6-1-2013

Arabic sentence-level sentiment analysis

Amira Magdy Shoukry

The American University in Cairo

School of Science and Engineering

ARABIC SENTENCE LEVEL SENTIMENT ANALYSIS

A Thesis Submitted to

The Department of Computer Science and Engineering

In partial fulfillment of the requirements for

The degree of Master of Science

By

Amira Magdy Shoukry

Under the supervision of

Dr. Ahmed Rafea

Spring 2013

ACKNOWLEDGMENT

In the name of Allah the most gracious and most merciful

I would like to thank my great supervisor, Dr. Ahmed Rafea, who always helped me, welcomed my questions and gave me a lot of recommendations and suggestions. I would not have reached this phase, if it were not for his permanent support, advice, and guidance.

I would also like to express my thanks to my thesis committee members, Dr. Mohamed Mostafa, Dr. Sherif Kassas and Dr. Reem Bahgat for their support, guidance and helpful feedback.

Moreover, I would like to recognize ITIDA for sponsoring this project entitled "Semantic Analysis and Opinion Mining for Arabic Web", and the Egyptian industrial company LINK-Development and its team for their help in developing a tool for collecting and annotating the tweets.

Finally, I thank my beloved parents, and friends for their permanent support, appreciation and patience. I am also grateful for my fiancé, who has given me a lot of help and support during doing my research and while running the experiments. I would like to dedicate this thesis to them all.

ABSTRACT

Sentiment analysis has recently become one of the growing areas of research related to text mining and natural language processing. The increasing availability of online resources and popularity of rich and fast resources for opinion sharing like news, online review sites and personal blogs, caused several parties such as customers, companies, and governments to start analyzing and exploring these opinions. The main task of sentiment classification is to classify a sentence (i.e. review, blog, comment, news, etc.) as holding an overall positive, negative or neutral sentiment. Most of the current studies related to this topic focus mainly on English texts with very limited resources available for other languages like Arabic, especially for the Egyptian dialect.

In this research work, we would like to improve the performance measures of Egyptian dialect sentence-level sentiment analysis by proposing a hybrid approach which combines both the machine learning approach using support vector machines and the semantic orientation approach. Two methodologies were proposed, one for each approach, which were then joined, creating the hybrid proposed approach. The corpus used contains more than 20,000 Egyptian dialect tweets collected from Twitter, from which 4800 manually annotated tweets will be used (1600 positive tweets, 1600 negative tweets and 1600 neutral tweets). We performed several experiments to: 1) compare the results of each approach individually with regards to our case which is dealing with the Egyptian dialect before and after preprocessing; 2) compare the performance of merging both approaches together generating the hybrid approach against the performance of each

approach separately; and 3) evaluate the effectiveness of considering negation on the performance of the hybrid approach. The results obtained show significant improvements in terms of the accuracy, precision, recall and F-measure, indicating that our proposed hybrid approach is effective in sentence-level sentiment classification. Also, the results are very promising which encourages continuing in this line of research.

Different classification effectiveness measures were used like: 1) the accuracy, 2) precision, 3) recall, and 4) F- Measure (F-score) to help us in evaluating the performance of the proposed prototype and the effectiveness of the suggested features set. Finally, we performed tests of significances for the resulting models within each algorithm (ML and SO) to evaluate the relative performance or the true difference between the models within each algorithm.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS

FN	**False Negative**
FP	**False Positive**
IG	**Information Gain**
ML	**Machine Learning**
NB	**Naive Bayes**
NER	**Named Entity Recognition**
NLP	**Natural Language Processing**
SO	**Semantic Orientation**
SVM	**Support Vector Machines**
TN	**True Negative**
TP	**True Positive**

CHAPTER 1

INTRODUCTION

The explosion of Web 2.0 and the rising numbers of web forums, reviews, blogs, social network websites, and others have caused the exposure to millions of individual comments or opinions to increase enormously. These online comments or opinions can be about several topics like books, movies, electronic products, cars, politics, and many others. This fact raised the interest of different parties such as customers, companies, and governments to start analyzing and exploring these opinions. For customers, the rapid growth of e-commerce has caused the people to buy more from online shops and stores, thus people started to review comments about these products and learn from other people's experiences to get a general idea about these products in order to help them in making the best choice (Rushdi-Saleh et al., 2011). While for companies and governments, both parties are interested in presuming the opinion of the public, the first with respect to their products and services, while the second with respect to the new rules and regulations they have set. After the spread of the revolutions all over the world, researchers started to become more interested in analyzing public opinion on social networks.

This chapter is organized as follows: section 1.1 gives a brief background on the thesis topic, section 1.2 describes the problem definition for choosing this specific area of research in our study; section 1.3 and 1.4 explain the motivation and the

objective for working in sentence-level Arabic sentiment classification; and finally section 1.5 will list the chapters and sections in this study, together with their contents.

1.1 Background

The idea of processing and analyzing the people's comments and reviews about different topics has attracted many researchers to work on creating some kind of an automated tool that can identify the sentiment or opinion of a piece of text whether a document, sentence, or phrase (Liu, 2010). This task has been given various names like sentiment analysis, sentiment orientation, subjectivity analysis, or opinion mining (OM), and it is considered to be an emerging new research field in machine learning (ML), computational linguistics, and natural language processing (NLP).

In this study, we are interested in the sentiment classification at the sentence-level for the Arabic language in which the aim is to classify a sentence whether a blog, review, tweet, etc. as holding an overall positive, negative or neutral sentiment with regards to the given target. After studying the majority of research done in this area, we would like to improve the performance of sentiment classification for sentences written in Arabic, particularly in the Egyptian dialect, by proposing a hybrid approach to be used.

Although Arabic is considered one of the top 10 languages mostly used on the Internet based on the ranking carried out by the Internet World State[1] rank in 2010 and it is spoken by hundreds of millions of people, there exist limited annotated resources for sentiment analysis such as labeled corpora, and polarity lexica. This could be considered the main reason which has motivated the generation of an opinion corpus for Arabic in this work.

The fields of text mining and information retrieval for the Arabic language has been the interest of many researchers, and various studies have been carried in these fields resulting in diverse resources, corpora, and tools available for implementing applications like text classification (Duwairi et al., 2009) or name entity recognition (Shaalan & Raza, 2009). However, Arabic resources that focus on mining and analyzing opinions and sentiments are very difficult to find. This may be because of the complex nature of Arabic language itself (Al-Shalabi and Obeidat, 2008) as numerous various forms can exist for the same word using different suffixes, affixes, and prefixes. Different words with completely different meanings can be produced using the same three-letter root.

