45	**ዕግር ካስ** (football/soccer)
		46	**አትሌትክስ** (athletics)
6	**ሳይንስና ቴክኖሎጂ** (Science & Technology)	47	**መገናኛ ብዙሐን** (mass media)
		48	**ምርምርና ጥናት** (research and dissertations)
		49	**ኢንፎርሜሽን ቴክኖሎጂ** (ICT)
		50	**የፈጠራ ስራዎች** (creative works)
7	**አደጋ** (Accident)	51	**ሰዉ ሰራሽ አደጋ** (Manmade accidents)
		52	**የተፈጠሮ አደጋ** (natural accidents)
		53	**አደጋ መከላከል** (accident protection)
8	**ኢኮኖሚ** (Economy)	54	**ማዕድንና ኢነርጅ** (Mining and energy)
		55	**ማይክሮ ኢንተርፕራይዝ** (micro-enterprise)
		56	**ባንክና ኢንሹራንስ** (Banking and insurance)
		57	**ንግድ** (Trade and commercial)
		58	**አጠቃላይ የኢኮኖሚ ዕድገት** (GDP)
		59	**ዕርዳታና የልማት ትብብር** (Development and Aid cooperation)
		60	**ኢንቨስትመንት** (investment)
		61	**ገብርናና ገጠር ልማት** (Agriculture and rural development)
		62	**መሰረታዊ ልማት** (basic infrastructures)
		63	**የዉሃ ሃብት ልማት** (Water resources)
		64	**የኢንዳስትሪ ልማት** (Industry devp't)
9	**ግንኙነት፣ መከላከያና ደህንነት** (relation, Defence	65	**አለም አቀፍና አህጉራዊ ክንዋኔ** (international and continental activities)
		66	**ሽብርተኝነት** (terrorism)
		67	**ዲፕሎማሲያዊ ግንኙነት** (diplomatic relations)
		68	**ወታደራዊ ስልጠናና ማዕረግ** (Militerial Trainig and status)
		69	**ወታደራዊ ተዕልኮ** (militarilial missions)
		70	**የሀገር ደህንነትና ለአላዊነት** (National security)

	and security)	71	የዉጭ ግንኙነቶችና ዉይይቶች (Foreign relations)
		72	ዜግነትና ስደተኞች (citizenship and e/immigration)
10	ፖለቲካ (politics)	73	ዲሞክራሲና መልካም አስተዳደር (democracy and good governance)
		74	ብሔራዊ ፖለቲካ (national politics)
		75	ምርጫ (election)
		76	ሰላምና መረጋጋት (peace & stabilization)
		77	ሰብአዊና ዲሞክራሲያዊ መብቶች (human & d/rights)
		78	ዉይይቶች፣ ዉሳኔዎችና አዋጆች (discussions, decisions and proclamations)
		79	አለም አቀፍ ፖለቲካ (international politics)
		80	የፖለቲካ ሹመት (political delegation)
		81	የፖለቲካ ፓርቲዎች (political parties)
11	የአካባቢ ጠበቃና የአየር ሁኔታ (Weather & Env'tal preservation)	82	በርሃማነትና ሙቀት መጨመር (Desert)
		83	ደን ልማት (forest development)
		84	የዱር ዕንስሳት ጥበቃ (wild animal protection)
		85	የአካባቢ ብክለት (environmental pollution)
		86	የአየር ትንበያ (weather forecasting)
12	ጤና (Health)	87	ባህላዊ ህክምና (traditional)
		88	በሽታና ህክምና (Disease treatment)
		89	በሽታን መከላከል (Diseas protection)
		90	ሌሎች በሽታዎች (other diseases)
		91	መድሃኒቶችና አደገኛ ዕፆች (Drugs and Pharmatituicals)
		92	ወባ፣ቲቪና ኤች አይ ቪ(Malaria, TB & HIV)
		93	የጤና ባለሞያዎች (health professionals)
		94	የጤና ተቃማት ግንባታ (health center development)
		95	የጤና አገልገሎቶች (health services)
		96	የሕፃናትና የዕናቶች ጤና (children's and maternity health)
		97	የሕክምና መሳሪያዎች (Medical materials)

Appendix5: The hierarchical leveled data used for the study associated with their unique codes

Level-1	Code	Level-2	Code	Level-3	Code	No. of dcts
Tourism and Culture	1	Religion and Holiday	1.1	Religious & national holidays	1.1.1	109
				Religious conferences and forums	1.1.2	105
		Tourism Attractions and Development	1.2	Arts	1.2.1	79
				Heritages	1.2.2	110
				History	1.2.3	79
				Nations and Nationalities of Peoples	1.2.4	84
				Tourism Development	1.2.5	91
				Tourists	1.2.6	92
Economy	2	Agricultural and rural development	2.1	Farming	2.1.1	157
				Forestry	2.1.2	35
				Pasturalism	2.1.3	41
		Basic Developments and co-operations	2.2	Basic Infrastructures	2.2.1	92
				Development Aid and Co-operations	2.2.2	51
				Mines and /energy	2.2.3	28
				Trade and industry	2.2.4	34
				Water resources development	2.2.5	24
		Investment and finance	2.3	Banking and insurance	2.3.1	32
				Investment	2.3.2	43
				Micro-enterprises	2.3.3	25
Education	3	Education Expansion and Development	3.1	Education coverage	3.1.1	87
				Education communication systems	3.1.2	26
				Educational institution development	3.1.3	159
				Educational materials	3.1.4	31
				Free educational scholarship	3.1.5	23

