AIN SHAMS UNIVERSITY
Faculty of Computer
& Information Sciences
Computer Science Department

AN INTELLIGENT SYSTEM FOR AUTOMATED ARABIC TEXT CATEGORIZATION

A Thesis Submitted to Computer Science Department,
Faculty of Computer & Information Sciences, Ain Shams University
In partial fulfillment of the requirements for Master of Science Degree

By

Mena Badieh Habib
B.Sc. in Computer Science, 2002.
Demonstrator, Computer Science Department,
Faculty of Computer & Information Sciences,
Ain Shams University, Cairo, Egypt.

Under Supervision of

Prof. Dr. Mostafa Mahmoud Syiam
Professor of Computer Science,
Computer Science Department,
Faculty of Computer & Information Sciences,
Ain shams University, Cairo, Egypt.

Dr. Zaki Taha Fayed
Associate Professor of Computer Science,
Computer Science Department,
Faculty of Computer & Information Sciences,
Ain shams University, Cairo, Egypt.

Dr. Tarek Fouad Gharib
Associate Professor of Information Systems,
Information Systems Department,
Faculty of Computer & Information Sciences,
Ain shams University, Cairo, Egypt.

2008

Acknowledgements

First and foremost, I could never forget the late Prof Dr. Mosatafa Syiam who walked with me on the first steps with this work. I dedicate this work to his soul.

I would like to express my sincere gratitude to my chief supervisor Dr. Tarek Gharib from whom I have learned a lot, due to his supervision, guidance, support and advising till this work come to light.

I would like to thank Dr. Zaki Taha, for his valuable scientific and technical notes.

Also I would like to express my gratitude to Prof Dr. Abdel-Badeeh Salem the head of computer Science department who gave me the basic idea of this thesis and helped me with his great experience.

My great thanks also go Prof Dr. Essam Khalifa and Prof Dr. Said Ghoniemy for their encouragement.

Finally, my deepest thanks go to my parents for their unconditional love, and to my friends for their support.

This thesis would have been much different (or would not exist) without these people.

Mena

Abstract

New technological developments have resulted in a dramatic increase in the availability of on-line text-newspaper articles, incoming (electronic) mail, technical reports, etc. This led to the need for methods that help users organize such information. Text Categorization may be the solution for the increased need for advanced techniques. Text Categorization is the classification of units of natural language texts with respect to a set of predefined categories. Categorization of documents is challenging, as the number of discriminating words can be very large. Machine learning approaches are applied to build an automatic text classifier by learning from a set of previously classified documents.

Few researches have tackled the area of Arabic text categorization till the time we start working on this research. Arabic language is a Semitic language that has a complex and much morphology than English. It needs a set of preprocessing routines to be suitable for manipulation. Stop words like prepositions and particles are considered insignificant words and must be removed; Words must be stemmed after stop words removal. Stemming is the process of removing the affixes from the word and extracting the word root. After applying preprocessing routines, document is represented as a weighted vector. Representation process consists of two phases:

a) Term selection which can be seen as a form of dimensionality reduction by selecting a subset of terms from the full original set of terms according to some criteria,

b) Term weighting in which, for every term selected in phase (a) and for every document, a weight is computed which represents how much this term contributes to the discriminative semantics of the document.

Finally, the classifier is constructed by learning the characteristics of every category from a training set of documents, and tested by applying it to the test set and checking the degree of correspondence between the decisions of the classifier and those encoded in the corpus.

This thesis presents an intelligent Arabic text categorization system. Experimental results performed on a text collection of 1132 document collected from the local newspapers show that using light stemming along with trigram stemmer is the most appropriate stemming approach for Arabic language. The main problem with the traditional methods of feature selection is founding a large set of sparse documents (most of the documents does not contain any term in the list of the selected terms). To solve this problem we removed words that rarely appear in the documents before using information gain, this gives better results. Also we combined global and local feature selection to reduce the number of empty documents without affecting the performance. Normalized term frequency inverse document frequency (normalized-tfidf) was the most suitable weighting criteria for representing the documents as a vector of the set of selected terms (words). Finally after testing four famous classifiers, it has been shown that Rocchio classifier performs better when the number of terms is small while Support Vector Machines (SVM) outperforms the other classifiers when the number of is large enough. Classification accuracy exceeds 90% when using over than 4500 feature to represent documents.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

IR	Information Retrieval
TC	Text Categorization
KE	Knowledge Engineering
ML	Machine Learning
K-NN	*K* Nearest Neighbor
SVM	Support Vector Machines
d_j	Document *j*
D	Documents collection
C	Set of predefined categories
c_i	The category *i*
l_j	The label of the document *j*
Φ	Classifier function
DPC	Document-pivoted categorization
CPC	Category-pivoted categorization
DR	Dimensionality Reduction
DF	Document frequency
IG	Information Gain
t_k	Term *k*
x2	Chi square
OR	Odds Ratio
NGL	Hwee Tou **N**g, Wei Boon **G**oh, and Kok Leong **L**ow Coefficient
GSS	Luigi **G**alavotti, Fabrizio **S**ebastiani and Maria **S**imi Coefficient
LSI	Latent semantic indexing
SVD	Singular-value decomposition
f	Feature set
A	Word by document matrix

f_{ik}	Frequency of word *i* in document *k*
N	The number of documents in the collection
M	The number of words in the collection after stop word removal and word stemming
n_i	The total number of times word *i* occurs in the whole collection
tf	Term frequency
tfidf	Term frequency inverse document frequency
NB	Naive Bayes
E	Euclidian distance
β	Positive control parameter of Rocchio classifier
γ	Negative control parameter of Rocchio classifier
π	Precision
ρ	Recall
α	Similarity threshold of Ngram stemmer

List of Publications

- Tarek F. Gharib, Mena B. Habib, and Zaki T. Fayed, "Arabic Text Classification Using Support Vector Machines", Accepted for publication on the International Journal of Computer Science and Network Security IJCSNS, Vol.7, No.12, 2007.

- Mena B. Habib, Zaki T. Fayed, and Tarek F. Gharib, "A Hybrid Feature Selection Approach for Arabic Documents Classification", Egyptian Computer Science Journal ECS , Volume 28, Number 4, September 2006, pages 1-7.

- Mostafa M. Syiam, Zaki T. Fayed, and Mena B. Habib, "An Intelligent System for automated Arabic Text Categorization", International journal of intelligent computing and information systems IJICIS, Volume 6, Number 1, January 2006, pages 1-19.

Chapter 1

Introduction

1.1 Overview

The growing importance of electronic media for storing and disseminating text documents has created a burning need for tools and techniques that assist users in finding and extracting relevant information from large data repositories. Information management of well organized and maintained structured databases has been a focus of the Data Mining research for quite sometimes now. However, with the emergence of the World Wide Web, there is a need for extending this focus to mining information from unstructured and semi-structured information sources such as on-line news feeds, corporate archives, research papers, financial reports, medical records, e-mail messages, etc.

In the last 10 years content-based document management tasks (collectively known as information retrieval (IR)) have gained a prominent status in the information systems field, due to the increased availability of documents in digital form and the need to access them in flexible ways. As a consequence, there is an increased need for hardware and software solutions for storing, organizing, and retrieving the large amounts of digital text that are being produced, with an eye towards its future use.

The design of such solutions has traditionally been the object of study of information retrieval (IR), the discipline that is broadly concerned with the computer mediated access to data with poorly specified semantics.

Text classification is one of the directions that provide convenient access to a large, unstructured repository of text by partitioning an unstructured collection of documents into meaningful groups.

There are two main variants of text classification. The first is text clustering, which is concerned with finding a latent yet undetected group structure in the repository, and the second is text categorization.

Text categorization (TC – also known as text classification, or topic spotting), the activity of labeling natural language texts with thematic categories from a predefined set, is one such task. TC dates back to the early ’60s, but only in the early ’90s did it become a major subfield of the information systems discipline because of the availability of more powerful hardware. Figure 1.1 shows the distribution of publications in the field of TC since 1960 till 2006. It is clear that TC was a hot topic of research during the last ten years [Evgeniy Gabrilovich, 2007].

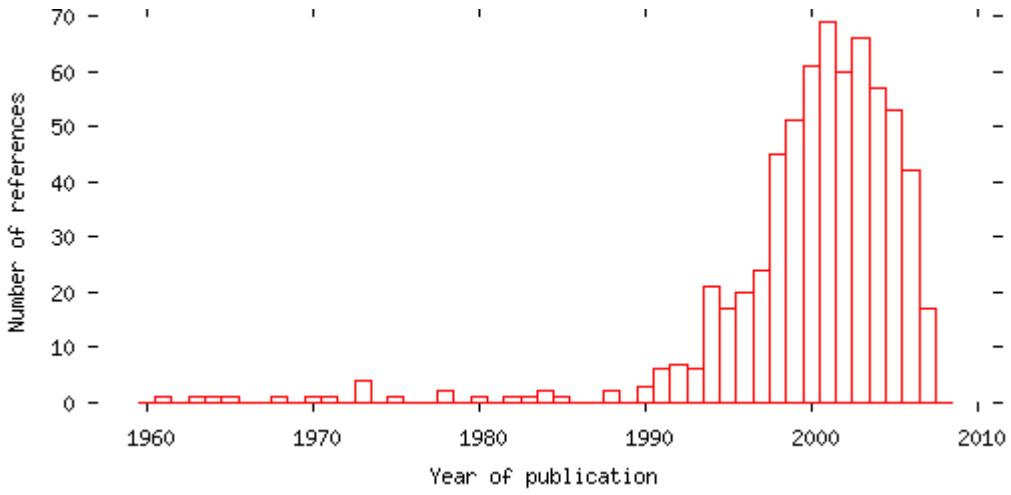

Figure 1.1 Distribution of publication dates of text categorization.

Text Categorization is a discipline at the crossroads of Machine Learning (ML) and Information Retrieval (IR), and as such it shares a

number of characteristics with other tasks such as information/knowledge extraction from texts and text mining. There is still considerable debate on where the exact border between these disciplines lies, and the terminology is still evolving. “Text mining” is increasingly being used to denote all the tasks that, by analyzing large quantities of text and detecting usage patterns, try to extract probably useful (although only probably correct) information. According to this view, TC is an instance of text mining. Figure 1.2 shows Text Categorization place between Machine Learning and Information Retrieval.

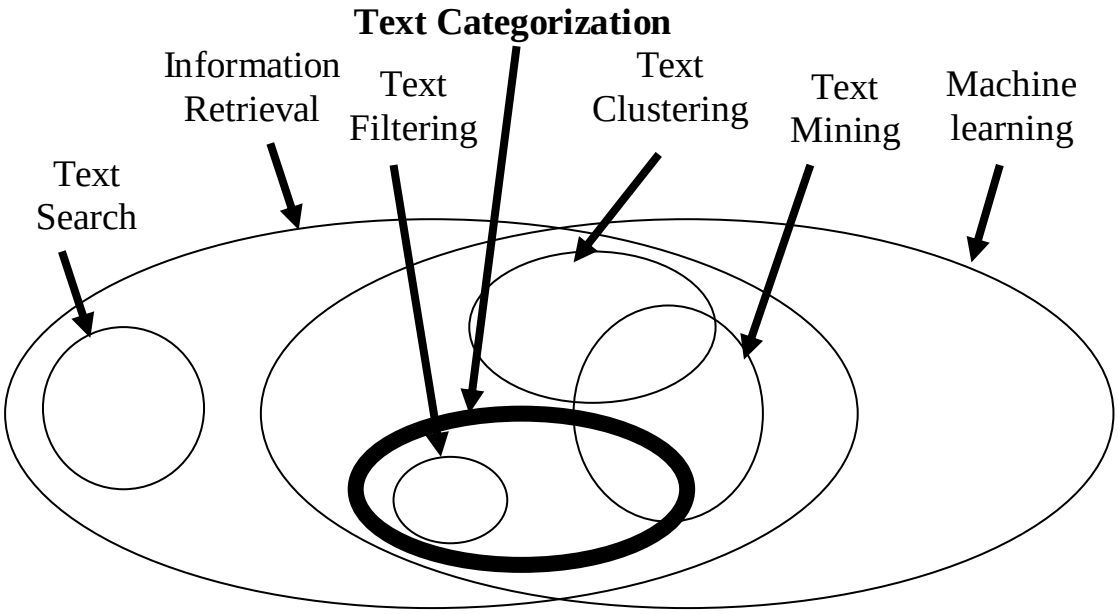

Figure 1.2 Text categorization between machine learning and information retrieval

Few researches tackled the area of Arabic text classification. Hassan Sawaf et al. [Hassan Sawaf et al, 2001] used a statistical method called maximum entropy to classify Arabic News articles. Another statistical classifier – Naïve Bayes – was used by Mohamed El Kourdi et al. [Mohamed El Kourdi et al, 2004] to categorize Arabic web documents; using the root based stemmer to extract the roots of the words.

Rehab M. Duwairi [Rehab Duwairi, 2006] proposed a distance-based classification technique on a set of 1000 document. Root based stemmer was used to extract the root of the words. The algorithm build a feature vector for each category and the test document is compared with the categories vector to find the most similar one. The comparison is performed using the Dice similarity measure.

An intelligent system based on statistical learning for searching in Arabic text was presented by Reda A. El-Khoribi el al. [Reda El-Khoribi et al, 2006]. The authors used the light stemmer for preprocessing, hidden markov models for feature extraction and Bayes classifier for classification.

The N-gram frequency statistics technique was used by Laila Khreisat [Laila Khreisat, 2006] along with the dice similarity measure. In this system an N-gram profile was generated for every document by extracting all the N-grams of the words in the document and sorting them according to their frequency on the document from most frequent to least frequent. The similarity measure used to determine the most similar document to the test document .

The first time to use SVM to classify the Arabic text was presented by Abdelwadood Moh'd A MESLEH [Abdelwadood Moh'd A MESLEH, 2007]. The CHI Square technique was used for feature selection step. No stemming algorithms were used. The results of SVM outperform the other algorithms, k-NN and Navie Bayes classifiers.

Alaa M. El-Halees [Alaa El-Halees, 2007] proposed an Arabic text categorization system that uses the maximum entropy method for classification. The paper studied the effect of stemming and words normalization accuracy. Also the authors presented a new method called part-of-speech for extracting nouns and proper nouns showing the importance of stemming and preprocessing on the classification accuracy.

1.2 Motivation and Objective

Although Text Categorization is now being applied in many contexts, very few text categorization techniques were applied for Arabic text. The main objective of this thesis is to propose a system for Arabic text categorization.

1.3 Thesis Organization

The rest of this thesis is organized as follows: Chapter 2 introduces the morphology of the Arabic language illustrating different approaches of stemming.Chapter 3 describes the machine learning approaches applied on text categorization systems; document preprocessing routines, dimensionality reduction, document representation, and classification. In Chapter 4 we propose a system for Arabic Text Categorization along with the performed experimental results. Finally in Chapter 5 the conclusion and the suggested future work are discussed.

Chapter 2

Arabic Language Morphology and Processing

2.1 Introduction

Arabic language is the largest member of the Semitic branch of the Afro-Asiatic language family that has a complex and more rich morphology than other languages like English. Holy Quran adds great value to Arabic which is widely studied and known throughout the Islamic world.

Arabic language is one of the six official languages of the United Nations (www.un.org). According to Egyptian Demographic Center, it is the mother tongue of about 300 million people in more than twenty two countries, from Morocco to Iraq, and as far south as Somalia and the Sudan [Mohammed Aljlayl et al, 2002]. Statistics shows that since 1995 when the first Arabic newspaper was launched online www.asharqalawsat.com, the number of Arabic websites has been growing exponentially. By 2000 there were about 20 thousand Arabic sites on the web, about 7% of the published sites on the web [Ahmed Abdelali et al, 2004].

2.2 Arabic Morphology

Arabic language is a highly inflected language; it has richer morphology than English. Unlike Latin-based alphabets, the orientation of writing in Arabic is from right to left; the Arabic alphabet consists of 28 letters and can be extended to ninety elements by writing additional shapes, marks, and vowels [Murat Tayli and Abdulah Al-Salamah, 1990].

Although different spoken Arabic dialects exist throughout the Arab world, there is only one form of the written language found in printed works, and it is known as (فصحه) or Standard Arabic (henceforth referred to as Arabic). Semitic languages differ in structure and grammar, but they share one characteristic that facilitated transition from one to another. In most cases, lexical forms (words) in these languages are derived from basic building blocks with tri-consonantal roots at their bases. The word building process starts with the three letters of a root and follows a regular set of word patterns. All traditional Semitic-language dictionaries and most modern ones are arranged by root. Instead of listing alphabetic entries, these dictionaries arrange words under entries of the roots that produce them. To look up a specific word, the user has to have enough knowledge to isolate the root then locate its entry. It is as though words like ascribe, describe, subscribe, circumscribe, proscribe, prescribe, inscribe were listed in an English dictionary under the Latin root "scribere" that describes the basic idea of writing/drawing [Terri De Young, 1999]. The difference is that the words grouped under an Arabic root can be analyzed down to the letters of a root and the predefined morphological patterns that created them.

One of the standard Arabic lexicons, (لسان العرب or the Language of the Arabs) lists 6,350 trilateral roots and 2,500 quadrilateral ones. Out of these, only about 1200 are still used in modern Arabic vocabulary [Hegazi et al, 1985].

Arabic words have two genders, feminine and masculine; three numbers, singular, dual, and plural; and three grammatical cases, nominative, accusative, and genitive. A noun has the nominative case

when it is subject; accusative when it is the object of a verb; and the genitive when it is the object of a preposition.

The most striking difference between Arabic and other languages is that Arabic text is usually presented without vowels. Vowels, when used, are presented by diacritical marks (ً ٌ ٍ َ ُ ِ ّ ْ) placed above or below the characters. Usually Diacritization defines how the word will be pronounced and represents its grammatical case; also it can change the meaning of the word. For example the word (مِصْرَ) means (Egypt), while the word (مُصِرّ) means (Insisting). However, the use of diacritics has lapsed in modern Arabic writing hence in the published material on the Internet.

Words are classified into three main parts of speech, nouns (including adjectives and adverbs), verbs, and particles. All verbs and some nouns are morphologically derived from list of roots. Words are formed by following fixed patterns, the prefixes and suffixes are added to the word to indicate its number, gender and tense. Figure 2.1 shows the morphological structure of the word (satalaaboon) (ستلعبون) (You (plural) will play).

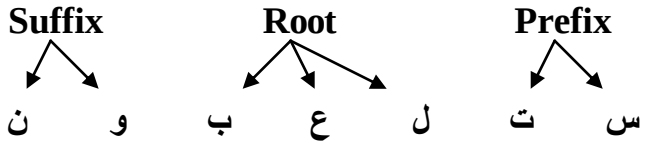

Figure 2.1 Morphological structure for the Arabic word (ستلعبون).