1.2 Problem Definition

Sentiment analysis or opinion mining is considered to be one of the newest emerging research fields caused by the great opinionated web contents coming from

[1] http://www.internetworldstats.com

reviews, blogs social network websites, and many others. Most of the research done in this field was focused on English texts with very limited research done for other languages such as Arabic, particularly the Egyptian dialect which is the language of interest for this research. The main reason behind this fact is the lack of resources that can analyze sentiments in other languages or dialects, given that generating these resources is considered very time and labor consuming.

Based on our conducted survey, we can conclude that the main problems in the context of Egyptian dialect sentiment analysis are:

- Preprocessing the text: The majority of the available preprocessing tools like stemmers, stop-words lists, etc. are mainly built for the modern standard Arabic (MSA) lacking the dialect specific rules.
- ML approach: The lack of suggested feature sets and classification algorithms to be used in the classification of the Egyptian dialect text.
- SO approach: The absence of dialect specific lexicon with weights for each sentiment word makes this approach to be less investigated in the field of sentiment analysis for Egyptian dialect.

1.3 Objective

The objective of this study is to investigate:

a. The different approaches (ML and SO) producing high sentiment classification quality for opinions embedded in certain sentences, like tweets or micro-blogs written in Egyptian dialect, as positive, negative or neutral.

b. The impact of preprocessing on the sentiment classification accuracy.

c. The impact of features for the ML approach and the training corpus size on the classification quality.

d. The impact of the semantic lexicon size for the SO approach on the classification quality.

e. Building a hybrid approach and measure the enhancement in quality of sentiment classification if any.

The approach followed to achieve the above objectives can be summarized in the following steps: 1) comparing the ML classifications methods: SVM and NB; 2) comparing the SO approach to the ML approach; 3) developing a mechanism for preprocessing the tweets (normalization, stemming and stop-words removal) and measuring the impact of this mechanism on the performance of both approaches; 4) identifying the features to be used in the ML approach; 5) building an annotated corpus which will be used to train and validate the best classifier at different corpus sizes; 6) building different sizes sentiment lexicon from the built annotated corpus; 7) experiment the different mechanisms for combining these approaches in order to benefit from the advantages of each approach; and 8) proposing a simple straight forward method for negation detection in the hybrid approach.

This study is part of a bigger project focusing on developing a prototype that can "feel" the pulse of the Arabic users with regards to a certain hot topic. This bigger project also includes extracting the most popular Arabic entities from online

Arabic content together with the users' comments related to these entities, and then these extracted popular entities will be used to build semantically-structured concepts. It also includes building relations between different concepts and analyzing them to get a sense of the most dominant sentiment, using online user feedback, and thus identifying the general opinion about the topic.

1.4 Motivation

The textual information usually falls into two main categories: facts and opinions. Facts focus on objective data transmission while opinions express the sentiment of their authors. In general, sentiment analysis aims to determine the attitude of a writer with respect to some topic or the overall tonality of a document. The attitude may be his or her judgment or evaluation, emotional state, or the intended emotional effect. A basic task is to classify the polarity of a given text at the document, sentence, or feature/phrase level (Michelle, 2010).

Choosing to work with the Arabic language is due to several factors. First, Arabic sentiment analysis is of growing importance due to its already large scale audience. Second, the Arabic language is both challenging and interesting because of its history, the strategic importance of its people, the region they occupy, and its cultural and literary heritage. And last but not least is the major role the Internet,

social media and social network websites like Twitter[2], TwitPic[3], Facebook[4], etc… played in the current Arab spring.

1.5 Organization of the Thesis

The rest of this Thesis is organized as follows: chapter 2 explains and surveys the different approaches used for sentence-level sentiment classification. Chapter 3 explains the tools used in the preprocessing stage of the tweets. Chapter 4 talks about the proposed methodologies for each approach for sentence-level sentiment classification, and how they are joined to produce hybrid approach. In chapter 5, we will discuss and analyze the different experiments that we have performed to evaluate the performance of our proposed hybrid approach against the baseline. Finally, in chapter 6, we will conclude the thesis and list some directions for future work. There are 4 appendices at the end of the document: Appendix A, which shows some sample tweets from the datasets we have used in our experiments; Appendix B, which shows the list of stop words used in the preprocessing stage; Appendix C shows samples from the list of positive sentiment words and samples from the list of negative sentiment words; and finally Appendix D lists the negation words used.

[2] https://twitter.com/
[3] http://twitpic.com/
[4] https://www.facebook.com/

CHAPTER 2

LITERATURE SURVEY

Several potential applications for organizations and businesses can now be developed using the concept of sentiment analysis or opinion mining from text. Examples of these applications may include deducing the opinion of the public with regards to a specific topic, building an automatic recommendation system, extracting the customer sentiments about certain product, etc… (Pang & Lee, 2004). The two main tasks involved in sentiment analysis are: (1) to determine whether a given piece of text is objective or subjective, i.e. whether it contains an opinion or is just a fact, and (2) to determine the sentiment of this given text by classifying it as positive, negative, or neutral with respect to the given target (Abbasi et al., 2008). Furthermore, sentiment analysis can be carried out on two levels; the first level is the sentence level, while the second level is the document level. For the sentence level, sentiment mining is difficult because the semantic orientation of words is highly context-dependent, while for the document level, sentiment mining is difficult because of the possible presence of more than one contradicting opinion about the same topic (Farra et al., 2010). Further review on sentiment analysis can be found in Liu (2010) and Pang & Lee (2008).

According to the type of the sentiment (positive or negative, subjective or objective), and the levels of classification (phrase, sentence, or document level),

techniques for sentiment classification differ. However, in order to be able to determine the sentiment of the sentence, two main assumptions have to be made first. The first assumption is that the sentence represents the opinion of just one author, and the second assumption is that the sentence holds the author's opinion about only one topic. Ensuring that these two assumptions are satisfied, we can then go into the process of determining the sentence's sentiment. There are two main approaches for sentiment classification: machine learning (ML) and semantic orientation (SO).

This chapter is organized as follows: section 2.1 talks more in-depth about the ML approach, mentioning the different feature sets and machine learning techniques used in the literature, together with a survey of some of the systems which have used this approach. Section 2.2 describes the SO approach and presents a survey about the recent work done in this direction. Finally, section 2.3 presents a comparison between the two approaches, while section 2.4 reviews the techniques for creating a hybrid model combining both approaches.