		Formal Education	*3.2*	Distance and continuing education	3.2.1	29
				Higher institutions	3.2.2	19
				Kindergarten	3.2.3	272
				Primary school	3.2.4	35
				Secondary school	3.2.5	67
				Technical and vocational education	3.2.6	37
				Women's education	3.2.7	71
		Informal education	3.3	Informal school	3.3.1	19
Health	4	Disease Prevention & Treatment	4.1	Children's' & Maternity Health	4.1.1	34
				Disease Prevention and Treatment	4.1.2	135
				Malaria, TB &HIV-AIDS	4.1.3	64
				Other Diseases	4.1.4	18
		Drugs & Pharmatituicals	4.2	Drugs & Heroines	4.2.1	30
				Traditional Treatment	4.2.2	17
		Health Development	4.3	Health Center Development	4.3.1	150
				Health Professionals	4.3.2	17
				Health Services	4.3.3	90
				Medical Devices & Materials	4.3.4	14
Law and justice	5	Constitutional Affairs	5.1	Constitution	5.1.1	62
				Justical and legal bodies	5.1.2	66
				Nations and Nationalities of justice	5.1.3	81
		Crime affairs	5.2	Corruption	5.2.1	131
				Genocide	5.2.2	39
				Other crime affairs	5.2.3	112
Politics		National Politics		Conferences ,discussions, decisions and proclamations	6.1.1	297
				Democracy , good governance and Development	6.1.2	21
				Election	6.1.3	48

		International Politics		Poetical party	6.2.1	154
				Political Delegation	6.2.2	159
			6.2	Foreign Relation	6.2.3	102
				Human and democratic rights	6.2.4	40
				Peace, security and stabilization	6.2.5	30
Social	7	Civil and gender affairs	7.1	Gender	7.1.1	14
				Impaired and disabled persons	7.1.2	34
				Kids and youths affairs	7.1.3	105
				Unemployment and entrepreneurship	7.1.4	79
				Women's Affairs	7.1.5	103
		Civil laws	7.2	Employee and Worker	7.2.1	43
				Marriage and devorce affairs	7.2.2	14
		Social Aid and Co-operations	7.3	'Idir'	7.3.1	52
				Older persons	7.3.2	14
				Professional and public corporations	7.3.3	135
				Social Aid	7.3.4	104
Sport	8	Modern Sport	8.1	Athletics	8.1.1	194
				Boxing	8.1.2	18
				Football	8.1.3	33
				Others	8.1.4	15
		Traditional sport	8.2	Horse Riding	8.2.1	15
				Swimming	8.2.2	17
				Others	8.2.3	14

Appendix6: Lists of affixes removed from the token

Prefixes	Suffixes	
ለ	ም	አቸችን
ስለ	ምና	አቸችንም
በ	ና	ዉ
በየ	ን	ዎቸቸው
እንደ	ንም	ዎቸቸዉን
እንደየ	ንና	ዎቸቸዉንም
እየ	እና	ዎች
ከ	ኡ	ዎቸችን
ወደ	አች	ዎችን
ወደየ	አችም	
የ	አችን	

Appendix 7: lists of special words which should co-occur with another word to have meaning in a concept

- አውደ
- ቤተ
- ስነ
- ኪነ
- ነፍሰ
- ኃይለ
- ወልደ
- ምድረ
- አብያተ
- ጸረ
- አቅመ
- መልከ
- ልበ
- ቀዱ
- ገብረ
- ያመ

DECLARATION

THIS THESIS IS MY ORGINAL WORK AND HAS NOT BEEN SUBMITTED FOR DEGREE IN ANY OTHER UNIVERSITY, AND THAT ALL SOURCES OF MATERIAL USED FOR THE THESIS HAVE BEEN DULY ACKNOWLEDGED

_____

ALEMU KUMILACHEW TEGEGNIE

THE THESIS HAS BEEN SUBMITTED FOR EXAMINATION WITH MY APPROVAL AS UNIVERSITY ADVISOR

_____

WONDOWOSSON MULUGETA

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

DEPARMENT OF INFORMATION SCIENCE

HIERARCHICAL AMHARIC NEWS TEXT CLASSIFICATION

BY
ALEMU KUMILACHEW TEGEGNIE

APPROVED BY

EXAMINING BOARD

WONDWOSSON MULUGETA (MSc), ADVISOR_____

ERIMIAS ABEBE (MSc), EXAMINER_____