Arabic words constructed from the same root constitute what is traditionally called a morpho-semantic field, where semantic attributes are assigned through patterns governed by morphological rules. The

meaning that is inherent in the root is shared by all words in this field. A similar process can be noticed in English if we look at "necessitate", "necessary", "unnecessary" and "necessarily". While all four words share the basic meaning that is inherent in necess (need), they convey different semantic messages: necessitate (to produce the need), necessary (needed), unnecessary (not needed), and necessarily (in need condition/mode). We could say that adding "itate" to the root created the verb necessitate, "ary" the adjective necessary, and so on.

In general, each pattern is associated with a meaning which, when combined with the meaning conveyed by the root, gives a final meaning to the derived word. Using patterns to create different morphological variations from a root is a fairly regular process. It is similar to a mathematical formula, where the original letters are constant variables, and changing variables are letters added in the beginning, middle or end of the root. Patterns may also be indicated by vowel changes only; in these cases no letters are added to the root and, for present purposes, the structure of the word is considered unchanged. Traditionally, Arab grammarians have used the letters (ف ، ع ، ل) as generic letters to represent the root and the patterns. Patterns are based on these three letters, and, in derived words, the order of these letters is always the same: ف is first, ع second, and ل last. Table 2.1 shows a selection of noun-derivation patterns and illustrates examples of their usage.

Table 2.1 A sample of noun-derivation Patterns

Pattern	Sample Roots	Derived Nouns
فاعول	جرر (to pull)	جارور (drawer)
	حسب (to count)	حاسوب (computer)
فعال	حرم (to deny)	حرام (unlawful)
	دوم (to last)	دوام (work shift)
فعاله	زرع (to plant)	زراعه (agriculture)
	صنع (to make)	صناعه (industry)
فعيل	كبر (to grow)	كبير (big)
	غسل (to wash)	غسيل (laundry)
فعلان	زعل (to grieve)	زعلان (sad)
	كسل (to neglect)	كسلان (lazy)
فعله	حرب (to battle)	حربه (spear)
	دفع (to pay)	دفعه (installment)
فعول	خجل (to hesitate)	خجول (shy)
	شكر (to thank)	شكور (thankful)

Table 2.1 shows only a small fraction of Arabic patterns. There are hundreds more that convey all kinds of meanings. It is important to keep in mind that these patterns are not arbitrary and should not be used as so. Learners of Arabic have traditionally relied on the root-and-pattern system to practice correct use of words and to enhance their vocabulary knowledge. This system is also used to derive different forms of a base noun as explained below.

In addition to the different forms of the Arabic word that result from the derivational process, most connectors, conjunctions, prepositions, pronouns, and possession forms are attached to the Arabic surface form as prefixes and suffixes. For instance, the definitive nouns are formed by attaching the article (ال) to the immediate front of the nouns (act as "The"). The conjunction word (و) (and) is often attached to the word. The letters (ف ، ل ، ب ، ك) can be added to the front to the word as prepositions. The suffix (ة) is attached to represent the feminine of the word, (ان) is for dual masculine in the nominative case, (ين) is for dual

masculine in both the accusative and the genitive cases, (ون) is for plural masculine in the nominative case, and (ين) for plural masculine in the accusative or genitive cases. The plural suffix (ات) is used in case of feminine gender for the three grammatical cases. Also some suffixes are added as possessive pronouns, the letter (هـ) is added to represent the possessive pronoun (His), (ها) for (Her), (ي) for (My), and (هم، هن) for (Their). Table 2.2 shows different affixes that may be added to the word (معلم) (Teacher), we underlined the affixes attached to the word, also the table shows the corresponding meaning of the word in English along with its gender and number state.

Some verbs may start with one of the following prefixes (نست ، يست ، استـ ، يست ، ست ، سيس) which indicates the tense and the number of the subject. Table 2.3 shows these prefixes with the corresponding English meaning for the word (استطاع), the prefixes are underlined.

Table 2.2 Affixes that may be added to the noun word (معلم).

Arabic word	English meaning	Gender	Number
معلم	Teacher	Masculine	Singular
معلمة	Teacher	Feminine	Singular
معلمان	Two teachers	Masculine	Dual
معلمون	Teachers	Masculine	Plural (accusative, genitive)
معلمين	Teachers	Masculine	Plural (nominative)
معلمات	Teachers	Feminine	Plural
المعلم	The teacher	Masculine	Singular
والمعلم	And the teacher	Masculine	Singular
كالمعلم	Like the teacher	Masculine	Singular
معلمي	My teacher	Masculine	Singular
معلمه	His teacher	Masculine	Singular
معلمها	Her teacher	Masculine	Singular
معلمهم	Their teacher	Masculine	Singular

Table 2.3 Different prefixes that may be added to the verb (استطاع).

Arabic word	English meaning
استطاع	He was able
يستطيع	He is able
سيستطيع	He will be able
نستطيع	We are able

The main challenges that Arabic imposes on the IR community are as follows:

- **Variations in Orthography:** Certain combinations of Arabic characters are not unique in their rendition. For example, sometimes in glyphs combining HAMZA with ALEF (أ) the HAMZA is dropped (ا). This makes the glyph ambiguous as to whether the HAMZA is present.
- **Complex Morphology:** Arabic has a high degree of inflection. For example, to convey the possessive, a word shall have the letter (ى) attached to it as a suffix. There is no disjoint Arabic-equivalent of "my".
- **Broken plurals:** In English, broken plurals have some resemblance to the singular form. This is not so in Arabic broken plurals. Essentially, Arabic broken plurals do not obey morphological rules are harder to relate to singular forms.
- **Arabic words are derived:** Arabic words are usually derived from a root (a simple bare verb form) that usually contains three letters. In some derivations, one or more of the root letters may be dropped. In such cases tracing the root of the derived word would be a much more difficult problem.
- **Vowel-diacritics:** These are usually omitted from written Arabic. Such omissions cause ambiguity when interpreting words.
- **Synonyms**: These are widespread in Arabic literature of all kinds. Arabic is considered one of the richest languages in the world. This makes exact keyword match is inadequate for Arabic retrieval and classification.

2.3 Stop Words

Stop words are frequently occurring, insignificant words that appear in an article or web page (i.e. pronouns, prepositions, conjunctions, etc.).
In Arabic words like (هذه ، انه ، تكون ، قد ، بين ، جدا ، لن ، نحو ، كان) are considered stop words. These words carry no information. Stop words are filtered out prior to processing of natural language data.

A stop words list is typically language specific; also there is no definite list of stop words which all natural language processing tools incorporate.

2.4 Stemming

Stemming is a method of word standardization used to match some morphologically related words. The stemming algorithm is a computational process that gathers all words that share the same stem and have some semantic relation [Chris Paice, 1996]. The main objective of the stemming process is to remove all possible affixes and thus reduce the word to its stem. It has shown to improve performance in information retrieval tasks especially with highly inflected language like Arabic.

Stemmers equate or conflate certain variant forms of the same word like (مليار , مليارات) and (الموضوع, الموضوعات, الموضوعي, الموضوعه, الموضوعيه … etc) to combat the vocabulary mismatch problem. In this work, the term stemming is used to refer to the process which conflates related forms or groups forms into equivalence classes, including but not restricted to affixes stripping.

Many stemmers have been developed for English and other European languages. These stemmers mostly deal with the removal of suffixes as this is sufficient for most information retrieval purposes. Some of the most widely known stemmers for English are Lovins [Julie Beth Lovins, 1968] and Porter [Martin Porter, 1980] stemming algorithms

Most Arabic language stemming approaches fall into three classes: root based stemming, light stemming and statistical stemming.

2.4.1 Root-Based Stemming

Root-Based stemmers use morphological analysis to extract the root of a given Arabic word. Many algorithms have been developed for this approach.

i. Al-Fedaghi and Al-Anzi Stemming Algorithm

Al-Fedaghi and Al-Anzi algorithm [Sabah Al-Fedaghi et al, 1989] tries to find the roots and patterns of words in running text. The program uses two files, a file of trilateral roots, and a file of all patterns with all possible affixes attached to them. The algorithm does not remove any prefixes or suffixes; it just checks each word against all possible patterns with the same number of letters. If the word and the pattern match, it extracts the root which is comprised of the three letters which are in the positions of the letters (ف ، ع ، ل). If this root is found in the file of roots, then the root and pattern are returned as output. This stemming algorithm takes into consideration that some letters may be deleted or modified during the derivation of words from their roots, and deals with these situations accordingly. The algorithm was tested on various texts, and the percentage of reduced words (extracted roots) ranged from 50 to 80%.

Al-Shalabi morphology system [Riyad Al-Shalabi et al, 1998] uses a different algorithm to find roots and patterns. This algorithm follows the same strategy as the algorithm of Al-Fedaghi and Al-Anzi [Sabah Al-Fedaghi et al, 1989].

Quadrilateral roots are usually formed as extensions of trilateral roots by reduplicating the final consonant. Thus, the standard trilateral pattern "فعل" becomes the quadrilateral pattern "فعلل". The other forms of quadrilateral verbs are then obtained by adding affixes to the root.

The first step of the algorithm for quadrilateral roots is to search the input form for a correct pattern. It takes a candidate pattern and look for the four letters in the input word (corresponding to ف, ع, ل, and ل), If the letters are found it labeles their positions, pos1, pos2, pos3, and pos4. Otherwise, it chooses the next candidate pattern and tries again. Once it gets a match in all four positions it goes to the second step.

The second step is to extract the root from the input word in the positions pos1, pos2, pos3, and pos4.

For trilateral roots, the first step is to remove the longest possible prefix. Then the algorithm looks at the remainder. The three letters of the root must lie somewhere in the first four or five characters of the remainder. What is more, the first letter of the remainder is the first letter of the root since it has removed the longest possible prefix.

The algorithm checks all possible trigrams within the first five letters of the remainder. That is, it checks the following six possible trigrams: (first, second, and third letters); (first, second, and fourth); (first, second,

and fifth); (first, third and fourth); (first, third and fifth) and (first, fourth and fifth)

ii. Khoja Stemming Algorithm

Khoja has developed an algorithm [Shereen Khoja, 1999], that removes prefixes and suffixes all the time checking that it's not removing part of the root and then matches the remaining word against the patterns of the same length to extract the root. The algorithm achieves accuracy rates of up to 96%. The algorithm correctly stems most Arabic words that are derived from roots.

However, the Khoja stemmer has several weaknesses. First, the root dictionary requires maintenance to guarantee newly discovered words are correctly stemmed.

Second, the Khoja stemmer replaces a weak letter with (و) which occasionally produces a root that is not related to the original word. For example, the word (منظمات) (organizations) is stemmed to (ظمأ) (thirsty) instead of (نظم). Here the Khoja stemmer removed part of the root when it removed the prefix and then added (أ) at the end.

Third, by following a certain order of affixes, the Khoja stemmer will in some cases fail to remove all of them. For example, the terms (تستغرق) and (ركبتيه) are not stemmed although they are respectively derived from the two regular roots (غرق) and (ركب).

iii. The (ISRI) Stemming Algorithm

The Information Science Research Institute's (ISRI) presents an implementation to a root-based stemmer for Arabic which is similar to

the Khoja stemmer but without a root dictionary [Kazem Taghva et al, 2005].

iv. Other Root-Based Stemming Algorithm

Literature contains other root-based stemmer techniques. Xerox Research Centre Europe [Kenneth Beesley, 2001] has described a finite-state morphological analyzer of written standard Arabic. The underlying lexicons include about 4930 roots; the system, however, still needed additional proper names to handle multiword expressions. De Roeck and Al-Fares [Anne De Roeck et al, 2000] have presented an automatic classification algorithm for Arabic words which share the same root based only on their morphological similarities. Rogati, McCarley, and Yang have developed an unsupervised learning approach to build an Arabic stemmer [Monica Rogati et al, 2003]. The stemming model is based on statistical machine translation and it uses an English stemmer and a small parallel corpus as its sole training resources.

A simple example for convention Root-Based stemmers is illustrated in Figure 2.2. The example tries to find the root of the word (المعلومات). The stemmer removes the affixes (ال ، ات) from the word then matches the rest of the word (معلوم) against a list of different patterns. The word (معلوم) is matched with the pattern (مفعول) and hence the root is the formed by the set of words that corresponds to the three letters (ف ، ع ، ل) which is (علم) .

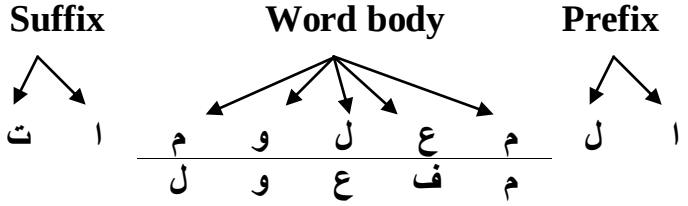

Figure 2.2 Root-based stemming for the word (المعلومات).

2.4.2 Light Stemming

Light stemming refers to the process of stripping off a small set of prefixes and/or suffixes without trying to deal with infixes or recognize patterns and find roots.

The basis of the light stemmer consists of several rounds that attempt to locate and strip out the most frequent prefixes and suffixes. The light stemming algorithm mainly processes the affixes of inflectional morphology that are typically associated with the syntax, and have relatively little influence on the word senses. As a matter of fact, the inflectional affixes are the most frequent.

Light stemmer does not need a dictionary like the Root-Based stemmer that is because it only removes list of predefined affixes from the word without checking if the remained stem is an entry for a dictionary of words.

Although light stemming can correctly conflate many variants of words into large stem classes, it can fail to conflate other forms that should go together. For example, broken (irregular) plurals for nouns and adjectives do not get conflated with their singular forms, and past tense verbs do not get conflated with their present tense forms, because they retain some affixes and internal differences.

Light stemmer is mentioned by some authors but till now there is no standard algorithm for Arabic light stemming, all trials in this field were a set of rules to strip off a small set of suffixes and prefixes, also there is no definite list of these strippable affixes.

i. Larkey and Connell light Stemming Algorithm

Leah Larkey and Margaret Connell [Leah Larkey el al, 2001] developed a light stemming algorithm and combined it with Root-Based stemmer for Khoja [Shereen Khoja, 1999]. The algorithm was tested for information retrieval tasks of both mono and bilingual. It was proved that the combined algorithm surpass the light and the Root-Based stemmers.

ii. Aljlayl and Frieder light Stemming Algorithm

Aljlayl and Frieder [Mohammed Aljlayl et al, 2002] focused on the improvement of Arabic information retrieval systems. They proposed a light stemming algorithm to minimize the sense ambiguity associated with the Root-Based stemmers and to conflate the various semantically related words into the same conflation class. This algorithm is not an aggressive as the Root-Based algorithm. The aim of this technique is not to produce the linguistic root of a given Arabic surface form; rather, it is to remove the most frequent suffixes and prefixes. They based their work on the hypothesis that developing a stemming algorithm that retains the word meaning improves the retrieval performance of an Arabic information retrieval system. Aljlayl and Frieder empirically investigated the effectiveness of the retrieval without stemming. This approach degrades retrieval precision since Arabic is highly inflected language.

They showed a statistically significant improvement over the retrieval without stemming. Later, Aljlayl and Frieder developed the light stemmer. This stemmer is mainly based on diacritics removal, normalization, and attempting to locate and strip out the most frequent prefixes and suffixes.

Aljlayl and Frieder tested the effectiveness in three cases: no stemming, Root-Based, and their light stemmer with the elimination of function words. Their results showed that all stemmers significantly perform better than no stemming at all, and light stemmer significantly outperforms the Root-Based algorithm. They found an average 87.4% and 24.1% increase in average precision over no stemming retrieval and Root-Based retrieval, respectively. This improvement has been statistically proved. At the low recall levels, the difference between the light stemmer and the other approaches is even more noticeable.

iii. Other light Stemming Algorithm

Both Darwish [Kareem Darwish, 2002] and Chen [Aitao Chen et al, 2002] designed a light Arabic stemmer that removes common prefixes and suffixes. Larkey [Leah Larkey el al, 2005] designed another light stemmer called light10; they claimed that this stemmer is significantly more effective than any other light stemmer.

2.4.3 Statistical Stemming

Statistical stemmers attempt to group word variants using clustering techniques. The different techniques vary from using character N-gram based retrieval to using co-occurrence analysis. The latter technique involves further analysis that is performed on the equivalence classes formed from the stemmers described above. This further processing is useful because those stemmers can suffer from two types of errors; terms might be incorrectly or insufficiently conflated. This process essentially involves a process of repartitioning and regrouping terms into new classes to correct the errors from earlier stemming-stages. A unique advantage that statistical stemmers enjoy is that they are somewhat more language independent.

Although n-gram systems have been used for many different languages, one would not expect them to perform well on infixing languages like Arabic. However, Mayfield [James Mayfield et al, 2001] have developed a system that combines word-based and 6-gram based retrieval, which performs remarkably well for many languages including Arabic.

De Roeck and Al-Fares [Anne De Roeck et al, 2000] used clustering technique to find the classes had the same root. Their clustering was based on morphological similarity, using a string similarity metric tailored to Arabic morphology, which was applied after removing a small number of obvious affixes. They evaluated the technique by comparing the derived clusters to "correct" classes.

Larkey [Leah Larkey et al, 2002] applied Xu and Croft's co-occurrence method to Arabic [Jinxi Xu et al, 1998]. They assumed that initial n-gram based stem classes were probably not the right starting point for languages like Arabic. However, co-occurrence or other clustering techniques can be applied to Arabic without using n-grams. Instead, they formed classes of words that mapped onto the same string if vowels were removed, and used co occurrence measures to split these classes further. The co-occurrence method did not work as well for Arabic as it did for English and Spanish. It did not produce a stemmer that worked as well as light stemmer.

A promising new class of statistical stemmers makes use of parallel corpora. Chen and Gey [Aitao Chen et al, 2002] used a parallel English-Arabic corpus and an English stemmer to cluster Arabic words into stem

classes based on their mappings to English stem classes. Rogati, McCarley, and Yang [Monica Rogati et al, 2003] use a statistical machine translation approach that learns to split words into prefix, stem, and suffix by training on a small hand annotated training set and using a parallel corpus These approaches work well considering how automated they are, but they are not as effective in an IR evaluation as a good light stemmer.

Chapter 3

Automated Text Categorization

This chapter introduced the theoretical baseline of machine learning techniques applied to the text categorization problem. First, a brief introduction of text categorization is introduced. Second, fundamental concepts, such as the documents preprocessing, term weighting, and feature selection, and classifiers construction and evaluation are introduced.

3.1 Text Categorization

3.1.1 A Mathematical Definition of Text Categorization

Text categorization may be defined as the task of assigning a Boolean value to each pair $(d_j, c_i) \in D \times C$, where D is a domain of documents and $C = \{c_1, \ldots, c_{|C|}\}$ is a set of pre-defined *categories*. A value of T assigned to (d_j, c_i) indicates a decision to file d_j under c_i, while a value of F indicates a decision not to file d_j under c_i. We are given a training set $D_{train} = \{(d_1, l_1), \ldots\ldots, (d_N, l_N)\}$ of labeled text documents where each document d_j belongs to a document set D and the label l_j of d_j is within the predefined set of categories C. The goal in text categorization is to devise a learning algorithm that given the training set D_{train} as input will generate a *classifier* (or a *hypothesis*) $\Phi : D \times C \rightarrow \{T, F\}$ that will be able to accurately classify unseen documents from D.