2.1 The Machine Learning Approach

The ML approach is typically a supervised approach in which a set of data labeled with its class such as "positive" or "negative" are converted into feature vectors. This conversion process focuses on the more important and salient features present in the sentence. Then, these vectors are used by a classifier, employing one of the ML algorithms, as a training data inferring that a combination of specific features yields a specific class. This process results in the creation of a model used for

predicting the class of unseen or new data called testing data. Examples of ML algorithms are Support Vector Machine (SVM), Naïve Bayesian Classifier, Maximum Entropy, etc… Several feature sets have been proposed by different researchers for the process of sentiment analysis (Lee & Pang, 2008), some of which are related to our work and are listed in section 2.1.1. Afterwards, section 2.1.2 explains briefly the theoretical foundation of SVM and NB.

2.1.1 Features used in the ML Approach

a. Term Presence vs. Frequency

In information retrieval (IR), feature vectors have been usually used to represent a piece of text. They are vectors in which the entries correspond to the individual terms in the text. In standard IR, Term Frequencies were used extensively with regards to the TF-IDF weighting's popularity. However, better results were obtained using the Term Presence (Pang et al., 2002) rather than Term Frequencies. Term Presence is a binary-valued feature vectors wherein the entries simply show whether a term appears taking the value of 1, or not taking the value of 0. The later approach was more effective in reviewing the polarity classification than the real-valued feature vectors (Liu, 2010). This finding reflected the fact that some topics are more likely to be highlighted by the repeated recurrences of some terms, whereas the overall sentiment may not be emphasized by the frequent use of the same terms. "On a related note, *hapax legomena*, or words that appear a single time in a given corpus, have been found

to be high-precision indicators of subjectivity" (Wiebe et al., 2004). However, in our proposed ML approach, we have used the frequencies of all the words present in the corpus to give higher weights to those frequently used words.

b. N-grams

N-grams are from the frequent features employed in the classification of text. A lot of discussions have been carried out on the appropriate size of the grams to be used. Grams are words which are frequently repeated in the corpus. Unigrams (only one word, like "فيلم" *film*) were found to perform better than bigrams (two consecutive words, like "ممثل سينمائي" *movie star*) in categorizing movie reviews using sentiment polarity, whereas bigrams and trigrams (three consecutive words, like "فيلم الخيال العلمي" *science fiction film*) result in improved product review polarity classification (Liu, 2010). This feature is used in our proposed ML approach as we have extracted all the unigrams, bigrams, and trigrams in the corpus with their corresponding frequencies.

c. Opinion words and phrases

Some words are sometimes used to express positive or negative sentiments; these words are called opinion words. Examples of positive opinion words are "رائع" (wonderful), "جميل" (beautiful), "مذهل" (amazing), and "حسن" (good). Examples of negative opinion words are "فقير" (poor), "سيئة" (bad), and "مروع" (terrible). Many of the opinion words are either adjectives or adverbs; however

nouns like ""قمامة الـ (rubbish), "هلابزلا" (junk), and "تالضف" (crap) and verbs like "هرك" (hate) and "(ekil) "ثل م can also be used to reveal opinions (Liu, 2010). Moreover, there are also phrases and idioms which can be used to express opinions. An example of an idiom is "cost someone an arm and a leg", which is usually used to reflect negative sentiment or opinion. That is why many researchers believe that opinion words and phrases have major roles in sentiment analysis. These features were used in the SO approach utilizing a comprehensive set of opinion words.

d. Dependency Relations

Dependency relations within the features sets were also considered by many researchers. This linguistic analysis is specifically more applicable with respect to short textual units. For example, a subtree-based boosting algorithm using words dependency based features (like higher-order n-grams) yields better results than the bag-of-words baseline. Parsing, which is identifying the words in the text, can also be used for representing *valence shifters* such as negation, intensifiers, and diminishers (Kennedy & Inkpenl, 2006). We didn't use this feature because of the absence of the valence shifters' lists representing intensifiers, diminishers, etc... for Egyptian dialect.

e. Negation

In sentiment analysis, dealing with negation words is very important as their presence usually alters the orientation of the opinion. For example, the sentences "I like this camera" and "I don't like this camera" are believed to be very similar by most frequently used similarity measures as the only different word is the negation term, putting the two sentences into opposite classes. Dealing with negations can be performed in two different ways; directly, and indirectly. In the direct way, the negation words are encoded directly into the initial features' definitions. While in the indirect way, a second-order feature of a text unit in which the feature vector used for initial representation is built essentially ignoring the negation words, which is then changed into a different negation-aware representation (Pang & Lee, 2008). In an attempt to represent negation words more precisely, certain part-of-speech tag patterns are searched for, and then the complete phrase is tagged as a negation phrase (Na et al., 2004). This approach has been applied on a dataset of electronics reviews, and it was observable that there was an improvement of about 3% in accuracy resulting from this negation modeling. Further improvements can be possibly reached by deeper (syntactic) analysis of the sentence (Liu, 2010). Moreover, sometimes negations are expressed in subtle ways like in an irony or sarcastic ways which are often very difficult to detect. For example, in the sentence "[it] avoids all clich´es and predictability found in Hollywood movies" the word "avoid" is considered to be an unexpected polarity reverser word. Negation words must be carefully handled as not all occurrences of such words mean negation. For example, "not" in "not

only … but also" does not change the orientation direction. This feature is considered one of the important features to consider as negations greatly shift the meaning of the sentence.

f. Stylistic Features

Stylistic features include the number of punctuation marks and function words, as well as the lexical and structural attributes as shown in Abbasi et al. (2008). The lexical features fall into two main categories which are word- based and character-based statistical measures. Some examples of the word-based lexical features are: 1) Total words number; 2) Words per sentence; 3) Word length distribution; etc…. Whereas examples of character-based lexical measures are: 1) Total characters' number; 2) Characters per sentence; 3) Characters per word; and 4) The frequency of individual letters (Morsy 2011). On the other hand, the structural features are more concerned with the layout and organization of the text. Examples of structural features are: 1) the number of paragraphs; 2) the average paragraph length; 3) the total number of sentences per paragraph. Using these features along with other features can improve the accuracy of the sentiment classification system in the case of analyzing rich texts (Abbasi et al., 2008). These features were applied in our proposed SO approach as reviews or opinions usually contain some smiley faces or punctuation marks expressing the user's sentiment with respect to a certain topic.