3.1.2 Single-Label vs. Multi-Label Text Categorization

Different constraints may be enforced on the categorization task, depending on the application requirements. For instance, we might need

to impose that, for a given integer k, exactly k (or $\leq k$, or $\geq k$) elements of C must be assigned to each element of D. The case in which exactly 1 category must be assigned to each document is often called the *single-label* (*non-overlapping categories*) case, whereas the case in which any number of categories from 0 to $|C|$ may be assigned to the same document is called the *multi-label* (*overlapping categories*) case. A special case of single-label categorization is *binary* categorization, in which each document d_j must be assigned either to category c_i or to its complement $\bar{c}_i$.

From a theoretical point of view, the binary case (hence, the single-label case too) is more general than the multi-label case, in the sense that an algorithm for binary classification can also be used for multi-label classification: one needs only transform a problem of multi-label classification under categories $\{c_1, \ldots, c_{|C|}\}$ into $|C|$ independent problems of binary classification under categories $\{c_i, \bar{c}_i\}$, for $i = 1, \ldots, |C|$. This requires, however, that categories are stochastically independent of each other. This is the assumed to be the case in this work.

The converse is not true: an algorithm for multi-label classification cannot be used for either binary or single-label classification. In fact, given a document d_j to classify, (i) the classifier might attribute $k > 1$ categories to d_j, and it might not be obvious how to choose a "most appropriate" category from them; or (ii) the classifier might attribute to d_j no category at all, and it might not be obvious how to choose a "least inappropriate" category from C. In the rest of the thesis, we will be dealing with the binary case. There are various reasons for this choice:

- The binary case is important in itself because important TC applications, including filtering, consist of binary classification problems.
- Solving the binary case also means solving the multi-label case, which is also representative of important TC applications, including automated indexing for Boolean systems.
- Most of the TC literature is couched in terms of the binary case.
- Most techniques for binary classification are just special cases of existing techniques that deal with the more general single-label case, and are simpler to illustrate than these latter.

This ultimately means that we will view the classification problem for $C = \{c_1, \ldots, c_{|C|}\}$ as consisting of $|C|$ independent problems of classifying the documents in D under a given category c_i, for $i = 1, \ldots, |C|$. A *classifier for* c_i is then a function $\Phi_i : D \times C \rightarrow \{T, F\}$.

3.1.3 Category-Pivoted vs. Document-Pivoted Text Categorization

Once we have built a text classifier there are two different ways for using it. Given a document, we might want to find all the categories under which it should be filed (*document-pivoted categorization* – DPC); alternatively, given a category, we might want to find all the documents that should be filed under it (*category-pivoted categorization* – CPC). Quite obviously this distinction is more pragmatic than conceptual, but is important in the sense that the sets C of categories and D of documents might not always be available in their entirety right from the start. It is also of some relevance to the choice of the method for building the classifier, as some of these methods (e.g. the *k*-NN method) allow the

construction of classifiers with a definite slant towards one or the other classification style.

DPC is thus suitable when documents become available one at a time over a long span of time, e.g. in filtering e-mail. CPC is instead suitable if it is possible that (i) a new category $c_{|C|+1}$ is added to an existing set $C = \{c_1, \ldots, c_{|C|}\}$ after a number of documents have already been classified under C, *and* (ii) these documents need to be reconsidered for classification under $c_{|C|+1}$. DPC is more commonly used than CPC, as the former situation is somehow more common than the latter.

3.1.4 Hard Categorization vs. Ranking Categorization

While a complete automation of the text categorization process requires a T or F decision for each pair (d_j, c_i), as argued in Section 3.1.1, a partial automation of this process might have different requirements.

For instance, given document d_j a system might simply *rank* the categories in $C = \{c_1, \ldots, c_{|C|}\}$ according to their estimated appropriateness to d_j, without taking any "hard" decision on either of them. Such a ranked list would be of great help to a human expert in charge of taking the final categorization decision, in that it would be possible for him to restrict the selection of the category (or categories) to the ones at the top of the list rather than having to examine the entire set. Alternatively, given category c_i a system might simply rank the documents in D according to their estimated appropriateness to c_i; symmetrically, for classification under c_i a human expert would just examine the top-ranked documents instead than the entire document set. These two modalities are sometimes called category-ranking

categorization and document-ranking categorization [Yiming Yang et al, 1999], respectively, and are the obvious counterparts of DPC and CPC.

Semi-automated, "interactive" classification systems [Leah Larkey et al, 1996] are useful especially in critical applications in which the effectiveness of a fully automated system may be expected to be significantly lower than that of a human professional. This may be the case when the quality of the training data is low, or when the training documents cannot be trusted to be a representative sample of the unseen documents that are to come, so that the results of a completely automatic classifier could not be trusted completely.

3.1.5 Text Categorization Applications

i. Automatic indexing for Boolean information retrieval systems

Automatic document indexing for use in information retrieval (IR) systems is the first use to which automatic categorizers were put at, and the application that spawned most of the early research in the field. The most prominent example of such IR systems is, of course, that of Boolean systems. In these systems, each document is assigned one or more keywords or keyphrases describing its content, where these keywords and keyphrases belong to a finite set of words, called controlled dictionary and often consisting of a hierarchical thesaurus. Usually, this assignment is performed by trained human indexers, and is thus an extremely costly activity. If the entries in the thesaurus are viewed as categories, document indexing becomes an instance of the document categorization task.

ii. Document organization

In general, all issues pertaining to document organization and filing may be addressed by automatic categorization techniques. For instance, at

the offices of a newspaper, incoming "classified" ads should be, prior to publication, categorized under the categories used in the categorization scheme adopted by the newspaper; typical categories might be e.g. Personals, Cars for sale, Real estate, … etc. While most newspapers would handle this application manually, those dealing with a high daily number of classified ads might prefer an automatic categorization system to choose the most suitable category for a given ad.

iii. Document filtering

Document filtering (also known as *document routing*) refers to the activity of categorizing a *dynamic*, rather than static, collection of documents, in the form of a stream of incoming documents dispatched in an asynchronous way by an information producer to an information consumer. A typical case of this is a newseed, whereby the information producer is a news agency (e.g. Reuters or Associated Press) and the information consumer is a newspaper. In this case, the filtering system should discard (i.e. block the delivery to the consumer of) the documents the consumer is not likely to be interested in (e.g. all news not concerning sports, in the case of a sports newspaper).

Filtering can be seen as a special case of categorization with non-overlapping categories, i.e. the categorization of incoming documents in two categories, the relevant and the irrelevant. Additionally, a filtering system may also perform a further categorization into topical categories of the documents deemed relevant to the consumer; in the example above, all articles about sports are deemed relevant, and should be further subcategorized according to which sport they deal with, so as to allow individual journalists specialized in individual sports to access only documents of high prospective interest for them.

iv. Hierarchical categorization of Web pages

Automatic document categorization has recently aroused a lot of interest also for its possible Internet applications. One of these is automatically classifying Web pages, or sites, into one or several of the categories that make up the commercial hierarchical catalogues hosted by popular Internet portals. When Web documents are catalogued in this way, rather than addressing a generic query to a general purpose Web search engine a searcher may find it easier to first navigate in the hierarchy of categories and then issue his search from (i.e. restrict his search to) a particular category of interest.

With respect to other previously discussed TC applications, the automatic categorization of Web pages has two essential peculiarities:

(1) The hypertextual nature of the documents: hyperlinks constitute a rich source of information, as they may be understood as statements of relevance of the linked page to the linking page.

(2) The hierarchical structure of the category set: this may be used by decomposing the classification problem into a series of smaller classification problems corresponding each to a branching decision at an internal node.

3.2 Documents Preprocessing

In order to cluster or classify text documents by applying machine learning techniques, documents should first be preprocessed. In the preprocessing step, the documents should be transformed into a representation suitable for applying the learning algorithms. The most widely used method for document representation is the vector space

model introduced by Gerard Salton [Gerard Salton et al, 1975], which we have also decided to employ.

In this model, each document is represented as a vector *d*. Each dimension in the vector *d* stands for a distinct term (word) in the term space of the document collection.

A term in the document collection can stand for a distinct single-word, a stemmed word or a phrase. In vector space representation, defining terms as distinct single words is referred to as “bag of words” representation. Some researchers state that using phrases rather than single words to define terms produce more accurate classification results [William Cohen et al, 1996][Johannes Fuernkranz et al, 1998]; whereas others argue that using single words as terms does not produce worse results [Susan Dumais et al, 1998][Mehran Sahami, 1998]. As “bag of words” representation is the most frequently used method for defining terms and it is computationally more efficient than the phrase representation, we have chosen to adapt this method to define terms of the feature space.

One challenge emerging when terms are defined as single words is that the feature space becomes very high dimensional. In addition, words which are in the same context such as "سياسة" and "سياسي" are defined as different terms. So, in order to define words that are in the same context with the same term and consequently to reduce dimensionality we have decided to define the terms as stemmed words.

Documents preprocessing routines include stop words removal to remove insignificant words, stemming to group words share the same

context. After that, the super vector is constructed. The super vector is the set of words that appear in the documents collection at least one time

Stop words are words in such as pronouns, prepositions and conjunctions that are used to provide structure in the language rather than content. These words are encountered very frequently and carry no useful information about the content and thus the category of documents. Removing stop words from the documents is very common in information retrieval. Eliminating the stop words from the documents will lead to a drastic reduction in the dimensionality of the feature space.

In order to define words that are in the same context with the same term and consequently to reduce dimensionality, we have decided to define the terms as stemmed words. Stemming approaches and stops words in Arabic language are previously discussed in details on chapter 2.

3.3 Dimensionality Reduction

The Vector Space Model implies the dimensionality of the space of the super vector to be the same of the vocabulary and it can reach the tens of thousands of terms. Even using stemming and stop-words lists, the dimension of the super vector remains still high and many unnecessary words are still present in the vocabulary. These words may provide no contribution to categorization performance and sometimes decrease accuracy.

Dimensionality Reduction (DR) is also beneficial since it tends to reduce over fitting, that is, the phenomenon by which a classifier is tuned also to the contingent characteristics of the training data rather than just

the constitutive characteristics of the categories. Classifiers those over fit the training data are good at reclassifying the data they have been trained on, but much worse at classifying previously unseen data. Experiments have shown that, in order to avoid over fitting a number of training examples roughly proportional to the number of terms used is needed; Fuhr and Buckley [Norbert Fuhr et al, 1991] have suggested that 50 – 100 training examples per term may be needed in TC tasks. This means that, if DR is performed, over fitting may be avoided even if a smaller amount of training examples is used.

There are two distinct ways of viewing DR in terms of the nature of the resulting terms:

- DR by term selection: where the final set of terms representing documents is subset of the original set of terms, it is the most popular method for DR and it is used in this work;
- DR by term extraction: where the final set of terms representing documents is not of the same type of the original set of terms (e.g., if the original terms are words, the terms used to represent documents may not be words at all), but are obtained by combinations or transformations of the original ones.

3.3.1 Feature Selection

Feature selection algorithms can be divided into two types: wrapper and filter, based on whether or not feature selection is done independently of categorization model [George John et al, 1994][Daphne Koller et al, 1996]. The wrapper algorithms consider the performance of a particular learning algorithm while an optimal subset of feature terms is selected. They measure the goodness of feature term set by the performance of the algorithm. Their results are dependant of the algorithm. But they are

prohibitively expensive and intractable sometimes. In contrast to wrapper approach, the filter algorithms measure the weight of every term and use rank criterion principle to select feature terms while any machine learning algorithm is not considered. They are dependent of the collection of documents and are independent of any algorithm. Filter methods are mostly used in reality for document categorization because of the computational expense of the wrapper methods [Sanmay Das, 2001].

Several methods to select only good features (or eliminate bad features) have been proposed [Yiming Yang, 1997]. These methods uses informative ness functions to evaluate the quality scores for the words and removes the terms according to some empirical rules:

- it can select a lower-bound value and remove all words with score less than it;
- an upper-bound can be chosen and all elements with score larger than it are removed;
- both lower-bound and upper-bound can be chosen and all elements outside the thresholds are removed;
- the first *k* elements with the highest or the lowest score can be selected, removing the others;

Many different functions have been proposed in the literature each one based on different assumptions. They can be used separately or in cascade. In the following, we present the most common functions describing the assumption which they come from.

i. Document Frequency (*DF*)

Document frequency *DF* uses the number of documents in which each word appears as the informative score for the term. If the ratio of *DF* to

the total number of documents in the collection of a word is too close to 1 then this word has a uniform distribution over the collection. It means that the term most probably appears in almost all documents and it is not a good feature to discriminate the topic. Moreover the words with too small values for *DF* appear in only few documents and, even if in the same class, they can not be discriminative features for that class. Therefore this score is often used to select those words having the *DF* values included to a range of values.

ii. Information Gain (*IG*)

Information gain measures the number of bits of information gained for category prediction when the presence or absence of a term in a document is known. When the set of possible categories is $C = \{c_1, \ldots, c_{|C|}\}$, the *IG* for each unique term t_k is calculated as follows [Yiming Yang, 1997]:

$$IG(t_k) = -\sum_{i=1}^{|C|} P(c_i).\log P(c_i) + P(t_k).\sum_{i=1}^{|C|} P(c_i \mid t_k).\log P(c_i \mid t_k) + P(\bar{t}_k).\sum_{i=1}^{|C|} P(c_i \mid \bar{t}_k).\log P(c_i \mid \bar{t}_k)$$

As seen from the equation, *IG* calculates the decrease in entropy when the feature is given vs. absent. $P(c_i)$ is the prior probability of category c_i. It can be estimated from the fraction of documents in the training set belonging to category c_i. $P(t_k)$ is the prior probability of term t_k. It can be estimated from the fraction of documents in the training set in which term t_k is present. Likewise, $P(\bar{t}_k)$ can be estimated from the fraction of documents in the training set in which term t is absent.

If the gain is high, that feature is considered important and informative for a topic and it should not be removed; on the contrary, it does not give information about the topic and it can be removed. Usually using the IG measure we can eliminate a high number of features reducing the dimension of the super vector and improving the results of the system. However this scheme can be applied only to the classification task.

iii. Chi square (x2**)**

x2 statistic measures the dependency between the term t_k and the class c_i. That is, it measures to what degree a certain term is indicative of membership or non-membership of a document in a certain category. The x2 is used for the task of document categorization as follows [Yiming Yang, 1997]:

$$\chi^2\ (t_k, c_i) = \frac{|T_r|.[P(t_k, c_i).P(\bar{t_k}, \bar{c_i}) - P(t_k, \bar{c_i}).P(\bar{t_k}, c_i)]^2}{P(t_k).P(\bar{t_k}).P(c_i).P(\bar{c_i})}$$

Where T_r is the number of total training documents. Category specific x2 statistic scores for a term t_k can be combined into a global x2 statistic score for that term in the following two ways:

$$\chi^2_{avg}\ (t_k, c_i) = \sum_{i=1}^{|C|} P(c_i).\chi^2(t_k, c_i)$$

or

$$\chi^2_{\max}\ (t_k, c_i) = \max_{i=1}^{|C|}\{\chi^2(t_k, c_i)\}$$

x2 has value zero if t_k and c_i are independent [Rafael Calvo et al, 2000]. Terms that have lower x2 values than a predetermined threshold are eliminated.

iv. Odds Ratio (*OR*)

Odds Ratio is used in Information Retrieval to rank documents on the basis of the relations existing between their features and the classes:

$$OR\ (t_k, c_i) = \frac{P(t_k, c_i).(1 - P(t_k, \overline{c_i}))}{(1 - P(t_k, c_i)).P(t_k, \overline{c_i})}$$

Two different measures can be computed based on OR:

$$OR_{avg}\ (t_k, c_i) = \sum_{i=1}^{|C|} P(c_i).OR(t_k, c_i)$$

or

$$OR_{max}\ (t_k, c_i) = \max_{i=1}^{|C|} \{OR(t_k, c_i)\}$$

In [Dunja Mladenic, 1998] and [Dunja Mladenic et al, 1998] Mladenic uses *OR* for text filtering in text categorization obtaining quite good results.

v. NGL coefficient (NGL)

NGL Coefficient (sometimes called correlation coefficient) is a simplified variant of the x2 statistics proposed by Hwee Tou Ng [Hwee Tou Ng et al, 1997]. Its formula is defined as fellow:

$$NGL\ (t_k, c_i) = \frac{\sqrt{|T_r|}.[P(t_k, c_i).P(\bar{t_k}, \bar{c_i}) - P(t_k, \bar{c_i}).P(\bar{t_k}, c_i)]^2}{\sqrt{P(t_k).P(\bar{t_k}).P(c_i).P(\bar{c_i})}}$$

The rationale behind the *NGL* correlation coefficient is related to the finding that local dictionary yields a better set of features.

Global *NGL* coefficient can be computed in two ways $NGL_{avg}\ (t_k, c_i)$ and $NGL_{max}\ (t_k, c_i)$.

vi. GSS coefficient (GSS)

GSS Coefficient is another simplified variant of the x2 statistics proposed by Galavotti [Luigi Galavotti et al, 2000], which is defined as:

$$GSS\ (t_k, c_i) = P(t_k, c_i).P(\bar{t_k}, \bar{c_i}) - P(t_k, \bar{c_i}).P(\bar{t_k}, c_i)$$

The positive values correspond to features indicative of membership, while negative values indicate non-membership. Therefore, only the positive terms are considered.

Similar to Chi square, Odds Ratio and *NGL* coefficient, global *GSS* coefficient can be computed in two ways $GSS_{avg}\ (t_k, c_i)$ and $GSS_{max}\ (t_k, c_i)$.

3.3.2 Feature Extraction

Feature extraction (sometimes called Re-parameterization) is the process or constructing new features as combinations or transformations of the original features. Two term extraction methods have been experimented with in TC, namely term clustering and latent semantic indexing.

i. Term Clustering

Term clustering tries to group words with a high degree of pairwise semantic relatedness, so that the groups (or their centroids, or a representative of them) may be used instead of the terms as dimensions of the vector space. Li and Jain [Yong Hong Li et al, 1998] viewed semantic relatedness between words in terms of their co-occurrence and co-absence within training documents.

ii. Latent Semantic Indexing

Latent semantic indexing (LSI) [Scott Deerwester et al, 1990] is based on the assumption that there is some underlying or latent structure in the pattern of word usage across documents, and that statistical techniques can be used to estimate this structure. LSI uses singular-value decomposition (SVD), a technique closely related to eigenvector decomposition and factor analysis. This technique compresses document vectors into vectors of a lower-dimensional space whose dimensions are obtained as combinations of the original dimensions by looking at their patterns of co-occurrence.

3.4 Feature Selection Approaches

There are two quite distinct ways of viewing DR, depending on whether the task is approached locally (i.e. for each individual category, in isolation of the others) or globally:

- Local feature selection: for each category c_i, features are chosen in terms of which the classifier for category c_i will operate [Hwee Tou Ng et al, 1997][Zhaohui Zheng et al, 2003][Bong Chih How et al, 2004][Bong Chih How et al, 2005];

- Global feature selection: features are chosen in terms of which the classifier for all categories $C = \{c_1, \ldots, c_{|C|}\}$ will operate [Yiming Yang, 1997][Yiming Yang 1999][Bong Chih How et al, 2005].

3.4.1 Local Feature Selection

In local feature selection, feature set *f* is extracted from each category of interest (positive class) of which the specific classifier will operate. Feature set extracted from c_1 thus will be differed from feature set derived from category other than c_1. This would mean that each document d_j has a different representation for each category c_i; in practice, though, this means that different subsets of d_j's original representation are used when categorizing under the different categories. The feature set *f* can be harvested in two ways: Selecting terms that belongs only to the interested class using relevant documents only (local dictionary); or combining features of both the positive and negative classes using relevant and irrelevant documents (universal dictionary) [Zhaohui Zheng et al, 2003]. In this thesis, universal dictionary is used as suggested in [Zhaohui Zheng et al, 2003].

3.4.2 Global Feature Selection

In global feature selection a feature set *f*, is extracted from all the classes $C = \{c_1, \ldots, c_{|C|}\}$. Selected set must preserve and obtain if possible every category-specific significant feature that may be important to classification task and can only safely removes features that will not be relevant to classification task. In this approach all documents have the same representation for all classes.

3.5 Documents Representation

Given a collection of documents, its feature vectors are represented by a word-by-document matrix *A*, where each entry represents the weight of a word in a document, i.e.,

$$A = (a_{ik})$$

Where a_{ik} is the weight of word *i* in the document *k*, since every word does not normally appear in each document, the matrix *A* is usually sparse. The number of rows, *M*, of the matrix corresponds to the number of words in the super vector *W*. *M* can be very large.

There are several ways of determining the weight a_{ik} of word *i* in document *k*, but most of the approaches are based on two empirical observations regarding text [Kjersti Aas et al, 1999]:

- The more times a word occurs in a document, the more relevant it is to the topic of the document.
- The more times the word occurs throughout all documents in the collection, the more poorly it discriminates between documents.

Let f_{ik} be the frequency of word *i* in document *k*, *N* the number of documents in the collection, *M* the number of words in the collection after stop word removal and word stemming, and n_i the total number of times word *i* occurs in the whole collection. Traditional methods for term weighting are used to determine the most suitable one for Arabic text categorization task.

3.5.1 Boolean Weighting

The simplest approach is to let the weight be 1 if the word occurs in the document and 0 otherwise:

$$a_{ik} = \begin{cases} 1 & \text{if } f_{ik} > 0 \\ 0 & \text{otherwise} \end{cases}$$

3.5.2 Term Frequency Weighting (*tf*)

Another simple approach is to use the frequency of the word in the document:

$$a_{ik} = f_{ik}$$

3.5.3 Term Frequency Inverse Document Frequency Weighting (*tfidf*)

The previous two schemes do not take into account the frequency of the word throughout all documents in the collection. A well-known approach for computing word weights is the *tfidf* (term frequency-inverse document frequency) weighting which assigns the weight to word i in document k in proportion to the number of occurrences of the word in the document, and in inverse proportion to the number of documents in the collection for which the word occurs at least once. The *tfidf* weight can be represented by the following function:

$$a_{ik} = f_{ik} * \log\left(\frac{N}{n_i}\right)$$

3.5.4 Normalized-*tfidf* weighting

The *tfidf* weighting does not take into account that documents may be of different lengths. The normalized-*tfidf* weighting is similar to the *tfidf*

weighting except for the fact that length normalization is used as part of the word weighting formula.

$$a_{ik} = \frac{f_{ik} * \log\left(\frac{N}{n_i}\right)}{\sqrt{\sum_{j=1}^{M}\left[f_{ik} * \log\left(\frac{N}{n_i}\right)\right]^2}}$$

3.6 Classification

Classification (also called Supervised Learning) is the process of finding a set of models (or functions) which describe and distinguish data classes or concepts, for the purpose of predicting the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known). In the context of document classification our objects are the documents and we aim to automatically assign thematic labels like "رياضة" (Sports), "فنون" (Arts), or "سياسة" (Politics) to these documents.

There are two different ways to build a classifier:

- Parametric: According to this approach, training data are used to estimate parameters of a distribution or discrimination function on the training set. The main example of this approach is the probabilistic Naive Bayes classifier.
- Non-parametric: These classifiers base classification on the training set itself. This approach may be further subdivided in two categories:
    - Example-based: According to this approach, the document d to be categorized is compared against the training set of documents. The document is assigned to the class of the

most similar training documents. Example of this approach is *k*-Nearest Neighbor (*K*-NN) classifier;

- Profile-based: In this approach, a profile (or linear classifier) for the category, in the form of a vector of weighted terms, is extracted from the training documents pre-categorized under c_i. The profile is then used as a training data against the document *d* to be categorized. Example of this approach is Rocchio classifier and Support Vector Machines (SVM).

The most famialir text categorization algorithms will be presents in the following sections.

3.6.1 Naive Bayes Classifier

The Naive Bayes (NB) classifier is a probabilistic model that uses the joint probabilities of terms and categories to estimate the probabilities of categories given a test document [Tom Mitchell, 1997]. The naive part of the classifier comes from the simplifying assumption that all terms are conditionally independent of each other given a category. Because of this independence assumption, the parameters for each term can be learned separately and this simplifies and speeds the computation operations compared to non-naive Bayes classifiers.

There are two common event models for NB text classification, discussed by Mccallum and Nigam [Andrew Mccallum et al, 1998], multinomial model and multivariate Bernoulli model. In both models classification of test documents is performed by applying the Bayes' rule [Tom Mitchell, 1997]:

$$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)}$$

Where d_j is a test document and c_i is a category. The posterior probability of each category c_i given the test document d_j, i.e. $P(c_i \mid d_j)$, is calculated and the category with the highest probability is assigned to d_j. In order to calculate $P(c_i \mid d_j)$, $P(c_i)$ and $P(d_j \mid c_i)$ have to be estimated from the training set of documents. Note that $P(d_j)$ is same for each category so we can eliminate it from the computation. The category prior probability, $P(c_i)$, can be estimated as follows:

$$P(c_i) = \frac{\sum_{j=1}^{N} y(d_j, c_i)}{N}$$

Where, N is number of training documents and $y(d_j, c_i)$ is defined as follows:

$$y(d_j, c_i) = \begin{cases} 1 & \text{if } d_j \in c_i \\ 0 & \text{otherwise} \end{cases}$$

So, prior probability of category c_i is estimated by the fraction of documents in the training set belonging to c_i. $P(d_j \mid c_i)$ parameters are estimated in different ways by the multinomial model and multivariate Bernoulli model. We present these models as follow.

i. Multinomial Model

Multinomial model for Naive Bayes classification is the event model we used and evaluated in our study. In the multinomial model a document

d_j is an ordered sequence of term events, drawn from the term space T. The Naive Bayes assumption is that the probability of each term event is independent of term's context, position in the document, and length of the document. So, each document d_j is drawn from a multinomial distribution of terms with number of independent trials equal to the length of d_j. The probability of a document d_j given its category c_i can be approximated as:

$$P(d_j \mid c_i) \approx \prod_{k=1}^{|dj|} P(t_k \mid c_i)$$

Where $| d_j |$ is the number of terms in document d_j; and t_k is the k^{th} term occurring in document d_j. Thus the estimation of $P(d_j \mid c_i)$ is reduced to estimating each $P(t_k \mid c_i)$ independently. The following Bayesian estimate is used for $P(t_k \mid c_i)$:

$$P(t_k \mid c_i) = \frac{1 + TF(t_k, c_i)}{|T| + \sum_{t_l \in |T|} TF(t_l, c_i)}$$

Here, $TF(t_k \mid c_i)$ is the total number of times term t_k occurs in the training set documents belonging to category c_i. The summation term in the denominator stands for the total number of term occurrences in the training set documents belonging to category c_i. This estimator is called Laplace estimator and assumes that the observation of each word is a priori likely [Thorsten Joachims, 1997].

ii. Multivariate Bernoulli Model

In Multivariate Bernoulli model a document is represented by a vector of binary features indicating the terms that occur and that do not occur in the document. Here, the document is the event and absence or presence of

terms is the attributes of the event. The Naive Bayes assumption is that the probability of each term being present in a document is independent of the presence of other terms in a document. To state differently, the absence or presence of each term is dependent only on the category of the document. Then, $P(d_j \mid c_i)$, the probability of a document given its category is simply the product of the probability of the attribute values over all term attributes:

$$P(d_j \mid c_i) = \prod_{k=1}^{|T|} (B_{jk} \cdot P(t_k \mid c_i) + (1 - B_{jk})(1 - P(t_k \mid c_i)))$$

Where |T| is the number of terms in the training set and B_{jk} is defined as follows:

$$B_{jk} = \begin{cases} 1 & \text{if term } t \text{ appears in document } d_j \\ 0 & \text{otherwise} \end{cases}$$

Thus, a document can be seen as a collection of multiple independent Bernoulli experiments, one for each term in the term space. The probabilities of each of these term events are defined by the class-conditional term probabilities $P(t_k \mid c_i)$. We can estimate the probability of term t_k in category c_i as follows:

$$P(t_k \mid c_i) = \frac{1 + \sum_{j=1}^{N} B_{jk} \cdot y(d_j, c_i)}{2 + \sum_{j=1}^{N} y(d_j, c_i)}$$

where, N is number of training documents and $y(d_j \mid c_i)$ is defined as shown above.

Mccallum and Nigam [Andrew Mccallum et al, 1998] found, by comparing both models over four different corpora, that the multinomial model was almost uniformly better than the multivariate Bernoulli model. In empirical results on five real-world corpora they found that the multinomial model reduces error by an average of 27%, and sometimes by more than 50%, that is why we have selected this model to be used in this work.

3.6.2 *K*-Nearest Neighbor Classifier

To classify an unknown document vector *d*, the *k*-nearest neighbor (*k*-NN) algorithm ranks the document's neighbors among the training document vectors, and use the class labels of the *k* most similar neighbors to predict the class of the input document [Fabrizio Sebastiani, 1999][Fabrizio Sebastiani, 2002]. The classes of these neighbors are weighted using the similarity of each neighbor to *d*, where similarity may be measured by for example the Euclidean distance or the cosine between the two document vectors. The Euclidean distance is used as a conventional method for measuring distance between two documents, the formula of the Euclidean distance between documents $d_1(w_{11},w_{12},\ldots,w_{1n})$ and $d_2(w_{21},w_{22},\ldots,w_{2n})$ is as follow:

$$E\,(d_1, d_2) = \sqrt{\sum_{i=1}^{n}(w_{2i} - w_{1i})^2}$$

k-NN has been applied to text categorization since the early days of its research. However, it has a set of drawbacks. *k*-NN is a lazy learning example-based method that does not have a off-line training phase. The main computation is the on-line scoring of training documents given a test document in order to find the *k* nearest neighbors, this makes *k*-NN not efficient because nearly all computation takes place at classification

time rather than when the training examples are first encountered, *k*-NN time complexity is *O*(*N***M*) where *N* is number of training documents and *M* is the number of terms in the super vector. Moreover, *k*-NN classifier has a major drawback of selecting the value of *k*, the success of classification is very much dependent on this value. The Rocchio method however can deal with those problems to some extent as shown in the next section.

3.6.3 Rocchio Classifier

Rocchio is the classic profile-based classifier used for document routing or filtering in information retrieval [J. Rocchio, 1971]. In this method, a prototype vector is built for each class c_i, and a document vector d is classified by calculating the distance between d and each of the prototype vectors [Fabrizio Sebastiani, 1999][Fabrizio Sebastiani, 2002]. The prototype vector for class c_i is computed as the weighted average vector over all training document vectors that belong to class c_i. This means that learning is very fast for this method compared to the *k*-NN classifier.

The weighted average of a category c_i (w_{i1},w_{i2},…,w_{in}) is computed as follow:

$$w_{ik} = \beta \sum_{d_j \in POS_i} \frac{w_{jk}}{|POS_i|} - \gamma . \sum_{d_j \in NEG_i} \frac{w_{jk}}{|NEG_i|}$$

where w_{jk} is the weight of the term t_k in document d_j, POS_i is the set of documents that belongs to c_i (positive examples), and NEG_i is the set of documents that doesn't belongs to c_i (negative examples). In this formula, β and γ are control parameters that allow setting the relative importance

of positive and negative examples. For instance, if β is set to 1 and γ to 0, the profile of c_i is the centroid of its positive training examples. In general, the Rocchio classifier rewards the closeness of a test document to the centroid of the positive training examples, and its distance from the centroid of the negative training examples. The role of negative examples is usually de-emphasized, by setting β to a high value and γ to a low one (e.g. use β=1.6 and γ=0.4) [William Cohen et al, 1999].

The Rocchio method deals with *k*-NN problems to some extent. It uses the generalized instances to replace the whole collection of training instances by summarizing the contribution of the instances belonging to each category. Besides its efficiency this method is easy to implement, since learning a classifier basically comes down to averaging weights and classifying a new instance only needs computing the Euclidean distance between the new instance and the generalized instances. It can be regarded as a similarity-based algorithm. Its time complexity is considered to be $O(L*M)$ where L is number of generalized instances and M is the number of terms in the super vector. Moreover, the Rocchio method can deal with noise to some extent via summarizing the contribution of the instances belonging to each category. For example, if a feature mainly appears in many training instances of a category, its corresponding weight in the generalized instance will have a larger magnitude for this category. Also if a feature mainly appears in training instances of other categories, its weight in the generalized instance will tend to zero. Therefore, the Rocchio classifier can distill out certain relevant features to some extent. On the other hand, one drawback of the Rocchio classifier is it restricts the hypothesis space to the set of linear separable hyper-plane regions, which has less expressiveness power than that of *k*-NN algorithms.

3.6.4 Support Vector Machine Classifier

Support Vector Machines (SVM) is a relatively new class of machine learning techniques first introduced by Vapnik [Vladmimir Vapnik, 1995] and has been introduced in TC by Joachims [Thorsten Joachims, 1998]. Based on the *structural risk minimization* principle from the computational learning theory, SVM seeks a decision surface to separate the training data points into two classes and makes decisions based on the *support vectors* that are selected as the only effective elements in the training set.

Given a set of N linearly separable points $S = \{x_i \in R^n \mid i = 1, 2, \ldots, N\}$, each point x_i belongs to one of the two classes, labeled as $y_i \in \{-1, +1\}$. A *separating hyper-plane* divides S into 2 sides, each side containing points with the same class label only. The *separating hyper-plane* can be identified by the pair (w,b) that satisfies

$$w \,.\, x + b = 0$$

$$\text{and} \begin{cases} w \cdot x_i + b \geq +1 \text{ if } y_i = +1 \\ w \cdot x_i + b \leq -1 \text{ if } y_i = -1 \end{cases}$$

for $i = 1, 2, \ldots, N$; where the dot product operation (.) is defined by

$$w \,.\, x = \sum_i w_i x_i$$

for vectors w and x. Thus the goal of the SVM learning is to find the optimal separating hyper-plane (OSH) that has the maximal margin to both sides. This can be formularized as:

$$\text{minimize } \frac{1}{2} w.w$$

$$\text{subject to } \begin{cases} w \cdot x_i + b \geq +1 \text{ if } y_i = +1 \\ w \cdot x_i + b \leq -1 \text{ if } y_i = -1 \end{cases} \quad \text{for } i = 1, 2, \ldots, N$$

Figure 3.1 shows how SVM finds the OSH

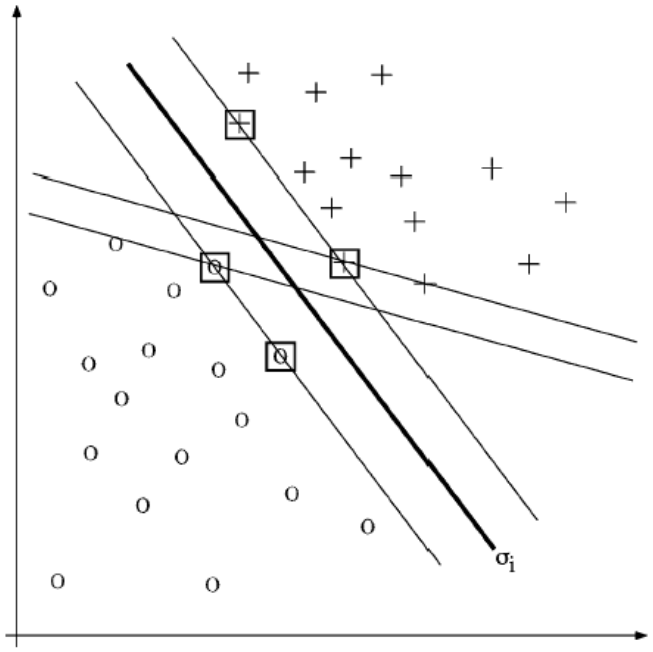

Figure 3.1 Learning support vector classifiers.

The small crosses and circles in figure 3.1 represent positive and negative training examples, respectively, whereas lines represent decision surfaces. Decision surface σi (indicated by the thicker line) is, among those shown, the best possible one, as it is the middle element of the widest set of parallel decision surfaces (i.e., its minimum distance to any training example is maximum). Small boxes indicate the support vectors.

During classification, SVM makes decision based on the OSH instead of the whole training set. It simply finds out on which side of the OSH the test pattern is located. This property makes SVM highly competitive, compared with other traditional pattern recognition methods, in terms of

computational efficiency and predictive accuracy [Yiming Yang et al, 1999].

The method described is applicable also to the case in which the positives and the negatives are not linearly separable. Yang and Liu [Yiming Yang et al, 1999] experimentally compared the linear case (namely, when the assumption is made that the categories are linearly separable) with the nonlinear case on a standard benchmark, and obtained slightly better results in the former case.

Support Vector Machines have been applied successfully in many text classification tasks due to their principle advantages [Thorsten Joachims, 1998]:

- They are robust in high dimensional spaces. Over-fitting does not affect so much the computation of the final decision margin.
- Any feature is important. Even some features that could be considered as irrelevant ones have been found to be good when calculating the margin.
- They are robust when there is a sparsely of samples.
- Most text categorization problems are linearly separable.