The majority of the research carried out in the field of sentiment analysis, especially in the case of Arabic, focused on using the different ML algorithms employing different feature sets and comparing their performance. In the study carried by Rushdi-Saleh et al. (2011) for classifying movie reviews as positive or negative, they have used two different weighting schemes in the validation process: term frequency–inverse document frequency and term frequency, as well as testing the effect of stemming in the text preprocessing process. They have reached an accuracy of 90% with SVM compared to 84% with NB using the same weighting scheme and n-gram model. The results they obtained were close to the ones obtained by Pang et al. (2002) who have also used the term frequency inverse document frequency weighting scheme employing the SVM classifier without applying stemming in the preprocessing process.

2.1.2 Theoretical Foundation of SVM and Naïve Bayes Classifiers

In order to build a model to be used in the classification problem of any unlabeled or unseen data, there has to be a set of labeled data with their target class. If there are only two target classes, then this is a binary classification problem; otherwise it is a multi-class classification problem. Building this model involves selecting the feature sets which are believed to be relevant to the target class. These feature sets will be extracted from the sentences in order to create the feature vector representation for each sentence with each feature having its corresponding value. Each classifier has a function: $f(x) : Rd \rightarrow R$ which assigns the sentence to its class

depending on the result of this function. For example, a binary classifier assigns a sentence to the positive class if its function value is greater than or equal to zero, and assigns it to the negative class if its function value is less than zero.

In this chapter, the theoretical foundations of both SVM and Naïve Bayes classifiers are briefly described, since they were used in the literature for sentiment classification and they will be used in our experiments.

a. Theoretical Foundation of SVM

The SVM main idea resides in defining the decision boundaries which are based on the decision planes' concept. These decision planes are defined to be those ones that separate between a set of objects having different class memberships. A special rule is constructed, called the linear classifier, in which its function can be written as:

$$f(x; w, b) = \langle w, x \rangle + b$$

Equation 2-1: Linear Classifier

where w and b are the function parameters and "<","> " signs are the inner product of the two vectors.

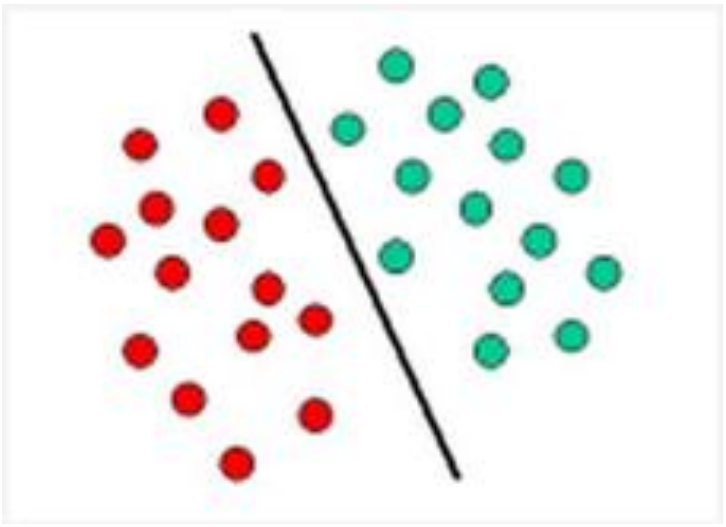

Figure 2-1: SVM Classifier[5]

The data points used to train the classifier are shown in the above figure. The figure shows that the classifier's main aim is to find the best hyper-plane which separates the negative class data points from the positive class data points with the maximum possible margin for each set of points from the hyper-plane (Fradkin & Muchnik, 2006). The data points on the margins are called "support vectors". The most important property of the training data in SVM is to be linearly separable, where:

$$y_i f(x_i) > 0, \forall i = 1, \ldots, l$$

Equation 2-2: Test for Linearly Separable Data

[5] http://www.statsoft.com/textbook/support-vector-machines/

which means that the two hyper-planes of both margins can be selected in a way that there are no data points between them (Fradkin & Muchnik, 2006).

b. Theoretical Foundation of Naïve Bayes

The NB main idea is based on the so-called Bayesian theorem which is more suitable for inputs with high dimensionality. The model is called naive because it assumes that the attributes are conditionally independent of each other given the class. This assumption gives it the ability to compute probabilities of the Bayes formula from a relatively small training set. The algorithm is based on conditional probabilities. The final classification is based on the product of two probabilities producing what is called the posterior probability. The first probability is called the prior probability, while the second probability is called the likelihood probability. The prior probability is an unconditional probability based on the previous experience. In other words, it is the knowledge's state before the data is observed. It is calculated for each class:

$$P(C = i) = \frac{\text{\# data points in class i}}{\text{Total Number of Points}}$$

Equation 2-3: Prior Probability

Since the objects are well clustered, it is reasonable to assume that the more data points of a particular class in the vicinity of X, the more likely that the new cases

belong to that particular class. To measure this likelihood, a circle is drawn around X which encompasses a number of points (to be chosen a priori) irrespective of their class labels. Then the number of points in the circle belonging to each class label is calculated. The likelihood is calculated as follows:

$$P(X \mid C = i) = \frac{\text{\# of i in vicinity of X}}{\text{Total \# of i cases}}$$

Equation 2-4: Likelihood Probability

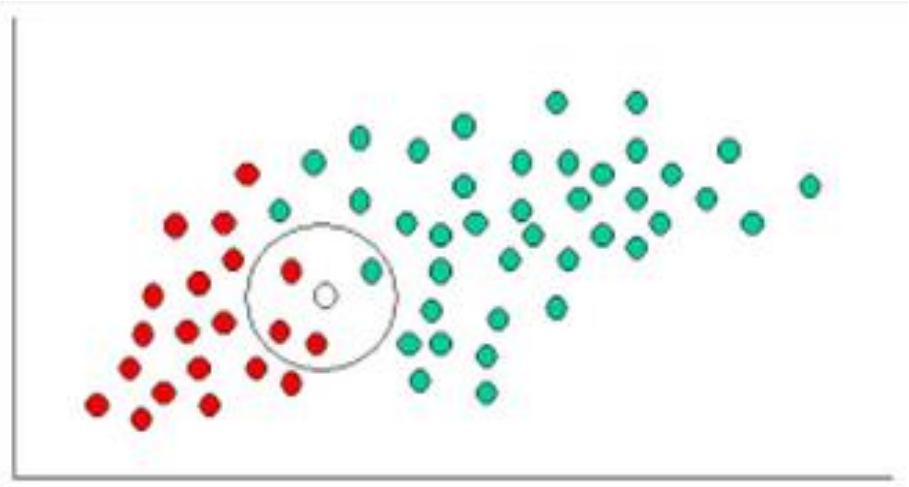

Figure 2-2: NB Classifier[6]

The above diagram illustrates the plane of the data points used to train the classifier, and the vicinity of the new object to be classified. Having obtained these

[6] http://www.statsoft.com/textbook/naive-bayes-classifier

two probabilities, the prior and the likelihood, the final classification is the result of their products to form the posterior probability using the so-called Bayes' rule:

$$f_i^{NB}(\mathbf{x}) = \Pi_{j=1}^{n} P(X_j = x_j | C = i) P(C = i)$$

Equation 2-5: Bayes' Rule

which is calculated for each class, and the class with the highest value will be the class of the new object.