3.7 Classification Evaluation

Classification generalization is usually measured in terms of the classic information retrieval notions of precision (π) and recall (ρ) [Ricardo Baeza-Yates et al, 1999], adapted to the case of text categorization. Precision (π_i) with respect to c_i is the probability that if a random document d_x is classified under c_i, this decision is correct. Analogously, recall (ρ_i) with respect to c_i is defined the probability that, if

a random document d_x ought to be classified under c_i , this decision is taken. Precision and recall are calculated as follow:

$$\pi_i = \frac{a_i}{a_i + b_i} \qquad \rho_i = \frac{a_i}{a_i + c_i}$$

Where:

" a " the number of documents correctly assigned to this category.

" b " the number of documents incorrectly assigned to this category.

" c " the number of documents incorrectly rejected from this category.

" d " the number of documents correctly rejected from this category.

For obtaining estimates of π and ρ, two different methods may be adopted:

- Microaveraging: π and ρ are obtained by summing over all individual decisions:

$$\pi^{\mu} = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|} a_i + b_i} \qquad \rho^{\mu} = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|} a_i + c_i}$$

- Macroaveraging: π and ρ are first evaluated "locally" for each category, and then "globally" by averaging over the results of the different categories:

$$\pi^{M} = \frac{\sum_{i=1}^{|C|} \pi_i}{|C|} \qquad \rho^{M} = \frac{\sum_{i=1}^{|C|} \rho_i}{|C|}$$

Since most classifiers can be arbitrarily tuned to emphasize recall at the expense of precision (and vice versa), only combinations of the two are significant. The most popular way to combine the two is the function $F_{\beta i} = \frac{(B^2+1)\pi_i\rho_i}{\beta^2\pi_i + \rho_i}$, for some value $0 \le \beta \le \infty$; usually, β is taken to be equal to 1, which means that the $F_{\beta i}$ function becomes $F_{1i} = \frac{2\pi_i\rho_i}{\pi_i + \rho_i}$, i.e. the harmonic mean of precision and recall. Similar to precision and recall, F_β function can be estimated using two methods: Micro average, and Macro average. *F1* measure is the evaluation criteria used in this work.

Chapter 4

The Proposed System Phases

4.1 Introduction

In this chapter we will discuss a classification system for Arabic documents in details. This chapter will be organized as follows: the system architecture will be introduced; the text collection used in the experiments will be described. Finally our implementations, modifications and experiments performed for each stage will be presented.

4.2 System Architecture

The proposed system contains a set of phases that describe the documents preprocessing routines, document representation techniques and classification process. Figure 4.1 shows the proposed Arabic text categorization system.

During the training phase (shown by the solid line), the document is converted from its format (html, xml, doc, … etc) to a raw text format. After conversion, stop words are removed according to a predefined set of words. Then documents are stemmed using the suitable stemming algorithm, at this step documents are now ready for selecting the most effective terms (words) using feature scoring method. Both stemming and feature selection phases are considered dimensionality reduction phase.

The terms selected form the feature selection phase are used to construct the super vector which includes all words that appear at least

once in the documents collection. All training documents are then represented as a vector of the terms of the super vector by assigning a weight to each term indicating its importance in identifying the document topic. Finally the classifier is trained using the training documents.

In the classification phase (shown by the solid line) the test document is converted, stemmed and represented as a vector before being classified by the pre-trained classifier.

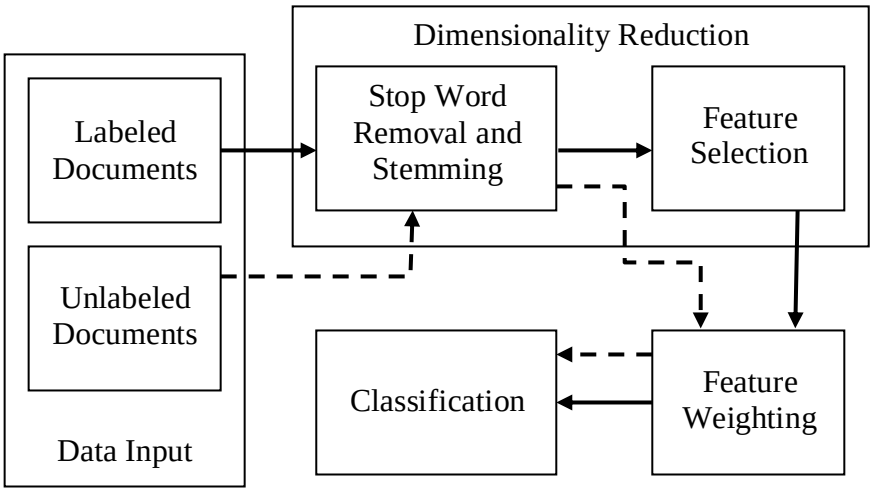

Figure 4.1 the proposed system architecture for Arabic text categorization.

4.3 Dataset Description

We have collected our own text collection. This collection consists of 1,132 documents that contain 95138 words (22347 unique words). These documents were collected from the three main Egyptian newspapers ElAhram, ElAkhbar, and ElGomhoria during the period from August 1998 to September 2004. These documents cover 6 topics. Table 4.1 shows the number of documents for each topic. Each document has average size of about 84 words before stemming and stop words removal. Document represents the first paragraph of an article, it has been chosen because it usually contains an abstract to the whole article.

Table 4.1 Number of documents for each topic in the text collection.

Topic	No. of documents
Arts	233
Economics	233
Politics	280
Sports	231
Woman	121
Information Technology	102

4.4 Stemming

For stemming, we introduce a comparative study for the Root-Based stemmer, light stemmer and statistical stemmer to decide which approach is suitable for Arabic text categorization task.

4.4.1 Root-Based Stemming

A Root-Based stemming algorithm has been developed. This algorithm removes the most common suffixes and prefixes from the word then matches the word against a set of suggested 67 patterns represent most of word forms. Also the algorithm aims at removing insignificant words from the text, these unvalued words are the stop words, foreign words, and digits. The used Root-Based stemming algorithm is shown in figure 4.2.

```
For every word in the text
1. IF the word is not an Arabic word THEN consider this
   word as a useless word.
2. IF the word contains digits THEN consider this word
   as a useless word.
3. IF the word length < 3 characters THEN consider this
   word as a useless word.
4. Remove diacritics.
5. Normalize the word.
6. IF the word is a stop word THEN consider this word
   as a useless word.
7. Remove prefixes.
8. Recursively remove suffixes.
9. IF the word is a stop word THEN consider this word
   as a useless word.
10.  Match word against 67 patterns and extract the
   root.
```

Figure 4.2 Developed root-based stemmer.

First the algorithm makes sure that the word is an Arabic word, also it considers any word contains less than 3 letters as an article and thus a not important word. Then it removes diacritics (ً, ٌ, ٍ, َ, ُ, ِ, ّ, ْ), which are marks above or below letters used in orthography and as a sign for the word grammatical case. After that it begins by normalizing the word as presented by many authors [Leah Larkey el al, 2001] [Mohammed Al-Jlayl et al, 2002] [Aitao Chen et al, 2002] [Leah Larkey et al, 2002]. Normalization is the process of unification of different forms of the same letter as follows.

- Normalize أ, إ, آ to ا.
- Normalize ـة to ـه.
- Normalize ى to ي.
- Normalize the sequence ء ي and the sequence ء ى to ئ.

After word normalization the algorithm checks if the word is one of the stop words list. The stop words list consists of 130 words taken from

Khoja list [Shereen Khoja, 1999] after being normalized as shown above beside some words have been added. A full list of these words is presented in Appendix A.

After stop word elimination, the algorithm removes a set of prefixes (و , لل , ال , كال , فال , بال) and the letter (ل) if the word starts with the sequence (لا) , after removing these prefixes it checks if the word length is less than 3 letters, in this case this prefix is considered as a main part of the word and so the removed prefix is returned back to the word.

In step 8 the suffixes (ها , هن , هم , ين , ون , ان , تي , ته , يه , ات , كما , هما , نا , وا , كم , كن , ي , ـه) are recursively removed from the tail of the word. The longest suffix is removed first, then the shorter. This process is recursive because most suffixes are compound of pronouns, gender and number suffixes, for example the word (مكتباتهم) (Their libraries) has a composite suffix (اتهم) which is made from two parts (ات) for feminine plural and the pronoun (هم). Also as done in the previous step the algorithm checks if the word length is greater than 3 letters in order not to remove a main part of the word.

After prefixes and suffixes removal the word is checked against the stop words list again because some stop words may have some prefixes and suffixes attached to them. Finally the word is checked against a set of 67 patterns to extract the root. A full list of these patterns is presented in Appendix B.

4.4.2 Light Stemming

A light stemming algorithm is developed. It removes the most common suffixes and prefixes and keeps the form of the word without

changing. The same steps used for Root-Based stemming algorithm are used with the light stemmer except the step number 10 in which the root is extracted.

4.4.3 Statistical Stemming

Finally, statistical n-gram stemmer is implemented. The following example shows how digram similarity between the word (سياسة) (Politic) and the word (سياسيا) (Political) is measured.

- سياسة $\Rightarrow$ سة ، اس ، يا ، سي . (We divided the word into set of digrams each of two adjacent letters)
- Unique digrams $\Rightarrow$ سة ، اس ، يا ، سي .

- سياسيا $\Rightarrow$ يا ، سي ، اس ، يا ، سي .
- Unique digrams $\Rightarrow$ اس ، يا ، سي .

$$\text{Similarity} = \frac{2C}{A+B} = \frac{2*3}{4+3} = 0.8571.$$

Where A and B are the numbers of unique digrams in the first and the second words. C is the number of unique digrams shared by A and B.

Similarity measures are determined for all pairs of terms in the corpus. Terms that have a similarity above a predefined threshold (α) are clustered and represented with only one term. Figure 4.3 shows the algorithm used to cluster words using N-Gram stemmer.

```
For each word w_k in the collection do
      if w_k has no cluster label then
            For each cluster c_i of words do
                  For each word w_j in c_i do
                        Measure the similarity between w_k and w_j
                        if similarity < threshold then
                              Assign w_k to c_i
      if w_k has no cluster label then
            Assign w_k to a new empty cluster
```

Figure 4.3 Clustering algorithm using N-Gram stemmer.

4.4.4 Experimental Results

Here we provide some details of how the runs, that produced the results we are about to discuss, were performed. For these experiments we used Document Frequency Thresholding criteria as a default term selection criteria. The Boolean weighting is used for representing the documents. The Rocchio classifier with β=1.6 and γ=0.4 [William Cohen et al, 1999] is used for classifications. The macroaveraged *F1* is used as an evaluation criterion. The macroaveraged *F1* measure is recorded for different number of terms ranged from 2500 to 5000 term selected according to Document Frequency Thresholding criteria . All tests are performed on the training documents.

For statistical n-gram stemming approaches different values for N are used, N=2 (digram) and N=3 (trigram). Also different similarities threshold values are used. Words with *n*-gram similarity above that threshold are assumed to be similar and have the same impact on the documents. An improvement has been performed in statistical stemmer by applying light stemmer before performing similarity measure in order to maximize the performance of the statistical stemmer.

A comparison of the categorization performance of the different approaches is shown in figure 4.4 and table 4.2. The highest classification rate is obtained by the hybrid approach of light and trigram stemming with $\alpha = 0.8$. Figure shows that all stemmers significantly perform better than no stemming and N-gram stemming with $\alpha = 0.9$. This comes from the fact that Arabic is a highly inflected language; thus, the stemming will group the huge variety of word forms into smaller conflation classes. On the other hand, root stemming gave intermediate accuracy. While light stemming, diagram stemmer with $\alpha < 0.9$, hybrid stemmer of light stemmer and *N*-gram stemmer with $\alpha < 0.9$ gave the best results, as it tries to group the words in some how, light stemming only removes some common prefixes and suffixes. However there are many other rare prefixes that aren't removed by light stemming like some prepositions that may be attached to the beginning of the word. This leads to think about using *N*-gram stemming after applying light stemmer. *N*-gram stemming has the ability to discover the similarity between words even if they are attached to any affixes. Appendix C shows samples for groups of words clustered using the suggested hybrid approach of light and trigram stemming with $\alpha = 0.8$.

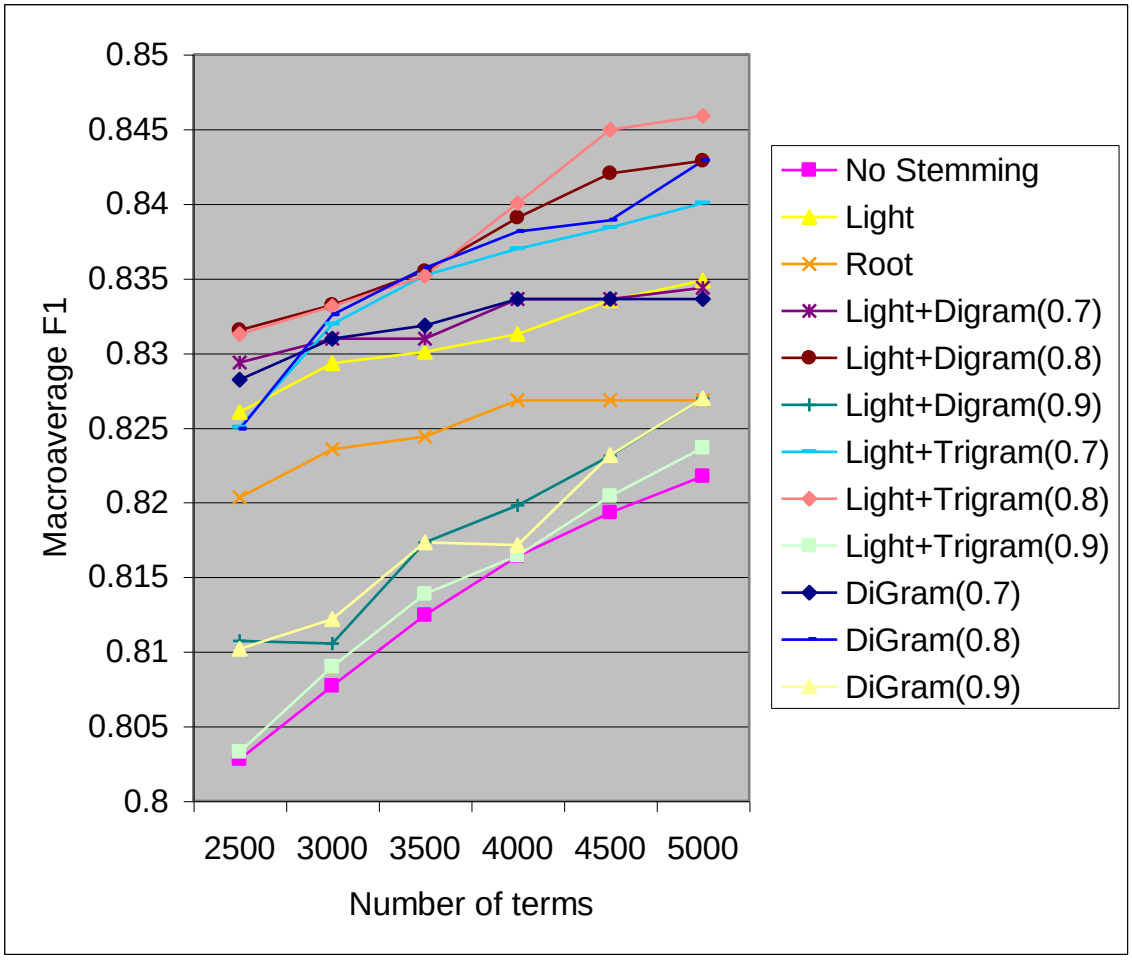

Figure 4.4 Effect of stemming in categorization accuracy.

Table 4.2 Effect of stemming in categorization accuracy.

Number of Terms	No Stemming	Root	Light	DiGram			Light + Digram			Light + Trigram		
				α = 0.7	α = 0.8	α = 0.9	α = 0.7	α = 0.8	α = 0.9	α = 0.7	α = 0.8	α = 0.9
2500	0.80278	0.82033	0.82602	0.8282	0.82491	0.8102	0.82937	**0.83152**	0.81073	0.82503	0.83125	0.80329
3000	0.80772	0.82354	0.82931	0.83097	0.83257	0.81216	0.83097	**0.83322**	0.81053	0.83193	0.83311	0.809
3500	0.81246	0.82439	0.83009	0.83184	**0.83568**	0.8173	0.83097	0.83548	0.8173	0.83523	0.83517	0.81386
4000	0.8164	0.82685	0.83127	0.8336	0.83815	0.81714	0.8336	0.83906	0.81978	0.83698	**0.84003**	0.81644
4500	0.81929	0.82685	0.83352	0.8336	0.83887	0.82318	0.8336	0.84205	0.8231	0.8384	**0.84495**	0.82042
5000	0.82174	0.82685	0.83484	0.8336	0.84289	0.82696	0.83436	0.84289	0.82696	0.84005	**0.84587**	0.82363
Average	0.813398	0.82480	0.830842	0.831968	0.835511	0.817823	0.832145	0.83737	0.818067	0.834603	**0.838397**	0.81444

4.5 Term Selection

4.5.1 Term Selection Criteria

After stemming phase, the dictionary of words is formed with thousands of terms, so that it is a must to find the most valuable set of terms by term selection.

Six methods are included in this study, each of which uses a term-goodness criterion thresholded to achieve a desired degree of term elimination from the full vocabulary of a document corpus. These criteria are: Document Frequency (DF), Information Gain (IG), Odds Ration (OR), x2 statistic (CHI), GSS coefficient (GSS) and NGL coefficient (NGL). Each of the five feature selection methods was evaluated with a number of different term-removal thresholds.

Figure 4.5 and table 4.3 displays the performance curves for Rocchio classifier after term selection using DF, CHI, GSS and NGL thresholding. An observation merges from the categorization that DF, GSS, NGL and CHI thresholding have similar effects on the performance of the classifiers. Also it is noticed that when using IG and OR, most of the documents didn't contain any term in the list of the selected terms. In other words IG and OR select terms with rare appearance in the data set (i.e. terms with very low document frequency). This problem affects the accuracy as we considered those sparse documents as misclassified. To avoid this problem we used a hybrid approach between Document Frequency Threshold and the other criteria. Document Frequency is used to remove rare terms and the other criteria to select terms from the remaining list.