Xia & Zong (2010) showed that in the case of SVM, unigrams perform better, whereas in the case of NB, higher-order n-grams and dependency relations perform better. That is because of the nature of the algorithms themselves. SVM, which is a discriminative model, can capture the complexity of relevant features and the independency which is present more in unigrams than in higher order n-grams. Whereas NB, which is a generative model, can capture the feature independence assumptions present in bigrams and dependency relations (Xia & Zong, 2010).

2.2 The Semantic Orientation Approach

The Semantic Orientation approach is an unsupervised approach in which a sentiment lexicon is created possibly in three ways: manually, semi-automatically, or automatically in which the semantic intensity of each word is represented by a number indicating its class. Using this lexicon, all the sentiment words in the sentence are extracted and their polarities are summed up to determine if the sentence has an overall positive or negative sentiment, together with its intensity to determine whether

the sentence holds strong or weak intensity (Turney, 2002). The SO approach is domain-independent, since one lexicon is built for all domains. Section 2.2.1 illustrates one of the methods used in calculating the semantic orientation, while section 2.2.2 describes some of the methods used in building the semantic lexicon.

2.2.1 Calculating the Semantic Orientation

Turney (2002) adopted one of the earlier methods used in this approach in which the class of the sentence was determined using the average semantic orientation of different phrases present in the sentence. Thus, the semantic orientation of each phrase is calculated as the difference between the similarity of the given phrase to a positive reference word "excellent" and the similarity of the given phrase to a negative reference word "poor":

$$\mathrm{SO}(\textit{phrase}) = \mathrm{PMI}(\textit{phrase}, \text{“excellent”}) - \mathrm{PMI}(\textit{phrase}, \text{“poor”})$$

Equation 2-6: Calculating Semantic Orientation

Therefore, the semantic orientation of the given phrase is positive when it has a strong association with the word "excellent" and negative when it has a strong association with the word "poor". The similarity score measure, Pointwise-Mutual Information (PMI), is used to measure the association between the pairs of words or phrases. The PMI between two words is calculated as follows:

$$\mathrm{PMI}(word_1, word_2) = \log_2 \left[\frac{\mathrm{p}(word_1 \ \& \ word_2)}{\mathrm{p}(word_1)\ \mathrm{p}(word_2)} \right]$$

Equation 2-7: The similarity score measure (PMI)

where *p(word1&word2)* denotes the probability that both words occur together. The degree of statistical dependence between the two words is measured by the ratio between *p*(*word*1 & *word*2) and p(*word*1) p(*word*2). Finally, the log of this ratio gives the amount of information gained about the presence of one of the words when the other word is observed.

2.2.2 Building the Semantic Lexicon

Given that the presence of semantic dictionaries or lexicons for Arabic sentiment mining is very limited, some researchers tried to build sentiment lexicons of Arabic words and expressions. For example, Farra et al. (2010) determines the class of the sentence using a list which stores the semantic orientation of some Arabic word roots which are extracted using a stemmer program. In the classification process, the root of each word is extracted using an Arabic stemmer, and then this root is checked against the stored dictionary. If the root is present, its sentiment is extracted as positive, negative, or neutral. Otherwise, the dictionary prompts the user to identify the polarity of a word it has not learned and adds its root to the list of learned roots.

Another method used in the semantic orientation approach is based on using one of lexical resources for sentiment analysis available in English like SentiWordNet[7]. SentiWordNet was built using the English lexical database, WordNet[8] (Miller, 1995). The class of each word in SentiWordNet was determined using a 2-step process. The first step started with a small list of positive and negative words, and then words with similar polarity were searched for using binary relations of WordNet (Baccianella et al., 2010). Then, in the second step the final polarity of the words was determined by running iterations on the words until they congregate to their final polarity. For each word, three different polarity score types; objective, positive and negative are associated, describing its intensity ranging from 0 to 1, together with its Part-Of-Speech (Esuli & Sebastiani, 2006). To use this lexicon, first the Arabic sentences are translated into English using one of the standard translation software. Then, these translated sentences are classified according to its sentiment into one of the classes "positive" and "negative". The process starts by extracting the sentiment words in each translated sentence, and then their polarities are determined using the scores available in SentiWordNet.

2.3 Comparing ML and SO Approaches

It is observable that the ML approach was adopted more than the SO approach in the literature because of the different sets of features and algorithms that can be used depending on the type of classes to predict, and the level the algorithm will be

[7] http://sentiwordnet.isti.cnr.it/
[8] http://wordnet.princeton.edu/

applied (document, sentence, or phrase). On the other hand, the SO approach is domain independent giving higher generality across domains. However, each approach has its concerns. For example, the ML approach concern is selecting the right features, and classification algorithm, while the SO approach concern is creating a comprehensive lexicon with correct prior polarity to the words and methods to handle their contextual polarity (Morsy 2011).

ML and SO approaches have some major differences. The first difference between them is that in the ML approach the classifier can be trained for a domain-specific polarity; e.g. "long lasting friendship" implying positive sentiment versus "long time to reach" implying negative sentiment, unlike the case of the lexicon in which for each word its polarity is initially defined. On the other hand, the accuracy of the classifier increases as the size of the trained data increases, meaning that a huge corpus labeled with its class (positive or negative) is required. This process involves collecting data from different domains and websites like news, blogs, reviews, news articles, movies, products, politics, etc…, then labeling these data manually, whereas in the case of a dictionary it doesn't need all this process. However for the SO approach, there is a tradeoff between saving the word's root and saving the actual encountered word with its derivative letters, prefixes, affixes and suffixes. In case the actual word is saved in the dictionary, it will enhance the dictionary's accuracy, but it will slow the learning curve. In case the stemmed word is saved, it will speed up the learning process and decrease the dictionary's size, but it will lead to lower accuracy as in Arabic a single root could sometimes correspond to either a positive, negative, or neutral word. For example, if the root of "جميل" (beautiful) is extracted to "جمل"

(the corresponding three-letter root), "جميل" will be identified by the user as positive and its root "جمل" will be stored as a positive root. However, if the dictionary then encounters the word (which can also mean 'camel') it will thus be labeled as positive while it is actually neutral (Farra et al., 2010).

2.4 Hybrid Approach

Given the advantages and the disadvantages of both ML and SO approaches, some of the proposed mechanisms for sentiment analysis tried to combine them together so that they can take advantage from the benefits of each approach. For example, Farra et al. (2010) proposed an approach to automate the Arabic sentence-level sentiment classification combining both syntactic and semantic features. The features they have used included: frequency of positive, negative, and neutral words in each sentence using the semantic interactive learning dictionary they have built, frequency of negations (such as , لا , لن ,لم ,ليس which are negation words in Arabic and hold several meanings such as not, won't, didn't, don't) , frequency of special characters (!) and (?), frequency of emphasis words (' خاصة especially', ' لدرجة to the extent that', ' كثيرا very much', ' جدا really', على الإطلاق at all' , etc.), frequency of conclusive words (' خلاصة in conclusion' , ' لذلك and that is why' , ' أخيرا finally', etc) , frequency of contradiction words, and other similar features. Similarly, Kouloumpis et al. (2011) proposed an approach to classify the sentiment of the English tweets in Twitter. They have built an ensemble model of two classifiers: one which uses ngrams only, while the other uses both ngrams and lexicon features. For the lexicon features

they have created three features for each tweet based on the presence of any words from the lexicon. They have used the English words listed in the MPQA subjectivity lexicon (Wilson et al., 2009) which are tagged with their prior polarity: positive, negative, or neutral. By comparing the results of both classifiers, it was noticeable that the addition of sentiment lexicons to the n-grams has increased the accuracy of the classification task.

CHAPTER 3

PREPROCESSING EGYPTIAN DIALECT TWEETS

The majority of the text produced by the social websites is considered to have an unstructured or noisy nature. This is due to the lack of standardization, spelling mistakes, missing punctuation, nonstandard words, and repetitions (Al-Shammari, 2009). That is why the importance of preprocessing this kind of text is attracting attention these days because of the presence of several websites producing this noisy text. There are three steps in the preprocessing process: 1) normalization, 2) stemming, and 3) stop words removal. Normalization is the process of transforming the text in order to be consistent, thus putting it in a common form; stemming is the process of reducing derived or inflected words to their stem, base or root form; and stop words removal is the process of removing those words which are natural language words having very little meaning, such as "في" (in), "علي" (on), "انت" (you), "من" (of), etc…

This chapter is organized as follows: section 3.1 describes the normalization process and shows the rules applied to Egyptian Dialect tweets; section 3.2 talks more in-depth about the stemming process and the dialect specific implemented stemmer; and finally section 3.3 explains the stop word removal process and how the list of stop words was constructed.

3.1 Normalization

The normalizing process puts the Arabic text in a consistent form, thus converting all the various forms of a word to a common form. For example the word "اذ ت" can have many different forms like "إنت", "أذ ت", "اذ ـــــت", etc. All these forms cause the word to be considered as three different words. Thus, they all need to be transformed to a single form. A normalizer[9] is implemented for doing this job using the programming language Ruby. This normalizer performs several tasks such as removing diacritics from the letters, removing 'ء' (Hamza), making both 'ي' and 'ى' change to 'ي'(y), etc… We have used this normalizer [9] as it is very efficient, and it handles most of the normalization rules. Table 3.1 defines language normalization rules:

Rule	Example
Tashkeel	حدثنا -> حَدَّثَنَا
Tatweel	الله -> اللـــــــــــــــه
Hamza	ؤ or ىء or ء -> ء
Alef	آ or أ or إ -> ا
lamalef	لا or لآ or لأor لإ -> لا
Yeh	ي or ى -> ي
Heh	ه or ة -> ة

Table 3-1: Normalization Rules

[9] http://arabtechies.sourceforge.net/projec/ normalization _ruby

3.2 Stemming

The stemming process reduces the words to their uninflected base forms. Sometimes the stem is different from the root, but it is useful as related words usually map to the same stem even if this stem is not in itself a valid root. Stemming is considered one of the most important stages in any Arabic information retrieval or text mining systems. Larkey et al. (2007) has proven that stemming Arabic terms is not an easy task because of its highly inflected and derivational nature. There are mainly two classes of stemmers for the Arabic language: aggressive stemmers (reducing a given word to its root) and light stemmers (identifying a set of prefixes and suffixes that will be removed). However, it is believed that the problem with aggressive stemmers is that as they reduce the words to their roots, most of the time it results in losing the specific meaning of the original words. This fact has caused this type of stemmers to be poor candidates for systems involving high accuracy in matching between similar words.

Due to the complexity of the Arabic language, several studies with various complexity levels were carried out to address stemming because of its significance in informational retrieval and text mining systems. However, most of these studies were mainly for modern standard Arabic (MSA) and so they can't handle the different dialect specific rules like the Egyptian dialect. For example, if we tried the MSA stemmer with the word "علشان" (because) the word would become "عش" (hut), since in MSA, when a word ends in "ان" it reflects duality; however, this word should not have been stemmed originally. This fact has forced us to implement our own customized stemmer. The main objective of the stemmer is to reduce the input word to its shortest

possible form without compromising its meaning. That is why we have adopted the light stemming methodology using Dialect specific set of prefixes and suffixes because in aggressive stemmers, reducing the word to its root can sometimes result in the mapping of too many related terms, each with a unique meaning, to a single root. Moreover, light stemmers are considered very simple to implement and have proven to be highly effective in several information retrieval systems. On the other hand, light stemmers are not applicable to some affixes and broken plurals which are very common in the Arabic language (Larkey et al., 2007). Consequently, in our implemented-light stemmer, we have combined some of the rules introduced in El-Beltagy and Rafea (2011), together with a set of rules we have introduced to handle broken plurals for Egyptian dialect which sometimes results in the addition of infixes to a word, as well as handling the removal of certain affixes. In our stemmer's implementation, we have built two lists: one for irregular terms (words that originally start or end by any of the prefixes or suffixes and should not be stemmed) and another one for irregular plurals and their singular forms. These lists are normalized and stemmed. Thus, the input word is first checked against these lists of irregulars; if it is present, then it won't be stemmed, otherwise the stemming rules will be applied.