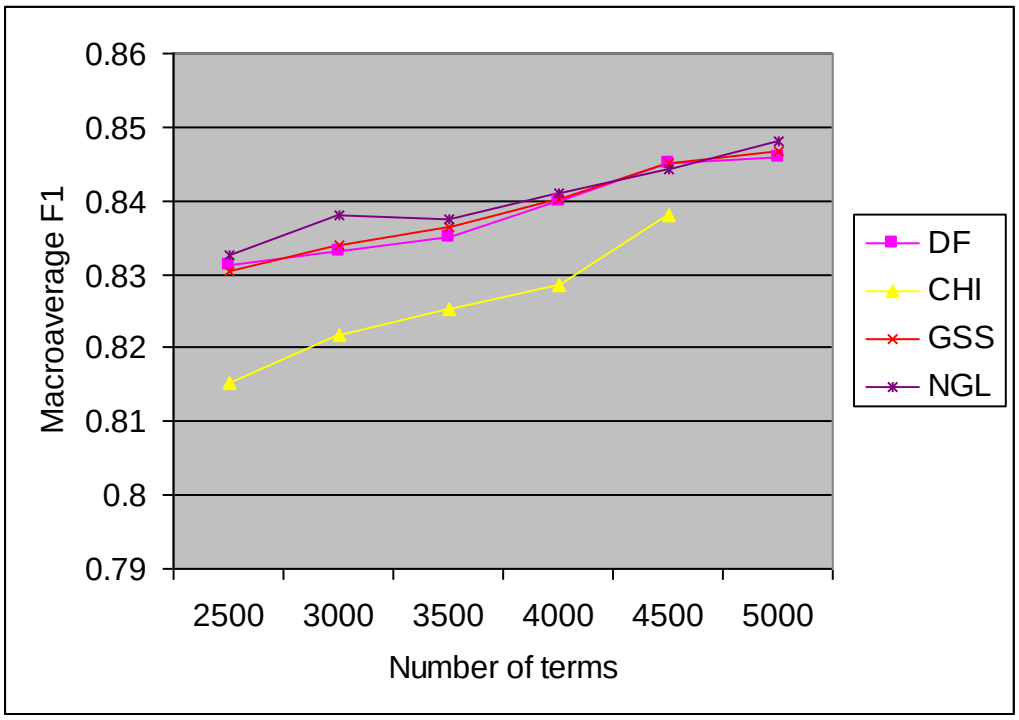

Figure 4.5 Effect of term selection criteria in categorization accuracy.

Table 4.3 Effect of term selection criteria in categorization accuracy.

Number of Terms	DF	CHI	GSS	NGL
2500	0.83125	0.81513	0.83032	**0.83254**
3000	0.83311	0.82169	0.83393	**0.83807**
3500	0.83517	0.82533	0.83633	**0.83735**
4000	0.84003	0.82865	0.84011	**0.84105**
4500	0.84495	0.83793	**0.84506**	0.84426
5000	0.84587	0.84188	0.84659	**0.84815**
Average	0.8384	0.82844	0.83872	**0.84024**

The proposed hybrid approach gave the better results than the other methods. It was noticed that the only 4100 terms remains after using Document Frequency Threshold to remove terms with document frequency less than 3. The overall accuracy was increased as the number of sparse documents decreased. Although number of empty documents is reduced, there still some few sparse documents espicially when representing documents with low number of terms. We tried to overcome

this draw back by applying local feature selection instead of global feature selection. Figure 4.6 and table 4.4 show the accuracy of classification process for different hybrid term selection approach.

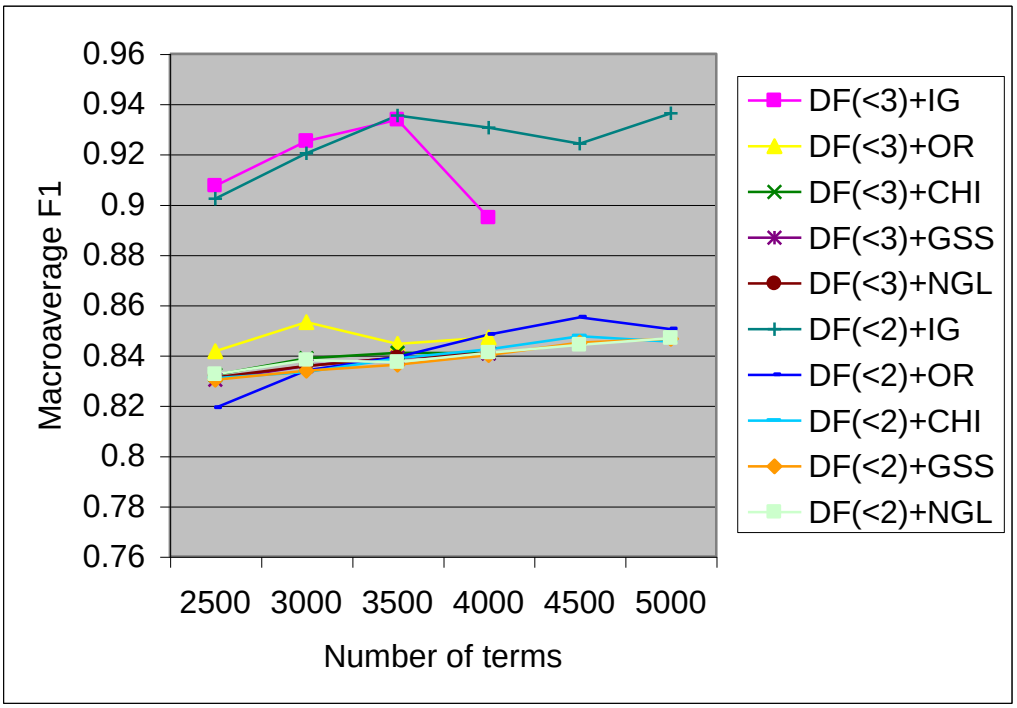

Figure 4.6 Effect of hybrid term selection criteria on categorization accuracy.

Table 4.4 Effect of hybrid term selection criteria on categorization accuracy.

Number of Terms	Hybrid feature selection criteria of (DF<3) and					Hybrid feature selection criteria of (DF<2) and				
	IG	OR	CHI	GSS	NGL	IG	OR	CHI	GSS	NGL
2500	**0.9074**	0.84172	0.83232	0.83032	0.83119	0.90231	0.81917	0.83094	0.83032	0.83254
3000	**0.92526**	0.85332	0.83882	0.83585	0.83586	0.92048	0.834	0.83345	0.83393	0.83807
3500	0.93373	0.84456	0.84108	0.83914	0.83914	**0.93547**	0.83944	0.83889	0.83633	0.83735
4000	0.89474	0.8472	0.84086	0.84075	0.84075	**0.93067**	0.84833	0.84242	0.84011	0.84105
4500						**0.92429**	0.85518	0.84762	0.84506	0.84426
5000						**0.93639**	0.85045	0.8455	0.8467	0.84684
Average	0.9152825	0.8467	0.83827	0.836515	0.836735	**0.924935**	0.841095	0.83980333	0.838741667	0.84001833

4.5.2 Term Selection Approaches

The results of the global feature selection used in the previous section showed the problem of generating empty documents.

To overcome this problem we used the local feature selection approach. In this case, the experimental results shows that the number of empty documents is reduced but the classification rate was less than that of global feature selection. Thus, we will use a hybrid approach to enhance the classification rate. Figure 4.7 shows the proposed hybrid feature selection approach.

```
Step0: Perform preprocessing routines for the documents. This
routines includes stop word removal and stemming.

Step1: Select a global set of features for all classes.

Step2: For each class ci Select a local set of features.

Step3: Represent each document di of the training documents as a
weighted vector of the global feature set and as a weighted vector
for each of the m classes' local feature set.

Step 4: In classification phase:
      For each document di
            If(di is not empty according to the global vector)
            Then Classify di using its global vector
            Else Classify di using its local vectors
```

Figure 4.7 Hybrid feature selection approach.

The proposed algorithm combines the two feature selection approaches. It selects set of global features and sets of local features for each class and represents the documents with the two representations. In classification phase, we will use the local vector representation if the document was empty, otherwise the global vector representation will be used.

4.5.3 Experimental Results

Figure 4.8, 4.9 and tables 4.5, 4.6 shows that the Macroaverage F1 measure of the different feature selection techniques.Also, shows that the hybrid approach gives high classification rate at different threshold.

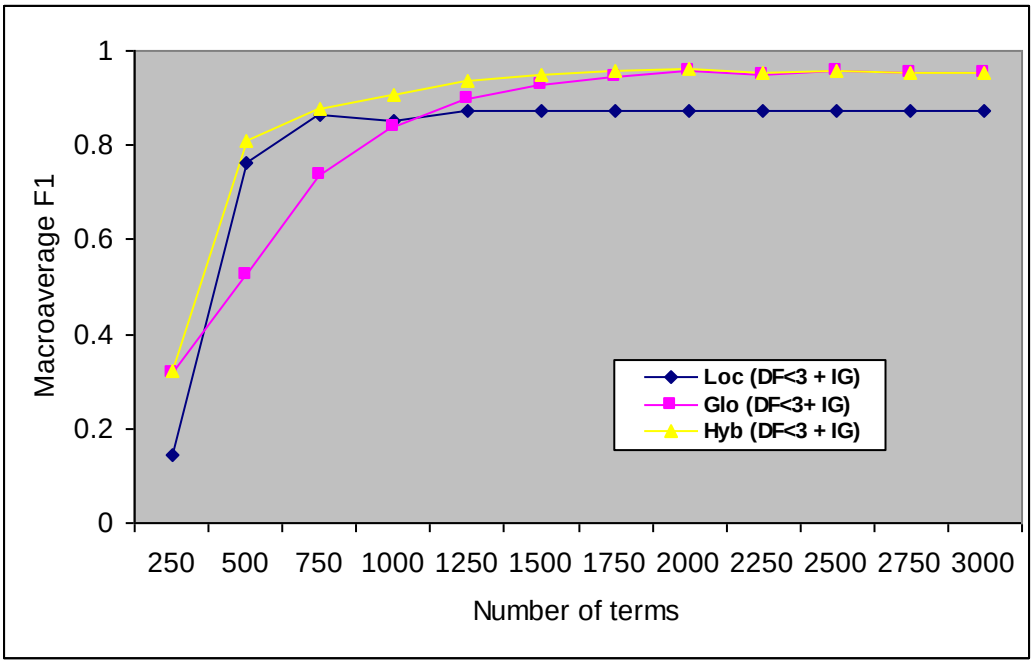

Figure 4.8 Performance of feature selection approaches using document frequency threshold=2 combined with information gain.

Table 4.5 Performance of feature selection approaches using document frequency threshold=2 combined with information gain.

Number of Terms	Loc (DF<3 + IG)	Glo (DF<3+ IG)	Hyb (DF<3 + IG)
250	0.14504	0.31937	**0.32394**
500	0.76113	0.52563	**0.80989**
750	0.86495	0.7372	**0.87656**
1000	0.87116	0.83831	**0.90782**
1250	0.87116	0.89949	**0.93816**
1500	0.87116	0.92888	**0.94954**
1750	0.87116	0.94579	**0.95833**
2000	0.87116	0.9563	**0.96076**
2250	0.87116	0.95014	**0.95159**
2500	0.87116	**0.95882**	**0.95882**
2750	0.87116	**0.95363**	**0.95363**
3000	0.87116	**0.95496**	**0.95496**
Average	0.7994	0.83071	**0.878667**

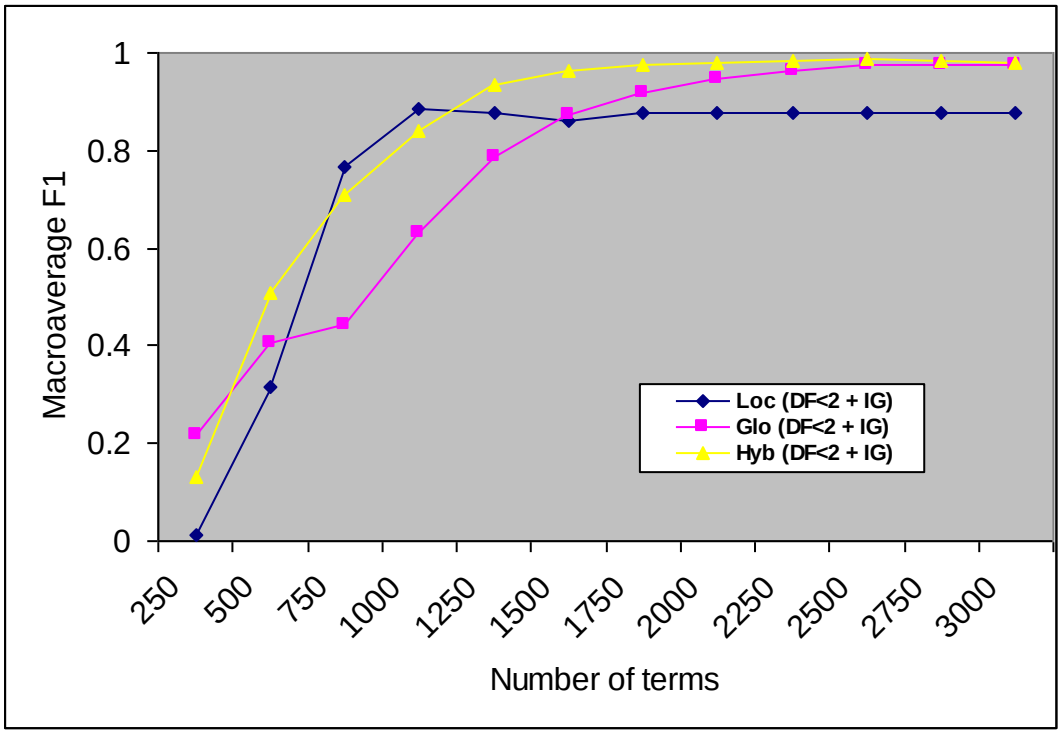

Figure 4.9 Performance of feature selection approaches using document frequency threshold=1 combined with information gain.

Table 4.6 Performance of feature selection approaches using document frequency threshold=1 combined with information gain.

Number of Terms	Loc (DF<2 + IG)	Glo (DF<2+ IG)	Hyb (DF<2 + IG)
250	0.01056	**0.21892**	0.13149
500	0.31405	0.40707	**0.50629**
750	**0.76449**	0.44378	0.71058
1000	**0.88648**	0.62961	0.84175
1250	0.87538	0.78874	**0.93387**
1500	0.86163	0.87325	**0.96235**
1750	0.8754	0.91834	**0.97383**
2000	0.8754	0.94607	**0.9786**
2250	0.8754	0.96451	**0.98296**
2500	0.8754	0.97631	**0.98728**
2750	0.8754	0.9771	**0.98323**
3000	0.8754	0.97457	**0.97746**
Average	0.7470825	0.7598558	**0.8308075**

Figure 4.10, 4.11 and tables 4.7, 4.8 shows the number of empty documents for the different feature selection techniques. Figures show that the number of empty documents for the hyprid and local approachs has the same numbers approximatly.

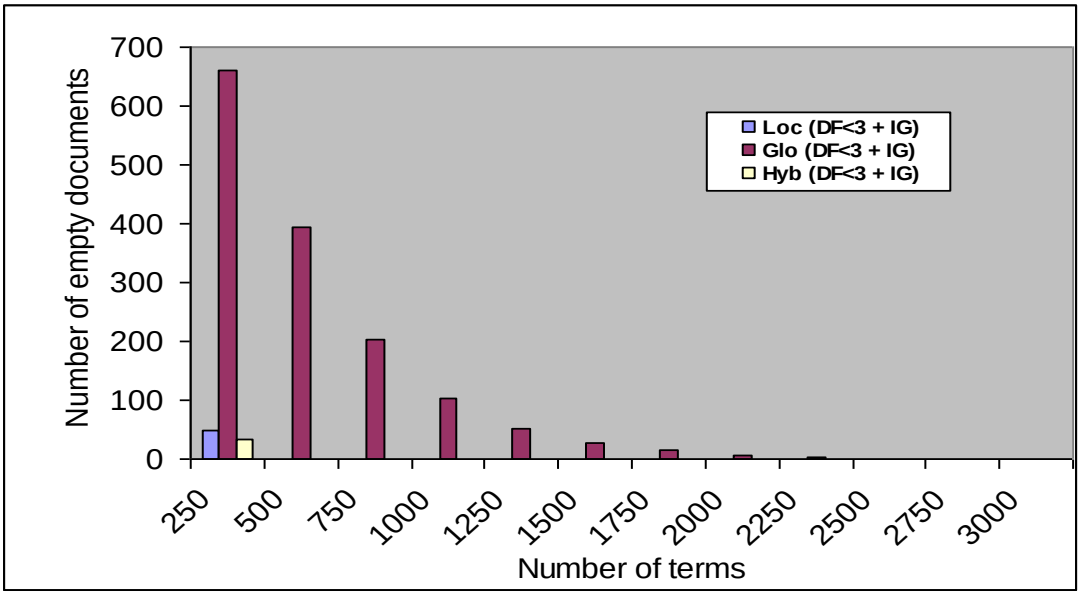

Figure 4.10 Number of empty documents using Document frequency threshold=2 combined with information gain.

Table 4.7 Number of empty documents using Document frequency threshold=2 combined with information gain.

Number of Terms	Loc (DF<3 + IG)	Glo (DF<3+ IG)	Hyb (DF<3 + IG)
250	50	661	**32**
500	1	395	**0**
750	**0**	202	**0**
1000	**0**	103	**0**
1250	**0**	52	**0**
1500	**0**	28	**0**
1750	**0**	16	**0**
2000	**0**	2	**0**
2250	**0**	**0**	**0**
2500	**0**	**0**	**0**
2750	**0**	**0**	**0**
3000	**0**	**0**	**0**

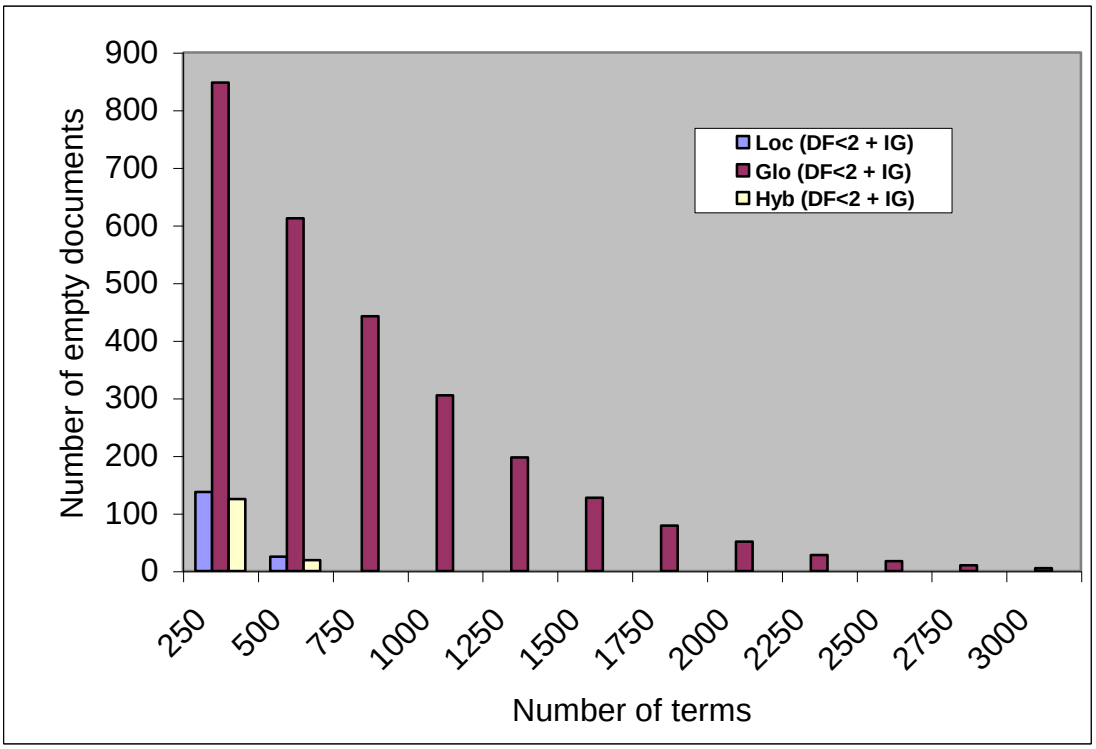

Figure 4.11 Number of empty documents using Document frequency threshold=1 combined with information gain.