The implemented stemmer consists mainly of three stages: 1) prefix removal, 2) suffix removal, and 3) infix removal which is mainly applying the rules for broken plurals. Generally, the prefix removal is the first stage attempted, followed by the suffix removal stage, and finally the infix removal stage. After each stage, the transformed word is checked against the dictionary to determine whether to continue with stemming it, or just stop. Tables 3.2 and 3.3 show the sets of prefixes and suffixes

proposed for the Egyptian dialect, while table 3.4 shows the set of rules for handling broken plurals. Most of the broken plurals' rules were inspired from the ones introduced in El-Beltagy and Rafea (2011). The new rules we proposed are highlighted in black.

Prefix	Meaning
ال	The
وال	And the
بال	With the
كال	Like the
فال	Then the
لل	For
وبال	And with the
ولل	And for
وكال	And like the
وفال	And then the
بي	he is
بت	she is
هي	he will
هت	she will
بن	we are
بتت بنت بيت	it is

Prefix	Meaning
و	and
ك	like
ف	then
ل	For or because
ب	With or at

Table 3-2: Set of Compound and Single Prefixes with their Meanings

Suffix set 1	يات , ات
Suffix set 2	ها , ون , وا , ين , ان , نا , هم , ت , ك , ي , ه , ة , و

Table 3-3: Sets of Suffixes

Rule ID	Condition	Rule	Example
R1	Length = 5 & 2nd char != و & 4th character = ئ & 3rd character = ا , & 5th char != ي or ه	Replace 3rd char with a (ي), delete 4th char, and add a (ة) at the end. If no match is found, attempt to find a match without the (ة)	حشائش
R2	Length = 5 & 2nd char = و & 4th character = ء & 3rd character = ا , & 5th char != ي or ه	Remove 2nd char and add a (ة) at the end. If no match is found, attempt to find a match without the (ة)	روائح
R3	Word ends with "ايا" and 2nd character != و	Remove last three chars, and append string (ية)	هدايا
R4	Length = 3 and 2nd character = 3rd character	Remove 3rd character	زمم
R5	Length = 4 and 3rd character = و and last character is not equal to ه	Remove 3rd character (و)	جذور
R6	Length = 5 and 3rd character is an ا	Remove 3rd character (ا)	مراكب
R7	Length = 5 and & 4th character is an ا & 5th character is ء	Remove 4th character (ا) & 5th character is ء and add an (ي) after the 2nd character	وزراء
R8	Length = 5 and 1st character is an ا and last character is a ة	Remove 1st character (ا) and last character (ة) and add an (ا) after the 2nd character	اتربة
R9	Length = 5 and 1st character is an ا and 4th character is an ا and second char != ي	Remove 1st and 4th characters(ا)	اشجار
R10	Length = 4 and 3rd character = و and 2nd character = 4th character	Remove 3rd and 4th characters	سدود
R11	Length = 5 && 2nd character = و and 3rd character is an ا	Remove 2nd character (و)	جوانب

Table 3-4: Rules for Broken Plurals

3.3 Stop Words Removal

There is not one definite list of stop words for Arabic. Depending on the type of the application they are implementing, authors use different stop words lists. Some authors build lists that consist mainly of the most common and short function words like "في" (in), "من" (of), "علي" (on), etc…[10]. On the other hand, some authors build lists that contain the most common words including lexical words like "مثل" (like), "يريد" (want), "يقول" (say), etc…[11].

Given the absence of any stop words list for the Egyptian dialect, we had to build this list from the beginning. The process started by identifying the words in the whole corpus (20,000 tweets) between different frequency ranges as shown in figure 3.1. The figure shows the number of the words in each frequency range, and it is clear from the graph that there is an inverse relationship between the frequency range and the number of words which complies with Zipf's law (Li, 1992). After that, we started with the first set of 11 words which had the highest frequency range to be our list of stop words after removing all the sentiment words "جميل" (beautiful), "بشع" (ugly), etc.. , named entities like "فلول" (Followers), "مصر" (Egypt), "مبارك" (Mubarak), etc…, and verbs like "يحاكم" (Trial), "قتل" (kill), etc…, and tested its effect on the accuracy of the classifier. At the beginning there were drops in performance reflecting that there might be some important words that should not have been removed, or there exist some other stop words that still need to be removed. So we worked on identifying these words manually. Then, this process continued accumulatively by adding lists from the

[10] http://www.ranks.nl/resources/stopwords.html
[11] http://arabicstopwords.sourceforge.net

following frequency ranges until we reached a list of stop words consisting of 128 words that increases the accuracy by almost 1.5%. Figure 3.2 shows the frequency of each stop word listed in appendix B.