Table 4.8 Number of empty documents using Document frequency threshold=1 combined with information gain.

Number of Terms	Loc (DF<2 + IG)	Glo (DF<2+ IG)	Hyb (DF<2 + IG)
250	137	848	**125**
500	25	612	**19**
750	**0**	442	**0**
1000	**0**	305	**0**
1250	**0**	197	**0**
1500	**0**	127	**0**
1750	**0**	79	**0**
2000	**0**	51	**0**
2250	**0**	28	**0**
2500	**0**	17	**0**
2750	**0**	10	**0**
3000	**0**	5	**0**

Finally, we conclude that the hyprid approach reduces the number of empty documents as local approach and gives a high classification rate like the global one.

4.6 Term Weighting

The most commonly used document representation is so called vector space model. In this model documents are represented by vector of terms. There are several ways to determine the weight of a term in a document. The *tf*, *tfidf*, Boolean, and normalized-*tfidf* methods are examined. Results presented in figure 4.12 and table 4.9 shows that normalized-*tfidf* is the preferable method for term weighting.

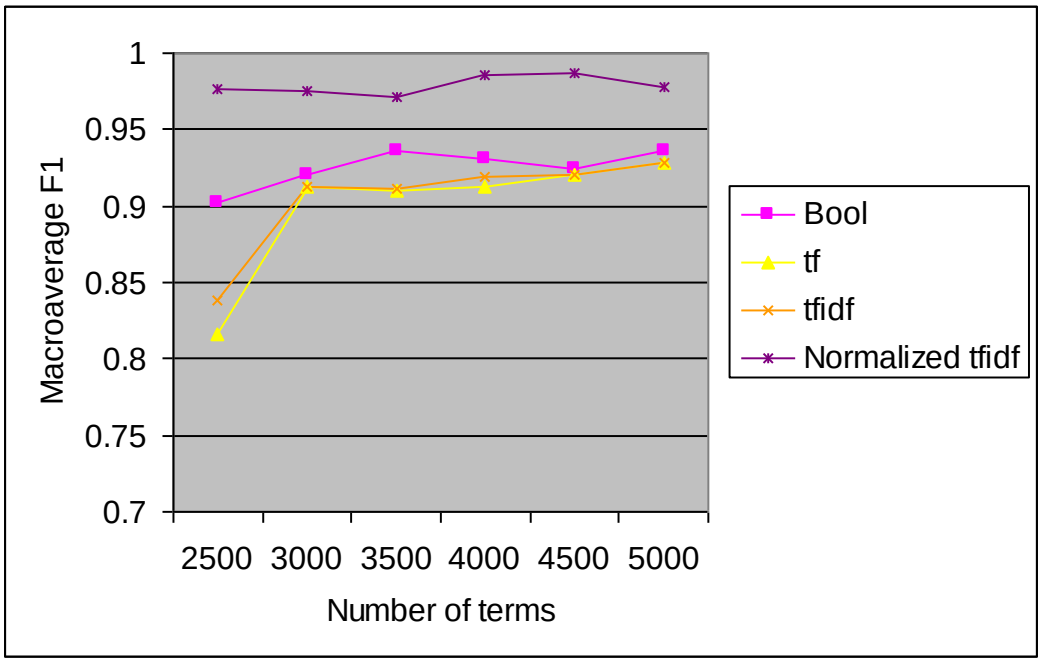

Figure 4.12 Effect of term weighting method in categorization accuracy.

Table 4.9 Effect of term weighting method in categorization accuracy.

Number of Terms	Bool	Tf	tfidf	Normalized tfidf
2500	0.90231	0.81632	0.83882	**0.97631**
3000	0.92048	0.9129	0.9122	**0.97457**
3500	0.93547	0.91045	0.91185	**0.97086**
4000	0.93067	0.91299	0.91893	**0.98625**
4500	0.92429	0.92	0.91983	**0.98699**
5000	0.93639	0.9278	0.92839	**0.97779**
Average	0.924935	0.900077	0.905003	**0.978795**

4.7 Classification

Four classifiers widely used with text categorization tasks were examined; Multinomial Naïve Bayes classifier, *k*-NN classifier, Rocchio classifier, and Support Vector Machines (SVM) (see section 3.6).

4.7.1 Experimental Results

Experiments were performed using different sizes of features ranges from 2500 to 5000 feature. For each number of features different values of β and γ for the Rocchio algorithm are used. The K in the K-NN classifier was set to the values {1, 3, 5, 7, …, 19}. The results for the parameters with the best performance on the test set are reported. The SVM implementation used was the Sequential Minimal Optimization (SMO) algorithm by Platt [John Platt, 1998] with a polynomial kernel function. The Naïve Bayes model used was the multinomial model.

Two experiments used to test the different classifiers. In the first experiment we used the training set of documents as a test set. *Leave One* method, in the second experiment, was used to test the classifiers. All experiments are performed on a computer with 2.8 GHz Pentium4 processor and 512 MB Ram.

When using the training documents as a test set, the results of SVM, Naïve Bayes and Rocchio classifiers were very high and the classification accuracy tend to be 100%, while using the *Leave One* method for testing gives more realistic results.

Table 4.10 shows the training time for the different classifiers (h:mm:ss). The Bayes classifier was the most efficient classifier while SVM was the most expensive.

Table 4.10 Time taken for training the classifiers

No. of terms	K-NN	Rocchio	Bayes	SVM
2500	0:00:03	0:00:16	**0:00:0.42**	0:03:45
3000	0:00:04	0:00:20	**0:00:0.49**	0:04:42
3500	0:00:05	0:00:22	**0:00:0.56**	0:05:27
4000	0:00:08	0:00:28	**0:00:0.63**	0:05:59
4500	0:00:13	0:00:30	**0:00:0.69**	0:06:23
5000	0:00:17	0:00:32	**0:00:0.88**	0:06:52

Figure 4.13, 4.14 and table 4.11, 4.12 shows that Rocchio classifier was competitive to the widely used classifier SVM. However the SVM classifier outperforms the other classifiers when the number of features is high. The same conclusion was presented by Joachims [Thorsten Joachims, 1998]. Also experimental results show that the best values for β and γ of the Rocchio classifier were 1.6 and 0.4 respectively.

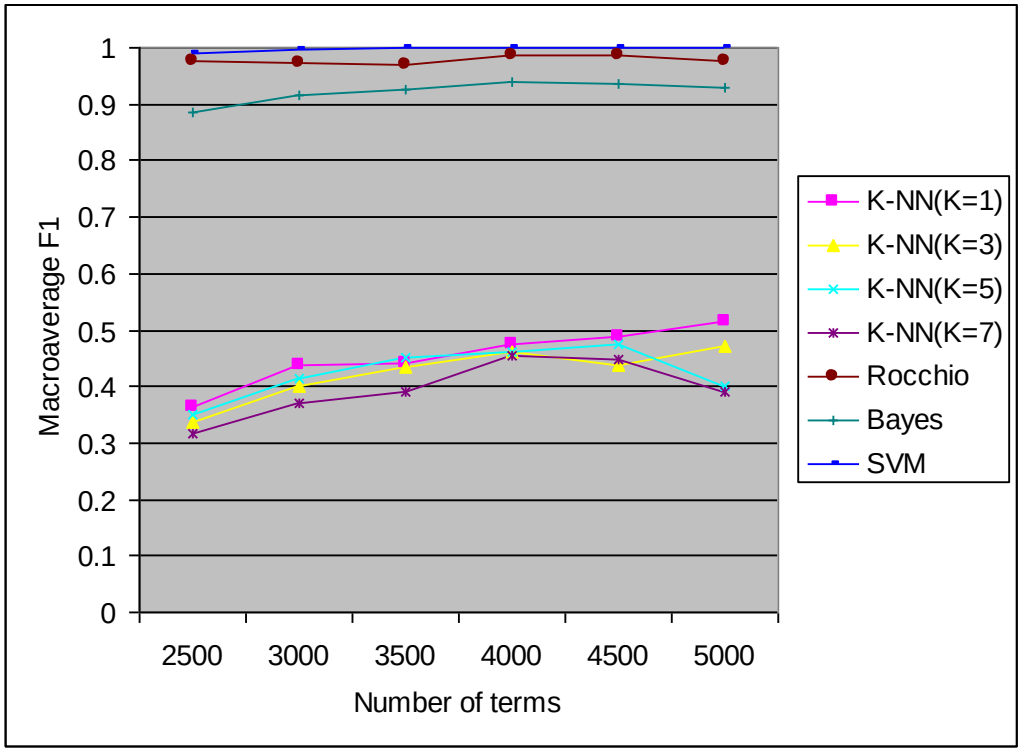

Figure 4.13 Classifiers performance using the training documents.

Table 4.11 Classifiers performance using the training documents.

No. of terms	*K*-NN			Rocchio	Bayes	SVM
	K=1	*K*=3	*K*=5			
2500	0.36246	0.33678	0.34871	0.97631	0.8855	**0.99**
3000	0.43637	0.40214	0.41303	0.97457	0.916333	**0.996333**
3500	0.43973	0.43402	0.44998	0.97086	0.9265	**1**
4000	0.47345	0.46204	0.46035	0.98625	0.939167	**1**
4500	0.48741	0.43776	0.47325	0.98699	0.936667	**1**
5000	0.51547	0.46975	0.40084	0.97779	0.929333	**1**
Average	0.45248167	0.4237483	0.42436	0.978795	0.92225	**0.997722**

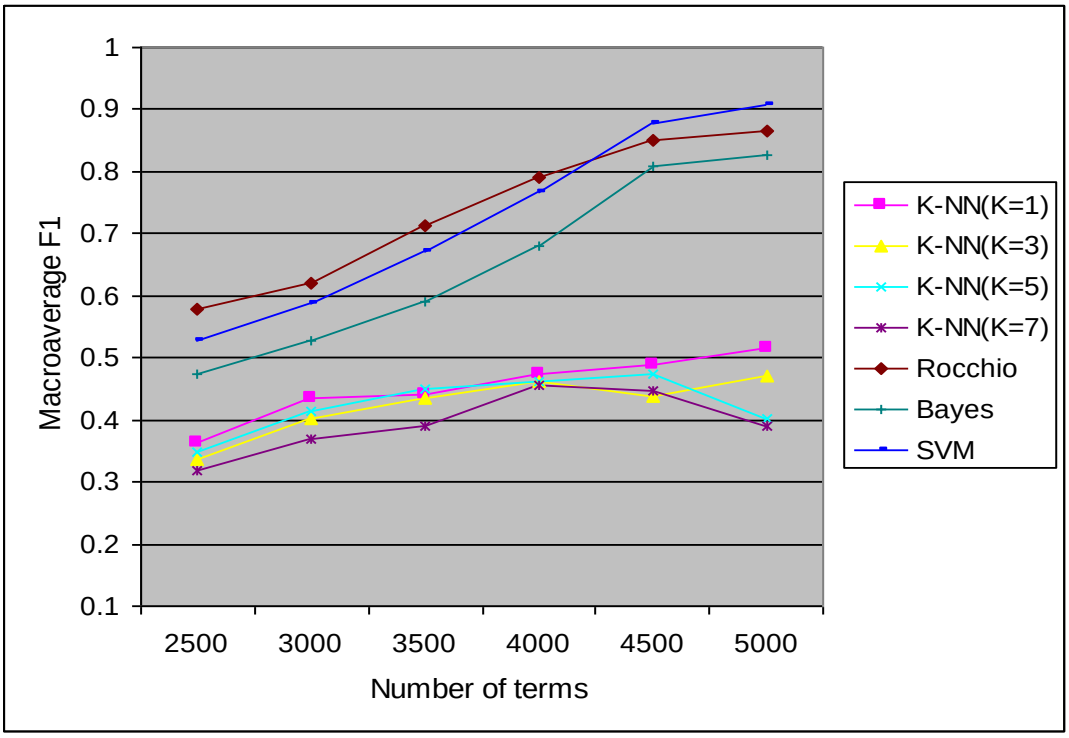

Figure 4.14 Classifiers performance using *Leave One* testing.

Table 4.12 Classifiers performance using *Leave One* testing.

No. of terms	*K*-NN			Rocchio	Bayes	SVM
	K=1	*K*=3	*K*=5			
2500	0.36246	0.33678	0.34871	**0.57986**	0.473333	0.527863
3000	0.43637	0.40214	0.41303	**0.62017**	0.527667	0.58765
3500	0.43973	0.43402	0.44998	**0.71242**	0.590167	0.671794
4000	0.47345	0.46204	0.46035	**0.78996**	0.679167	0.767182
4500	0.48741	0.43776	0.47325	0.84982	0.809333	**0.87858**
5000	0.51547	0.46975	0.40084	0.86558	0.825167	**0.908729**
Average	0.45248167	0.4237483	0.42436	**0.736302**	0.650806	0.723633

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This Thesis presents a system for Arabic text categorization. Many literatures discussed text categorization systems for other languages but few researches presented systems for Arabic language.

The research presented in this thesis focuses on the different phases of text categorization system, studying the strength and weakness of the existing approaches, and introducing new methods to improve performance.

Three stemming approaches are tested. Results indicate that hybrid approach of light and statistical stemmer is the most suitable for text categorization task in Arabic language. Due to the very high dimensionality of terms, several methods for selecting highly informative terms are used. Document frequency is used to remove rare terms and information gain to select most informative terms from the remaining list.

Two different feature selection approaches are also tested, the global approach and the local approach. The two approaches are combined to gain their advantages and discard their disadvantages. This hybrid approach gives a high classification rate comparatively with global feature selection and reduces the number of empty documents at the same time.

After term selection, every document is represented as vector of terms' weights. Four term weighting criteria are used; normalized-tfidf is the suggested weighting method. Finally, four classifiers are used; the Bayes classifier, K-NN classifier, Rocchio classifier and the Support Vector Machines (SVM) classifier. The Rocchio and the SVM classifiers significantly outperform the Naïve Bayes classifier. The SVM classifier outperforms the Rocchio classifier in high dimension feature space.

The experimental results show that the performance of the categorization system is related to many factors, such as preprocessing, feature selection methods, document representation, and categorization methods.

5.2 Future Work

In future, some related points may be covered:

1. Investigating the effect of using phrases instead of words as features of documents.
2. Study hierarchical categorization problem in which categories are organized as a hierarchy.
3. Applying classifier committees that are based on the idea that *k* different classifiers may be better than one if their individual judgments are appropriately combined.
4. Building a search engine or spam filtering application based on the proposed categorization system.

References

[Abdelwadood Moh'd A MESLEH, 2007] Abdelwadood Moh'd A MESLEH. "*Chi Square Feature Extraction Based Svms Arabic Language Text Categorization System*". Journal of Computer Science, Vol. 3, No. 6, Pages 430-435. 2007.

[**Ahmed Abdelali et al, 2004**] Ahmed Abdelali, jim Cowie, and Hamdy Soliman, "*Arabic Information Retrieval Perspectives*", Proceedings of JEP-TALN 2004 Arabic Language Processing, Fez 19-22 April 2004.

[Aitao Chen et al, 2002] Aitao Chen, and Fredric Gey. "*Building an Arabic stemmer for information retrieval*". In TREC 2002. Gaithersburg: NIST. Pages 631-639. 2002.

[Alaa El-Halees, 2007] Alaa El-Halees. "*Arabic Text Classification Using Maximum Entropy*". The Islamic University Journal, Vol. 15, No. 1, Pages 157-167. 2007.

[**Andrew Mccallum et al, 1998**] Andrew Mccallum, Kamal Nigam. "*A comparison of event models for naive bayes text classification*". In AAAI-98 Workshop on Learning for Text Categorization. Pages 41-48. 1998.

[**Anne De Roeck et al, 2000**] Anne De Roeck and Waleed Al-Fares. "*A Morphologically Sensitive Clustering Algorithm for Identifying Arabic Roots*". In Proceedings of the 38 th Annual Meeting of the Association for Computational Linguistics (ACL), Hong Kong. Pages 199 - 206, 2000.

[**Bong Chih How et al, 2004**] Bong Chih How, and Kulathuramaiyer Narayanan. "*An Empirical Study of Feature Selection for Text Categorization based on Term Weightage*". In proceeding of the IEEE/WIC/ACM International Conference on Web Intelligence WI'04. Pages 599-602. 2004.

[**Bong Chih How et al, 2005**] Bong Chih How, and Wong Ting Kiong, "*An Examination of Feature Selection Frameworks in Text Categorization*". Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Vol 3689 / 2005. Pages 558 – 564. 2005.

[**Chris Paice, 1996**] Chris Paice, *"**Method for evaluation of stemming algorithms based on error counting***". Journal of the American Society for Information Science, Vol. 47, No. 8, Pages 632 – 649. 1996.

[**Colin White, 2005**] Colin White. "*Consolidating, Accessing and Analyzing Unstructured Data*". Online Article. http://www.b-eye-network.com/view/2098. 2005

[**Daphne Koller et al, 1996**] Daphne Koller and Mehran Sahami. "*Toward Optimal Feature Selection*". In proceedings of the 13th International Conference on Machine Learning. Italy. Pages 284-292. 1996.

[**Dunja Mladenic et al, 1998**] Dunja Mladenic and Marko Grobelnik. "*Feature selection for classification based on text hierarchy*". Conference on Automated Learning and Discovery CONALD-98. 1998.

[**Dunja Mladenic, 1998**] Dunja Mladenic. "*Feature subset selection in text-learning*". In 10th European Conference on Machine Learning ECML98. Pages 95-100. 1998.

[**Evgeniy Gabrilovich, 2007**] http:// liinwww.ira.uka.de / bibliography / Ai / automated.text.categorization.html

[**Fabrizio Sebastiani, 1999**] Fabrizio Sebastiani. "*A tutorial on automated text categorisation*". In Analia Amandi and Ricardo Zunino, editors, Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence, Pages 7-35. 1999.

[**Fabrizio Sebastiani, 2002**] Fabrizio Sebastiani. "*Machine learning in automated text categorization*". ACM Computing Surveys, Nol 34 No 1. Pages 1-47. 2002.

[**George John et al, 1994**] George John, Ron Kohavi and Karl Pfleger. "*Irrelevant features and the subset selection problem*". In proceedings of the 11th International Conference on Machine Learning. Pages 121–129. 1994

[**Gerard Salton et al, 1975**] Gerard Salton, Changsheng Yang, and Benjamin Wong. "*A vector-space model for automatic indexing*". Communications of the ACM, Vol 18 No 11. Pages 613–620. 1975.

[Hassan Sawaf et al, 2001] Hassan Sawaf, Jörg Zaplo, and Hermann Ney. "*Statistical Classification Methods for Arabic News Articles*". In Proceedings of the ACL/EACL 2001 Workshop on Arabic Language Processing: Status and Prospects. Toulouse, France. 2001.

[**Hegazi et al, 1985**] Hegazi, N.H., & El-Sharkawi, A.A. " *An approach to a computerized lexical analyzer for natural Arabic text*". In Proceedings of the Arabic Language Conference, Kuwait, 1985.

[**Hwee Tou Ng et al, 1997**] Hwee Tou Ng, Wei Boon Goh, and Kok Leong Low. "*Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization*". In Proceedings the 20th ACM International Conference on Research and Development in Information Retrieval SIGIR-97. Philadelphia, PA. Pages 67–73. 1997.

[**J. Rocchio, 1971**] J. Rocchio, "*Relevance feedback in information retrieval*". In G.Salton (Ed.), The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice-Hall. Pages 313-323. 1971.

[**James Mayfield et al, 2001**] James Mayfield, Paul McNamee, Cash Costello, Christine Piatko, and Amit Banerjee. "*A. JHU/APL at TREC 2001: Experiments in filtering and in Arabic, video, and web retrieval*". In TREC 2001. Gaithersburg: NIST, 2001.

[**Jinxi Xu et al, 1998**] Jinxi Xu and Bruce Croft. "*Corpus-Based Stemming using Cooccurrence of Word Variants*". In journal of ACM Transactions on Information Systems, Vol 16 No 1. Pages 61-81. 1998.

[**Johannes Fuernkranz et al, 1998**] Johannes Fuernkranz, Tom Mitchell, and Ellen Riloff. "*A case study in using linguistic phrases for text categorization on the www*". Sahami, M., editor, In Learning for Text Categorization: Papers from the 1998 AAAI Workshop (Technical Report WS-98-05), 1998.

[**John Platt, 1998**] John Platt. "*Sequetial minimal optimization: A fast algorithm for training support vector machines*". In Technical Report MST-TR-98-14. Microsoft Research, 1998.

[**Julie Beth Lovins, 1968**] Julie Beth Lovins, "*Development of a Stemming Algorithm*". Mechanical Translation and Computational Linguistics, Vol. 11, No. 1&2, Pages 22--31, 1968.

[**Kareem Darwish, 2002**] Kareem Darwish. "*Building a shallow Arabic morphological analyzer in one day*". In Proceedings of the Association for Computational Linguistics (ACL-02), 40th Anniversary Meeting, University of Pennsylvania, Philadelphia. Pages 47–54. 2002.

[**Kazem Taghva et al, 2005**] Kazem Taghva, Rania Elkhoury, and Jeffrey Coombs. "*Arabic Stemming Without A Root Dictionary*". In proceedings of ITCC 2005 International Conference on Information Technology: Coding and Computing, Las Vegas, NV, USA, Pages 152-157, April 2005.

[**Kenneth Beesley, 2001**] Kenneth Beesley. "*Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans*". ACL/EACL01: Conference of the European Chapter, Workshop: Arabic Language Processing: Status and Prospects. 2001.

[**Kjersti Aas et al, 1999**] Kjersti Aas, and Line Eikvil. "*Text Categorisation: A Survey*". Technical report, Norwegian Computing Center. 1999.

[Laila Khreisat, 2006] Laila Khreisat. "*Arabic Text Classification Using N-Gram Frequency Statistics A Comparative Study*". In proceedings of the 2006 International Conference on Data Mining, DMIN 2006. Pages 78-82. 2006.

[**Leah Larkey el al, 2001**] Leah Larkey and Margaret Connell. "*Arabic information retrieval at UMass in TREC-10*". In Proceedings of the 10th Text Retrieval Conference (TREC2001). Gaithersburg, Maryland. Pages 562–570. 2001

[**Leah Larkey el al, 2005**] Leah Larkey, Lisa Ballesteros, and Margaret Connell. "*Light Stemming for Arabic Information Retrieval*". Chapter on Arabic Computational Morphology: Knowledge-based and Empirical Methods. Kluwer/Springer's series on Text, Speech, and Language Technology. 2006

[**Leah Larkey et al, 1996**] Leah Larkey and Bruce Croft. "*Combining classifiers in text categorization*". In Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval. Zurich, Switzerland. Pages 289–297.1996

[**Leah Larkey et al, 2002**] Leah Larkey, Lisa Ballesteros, and Margaret Connell. "*Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis*". In Proceedings of the 25th Annual International Conference on Research and Development in Information Retrieval (SIGIR 2002), Tampere, Finland.Pages 275-282. 2002.

[**Luigi Galavotti et al, 2000**] Luigi Galavotti, Fabrizio Sebastiani and Maria Simi. "*Experiments on the use of feature selection and negative evidence in automated text categorization*". In the proceedings the 4th European Conference on Research and Advanced Technology for Digital Libraries ECDL-00. Pages 59-68. 2000.

[**Martin Porter, 1980**] Martin Porter, "*An Algorithm for Suffix Stripping*", Program, Vol. 14, No. 3, Pages 130-137, 1980.

[**Mehran Sahami, 1998**] Mehran Sahami. "*Using Machine Learning To Improve Information Access*". Ph.D. thesis, Stanford University, 1998.

[Mohamed El Kourdi et al, 2004] Mohamed El Kourdi, Amine Bensaid, and Tajje-eddine Rachidi. "*Automatic Arabic Document Categorization Based on the Naive Bayes Algorithm*". In proceedings of the COLING-2004 Workshop on Computational Approaches to Arabic Script-Based Languages, Switzerland. Pages 51-58. 2004.

[**Mohammed Aljlayl et al, 2002**] Mohammed Aljlayl, Ophir Frieder, and David Grossman, "*On Arabic-English Cross-Language Information Retrieval: A Machine Translation Approach*", IEEE Third International Conference on Information Technology: Coding and Computing (ITCC), Las Vegas, Nevada, April 2002.

[**Mohammed Al-Jlayl et al, 2002**] Mohammed Al-Jlayl, and Ophir Frieder. "*On Arabic search: Improving the retrieval effectiveness via light stemming approach*". In Proceedings of the 11th ACM International Conference on Information and Knowledge Management, Illinois Institute of Technology. New York: ACM Press. Pages 340–347. 2002

[**Monica Rogati et al, 2003**] Monica Rogati, Scott McCarley and Yiming Yang. "*Unsupervised Learning of Arabic Stemming using a Parallel Corpus*". In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL), Sapporo, Japan. Pages 391 - 398. July 2003.

[**Murat Tayli and Abdulah Al-Salamah, 1990**] Murat Tayli, and Abdulah Al-Salamah, "*Building Bilingual Microcomputer Systems*", In Communications of the ACM, Vol. 33, No.5,1990, Pages 495-505, 1990.

[**Norbert Fuhr et al, 1991**] Norbert Fuhr and Chris Buckley. "*A probabilistic learning approach for document indexing*". ACM Transactions on Information Systems, Vol 9 No 3. Pages 223–248. 1991.

[**Rafael Calvo et al, 2000**] Rafael Calvo and HA Ceccatto. "*Intelligent document classification*". Intelligent Data Analysis. Vol 4 No 5. Pages 411–420. 2000.

[Reda El-Khoribi et al, 2006] Reda El-Khoribi and Mahmoud Ismael. "*An Intelligent System Based on Statistical Learning For Searching in Arabic Text*". In the ICGST International Journal on Artificial Intelligence and Machine Learning, AIML, Vol.6, No. 3, Pages 41-47. 2006.

[Rehab Duwairi, 2006] Rehab Duwairi. "*Machine learning for Arabic text categorization: Research Articles*". Journal of the American Society for Information Science and Technology, Vol. 57, No.8, Pages 1005-1010. 2006.

[**Ricardo Baeza-Yates et al, 1999**] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. "*Modern Information Retrieval*". ACM Press / Addison-Wesley. 1999.

[**Riyad Al-Shalabi et al, 1998**] Riyad Al-Shalabi and Martha Evens. "*A computational morphology system for Arabic*". In Workshop on Computational Approaches to Semitic Languages, COLING-ACL98, University of Montreal, Montreal, PQ, Canada, Pages 66-72, 1998.

[**Sabah Al-Fedaghi et al, 1989**] Sabah Al-Fedaghi and Fawaz Al-Anzi. "*A new algorithm to generate Arabic root-pattern forms*". In proceedings of the 11th national Computer Conference and Exhibition. King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia. Pages 391-400. 1989.

[**Sanmay Das, 2001**] Sanmay Das. "*Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection*". In the proceedings of the 18th International Conference on Machine Learning. Pages 74 - 81. 2001

[**Scott Deerwester et al, 1990**] Scott Deerwester and Susan Dumais and Thomas Landauer and George Furnas and Richard Harshman. "*Indexing by Latent Semantic Analysis*". Journal of the American Society of Information Science, Vol 41 No 6. Pages 391-407. 1990.

[**Shereen Khoja, 1999**] S. Khoja. "*Stemming Arabic Text*". Lancaster, U.K., Computing Department, Lancaster University. 1999.

[**Susan Dumais et al, 1998**] Susan Dumais, John Platt, David Heckermann, and Mehran Sahami, "*Inductive learning algorithms and representations for text categorization*", Proceedings of the 7th International Conference on Information and Knowledge Management. Bethesda, Maryland, United States. Pages 148-155. 1998.

[**Terri DeYoung, 1999**] Terri De Young,"*Arabic language history and Middle East/North African Culture Studies*", http:// www.indiana.edu / ~arabic / arabic_history.htm, 1999.

[**Thorsten Joachims, 1997**] Thorsten Joachims. "*A probabilistic analysis of the rocchio algorithm with tfidf for text categorization*". In proceedings of the 14th International Conference on Machine Learning {ICML}-97. Nashville, USA. Pages 143-151. 1997.

[**Thorsten Joachims, 1998**] Thorsten Joachims. "*Text categorization with support vector machines: learning with many relevant features*". In Proceedings of the 10th European Conference on Machine Learning ECML-98, Chemnitz, Germany. Pages 137–142. 1998.

[**Thorsten Joachims, 1999**] Thorsten Joachims. "*Making large-Scales SVM learning Pracical. Advances in Kernel Methods - Support Vector Learning*". MIT Press, Cambridge, MA. 1999.

[**Tom Mitchell, 1997**] Tom Mitchell. "*Machine Learning*". McGraw Hill. 1997.

[**Vladmimir Vapnik, 1995**] Vladmimir Vapnik. "*The Nature of Statistical Learning Theory*". Springer, New York, 1995.

[**William Cohen et al, 1996**] William Cohen and Yoram Singer, "*Context-sensitive learning methods for text categorization*". Proceedings of the 19th Annual ACM SIGIR Conference on Research and development in information retrieval. Zurich, Switzerland. Pages 307-315. 1996.

[**William Cohen et al, 1999**] William Cohen, and Yoram Singer. "*Context sensitive learning methods for text categorization*". ACM Transactions on Information Systems Vol 17 No 2. Pages 141–173. 1999.

[**Yiming Yang et al, 1999**] Yiming Yang, Xin Liu. "*A re-examination of text categorization methods*". In Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval. Pages 42–49. 1999.

[**Yiming Yang, 1997**] Yiming Yang and Jan O. Pedersen. "*A comparative study on feature selection in text categorization*". In Douglas H. Fisher, editor, Proceedings of the 14th International Conference on Machine Learning (ICML-97). Nashville, US. Pages 412–420. 1997.

[**Yiming Yang, 1999**] Yiming Yang. "*An Evaluation of Statistical Approaches to Text Categorization*". Journal of Information Retrieval, Vol 1 No 1/2. Pages 69-90. 1999.

[**Yong Hong Li et al, 1998**] Yong Hong Li and Anil K. Jain. "*Classification of Text Documents*" The Computer Journal Vol 41 No 8. Pages 537-546. 1998.

[**Zhaohui Zheng et al, 2003**] Zhaohui Zheng, Rohini Srihari, and Sargur Srihari, "*A Feature Selection Framework for Text Filtering*". In proceeding of the 3rd IEEE International Conference on Data Mining ICDM'03. Pages 705-708. 2003.

[**Zhaohui Zheng et al, 2003**] Zhaohui Zheng and Rohini Srihari. "*Optimally combining positive and negative features for text categorization*". In proceedings of the Workshop on Learning from Imbalanced Datasets ICML2003. Washington DC. 2003.

Appendix A

Stop Words List

ابا	اول	ذا	قد	ما	وسط
ابو	اين	ذات	ك	ماانفك	يكون
ابي	أي	ذلك	كان	مابرح	يلي
احد	ب	ذو	كذلك	ماذا	يمكن
اخا	بات	ذي	كل	مازال	يوم
اخر	بان	رغم	كم	مافتئ	
اخو	بد	شئ	كون	مايزال	
اخي	بدل	صار	كي	متى	
اذا	بعد	صبح	كيف	مساء	
الا	بعض	صير	ل	مسي	
الان	بل	ضحي	لا	مع	
التي	بيت	ضد	لازال	مما	
الذي	بين	ضمن	لاسيما	من	
الذين	تحت	ظل	لايزال	منذ	
اللتان	تكون	عل	لدي	نحو	
اللتين	تلك	على	لذلك	نفس	
اللذان	ثم	عن	لعل	هؤلاء	
اللذين	جدا	عند	لكن	هذا	
الي	حالي	عين	لم	هذه	
اليوم	حتي	غير	لماذا	هل	
اما	حول	ف	لن	هل	
امام	حيث	فقط	له	هن	
امس	حين	في	لو	هنا	
ان	خلال	فيما	ليت	هو	
او	دون	قبل	ليس	هي	

Appendix B

Arabic Words Patterns

Patterns with four letters	Patterns with five letters	Patterns with six letters	Patterns with seven letters	Patterns with eight letters	Patterns with nine letters
فاعل	فاعلة	فعلانة	استفعال	أستفعالي	أستفعالية
فعلى	فاعول	تفعيلة	انفعالة	افتعالية	
فعلة	فعلاء	أفتعال	أفتعالة		
تفعل	فعلان	انفعال	أفتعالي		
فعول	فعولة	إفعالي	أفعالية		
فعيل	فعيلة	إفعالا	مستفعلة		
فعال	فعلية	أفعلاء	مفعولية		
مفعل	فعائل	أفعلية	متفاعلة		
أفعل	فعالي	فاعولة			
	فعالة	فعلولة			
	فواعل	فعالية			
	فعالل	مستفعل			
	أفعال	مفاعلة			
	أفعلة	مفاعيل			
	تفعلة	منفعلة			
	تفعيل	مفعيلة			
	مفعال	مفعولة			
	مفعيل	مفتعلة			
	مفعول	متفعلة			
	مفعلة	مفعللة			
	مفاعل				
	مفتعل				
	متفعل				
	منفعل				
	مفعلل				
	فعللة				
	فعللي				

Appendix C

Words Clusters

The following words clusters are samples from those generated by the hybrid stemming approach of light and Trigram stemmer with $\alpha = 0.8$.

- تقرير تقريرا
- استثمار باستثمار استثماريا
- مليار مليارا بمليار
- جنيه جنيها
- رصيد لرصيد برصيد
- مكشوف كشوف
- قطاع بقطاع لقطاع
- اعمال عمال باعمال اعمالا لعمال
- موضوع موضوعا بموضوع
- عقود لعقود معقود
- قصور مقصور لقصور مقصورا
- كاملا كامل تكامل بكامل متكامل املا
  شاملا املاء حاملا
- مجال مجالس مجالا لمجال جالس
- رئيس لرئيس كرئيس رئيسا برئيس
- اقتصاد اقتصاديا
- تجاوز يتجاوز تجاوزت لتجاوز تتجاوز
  بتجاوز
- جهود كجهود جهودا مجهود لجهود
  بجهود بمجهود

- سياس سياسيا لسياس بسياس قياسيا سياسا
- نجاح بنجاح انجاح لنجاح نجاحا
- تمثل تتمثل متمثل تمثلت يتمثل
- فارق افارق بفارق لفارق
- عالم عالمن لعالم معالم عالما معال معالج بعالم بمعالج
- لمجلس مجلس بمجلس مجلسا
- وزراء زراء لوزراء
- حكوم لحكوم بحكوم محكوم بحكومت
- تقدمت تقدم متقدم تتقدم ستقدم قدمت ستتقدم لتقدم يتقدم تقدما
- قرار قرارا بقرار اقرار باقرار بقرارا اقرا اقراص
- حصول محصول لحصول بحصول
- محافظ لمحافظ بمحافظ حافظ تحافظ يحافظ محافظا
- توقع نتوقع يتوقع متوقع تتوقع اتوقع توقعت ستوقع
- تحسن تحسنت بتحسن يتحسن بتحس
- معامل عامل بمعامل عاملا يعامل نعامل
- بورص ببورص لبورص
- تراجع بتراجع تراجعت تتراجع يتراجع
- مشروع بمشروع لمشروع مشروعا شروع
- سلام اسلام سلاما بسلام لسلام سلامت
- بمؤسس مؤسس لمؤسس كمؤسس مؤسسا

- لتوزيع توزيع بتوزيع
- تناول تتناول تناولو متناول تناولت
  يتناول نتناول بتناول سنتناول
- سابق مسابق لمسابق سابقا مسابقت
  تسابق
- يشارك شارك يشار شاركت مشاركت
  سيشارك
- حاسب محاسب تحاسب يحاسب لحاسب
  حاسبت حاسبا
- تكنولوجيا تكنولوج لتكنولوجيا فتكنولوجيا بتكنولوجيا
- مجموع لمجموع مجموعت بمجموع جموع
  كمجموع
- برئاس رئاس لرئاس
- افريق فريق لفريق فريقا بفريق
  تفريق
- بمرحل مرحل مرحلت كمرحل رحلت
- مؤتمر لمؤتمر مؤتمرا بمؤتمر
- تشهد ستشهد استشهد
- مباحثا مباحث باحث مباح لمباحث
  بمباحث لباحث
- توظيف وظيف لتوظيف
- يتردد تردد ترددت متردد تتردد
- مناقش ناقش يناقش سيناقش لمناقش
  ناقشت مناقشا ناقشا مناقشت بمناقشت
- طالب يطالب مطالب طالبت اطالب
  مطالبا تطالب لمطالب لطالب
- مسائل سائل وسائل بوسائل رسائل
- شاعر مشاعر بمشاعر شاعرا لشاعر

- فاعل تفاعل تفاعلت تفاعلا يتفاعل
- تصريح صريح تصريحا بتصريح
- ترجم ترجمت مترجم بترجم يترجم