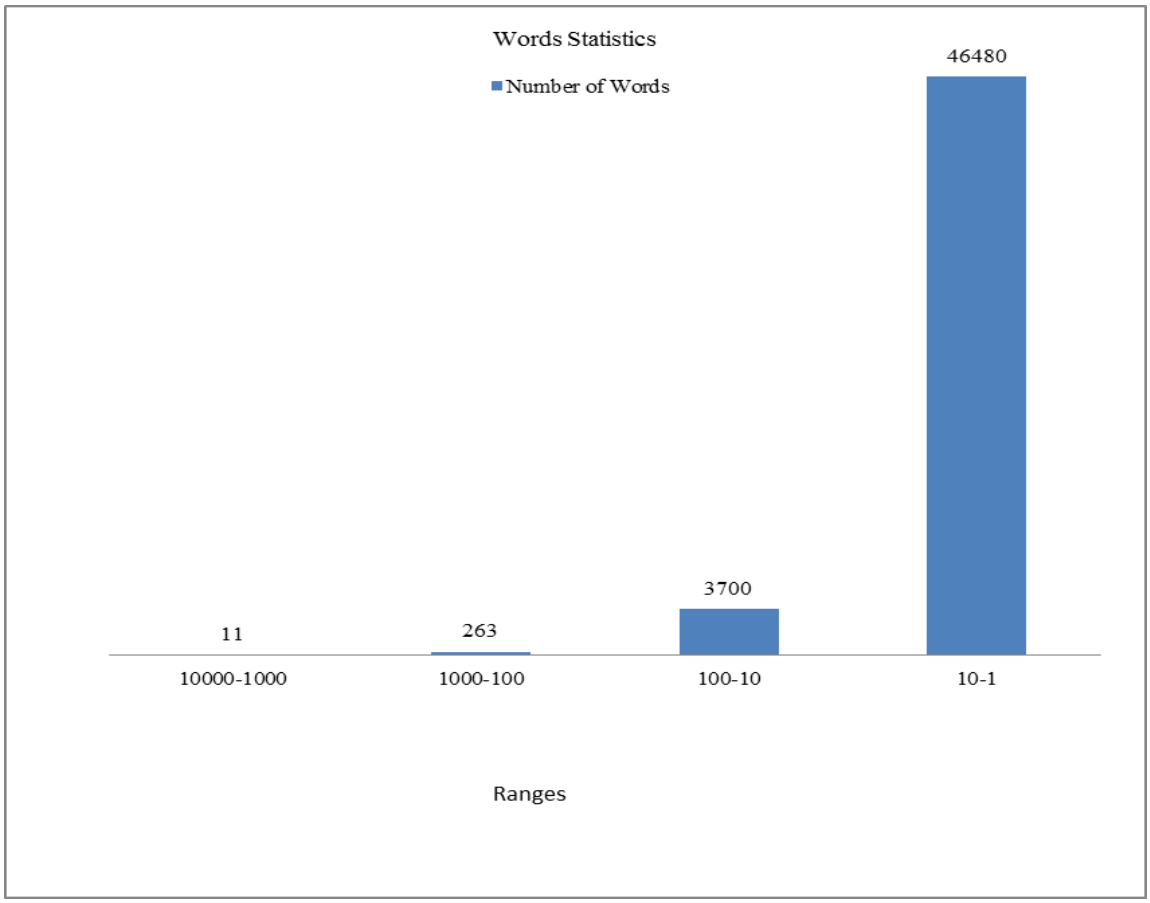

Figure 3-1: The Number of Words in Different Frequency Ranges

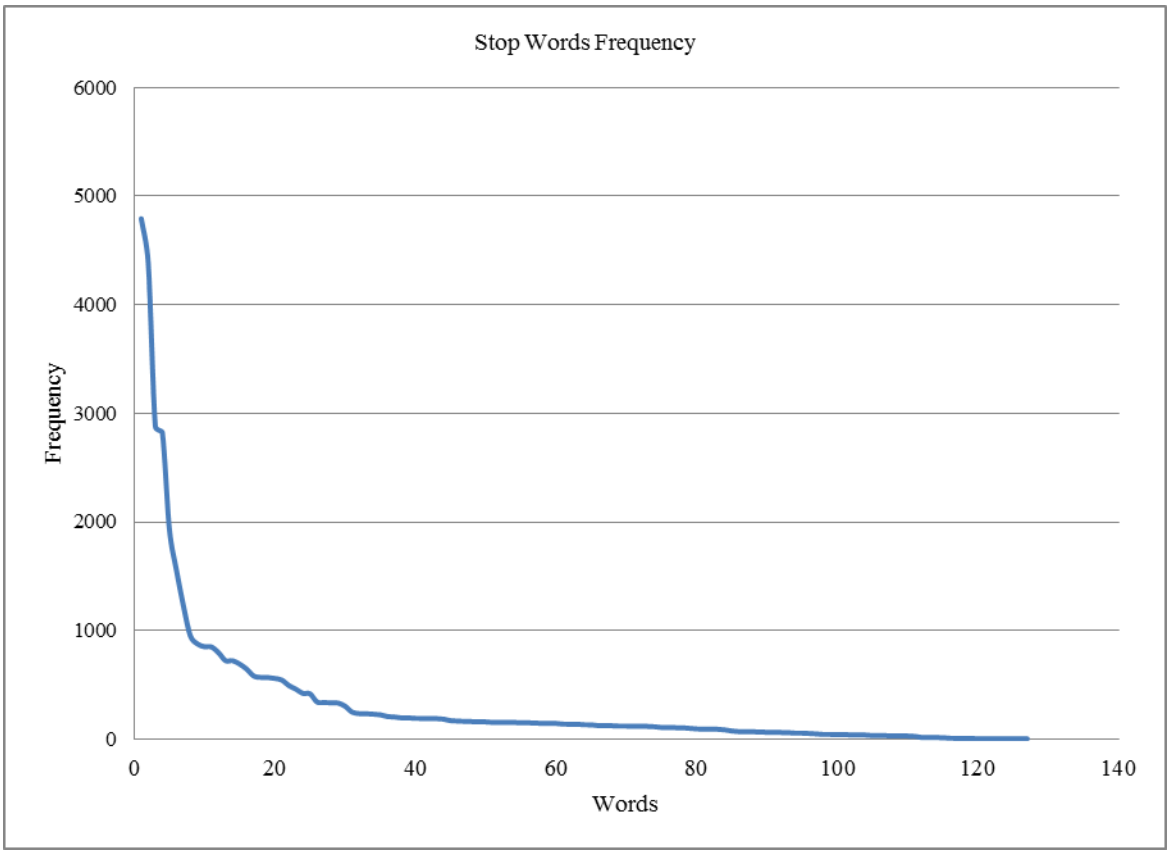

Figure 3-2: The Frequency of Each Stop Word

CHAPTER 4

PROPOSED APPROACH FOR ARABIC SENTENCE SENTIMENT ANALYSIS

The aim of this research is to create a kind of hybrid approach that can be used in the classification of Arabic text, especially in the Egyptian dialect. This will involve incorporating some of the semantic features together with some of the language features whose performance has been tested and proven to be important in the sentiment analysis process. This approach is employed to build a supervised model which is considered a more accurate model to use, rather than using an unsupervised model utilizing one or more of the sentiment lexicons. However, the unsupervised approach will also be investigated.

This chapter is organized as follows: section 4.1 explains in details the ML methodology, section 4.2 then fully describes the SO methodology, section 4.3 clarifies how these two methodologies are combined together, and finally section 4.4 lists the evaluation measures followed in order to evaluate our proposed approaches.

4.1 The ML Methodology for Sentence-level Sentiment Classification

The methodology used for building the ML classifiers consists mainly of 5 steps: 1) crawling tweets from Twitter to form a corpus; 2) cleaning this created corpus and manually annotating 4,800 tweets (1,600 positive tweets, 1,600 negative

tweets and 1,600 neutral tweets); 3) preprocessing these cleaned tweets (normalizing, stemming and removing the stop words); 4) identifying unigrams, bigrams, and trigrams to be used as candidates features in building the feature vectors for the annotated tweets; and 5) developing and testing the most known classifier used in sentiment classification, SVM and NB. Based on the analysis of the results obtained from the classifiers, we will go back to step 4 after changing the features used in building the feature vectors, and we will continue repeating these steps until we reach the most useful set of features to be used. The methodology phases are described in figure 4.1.

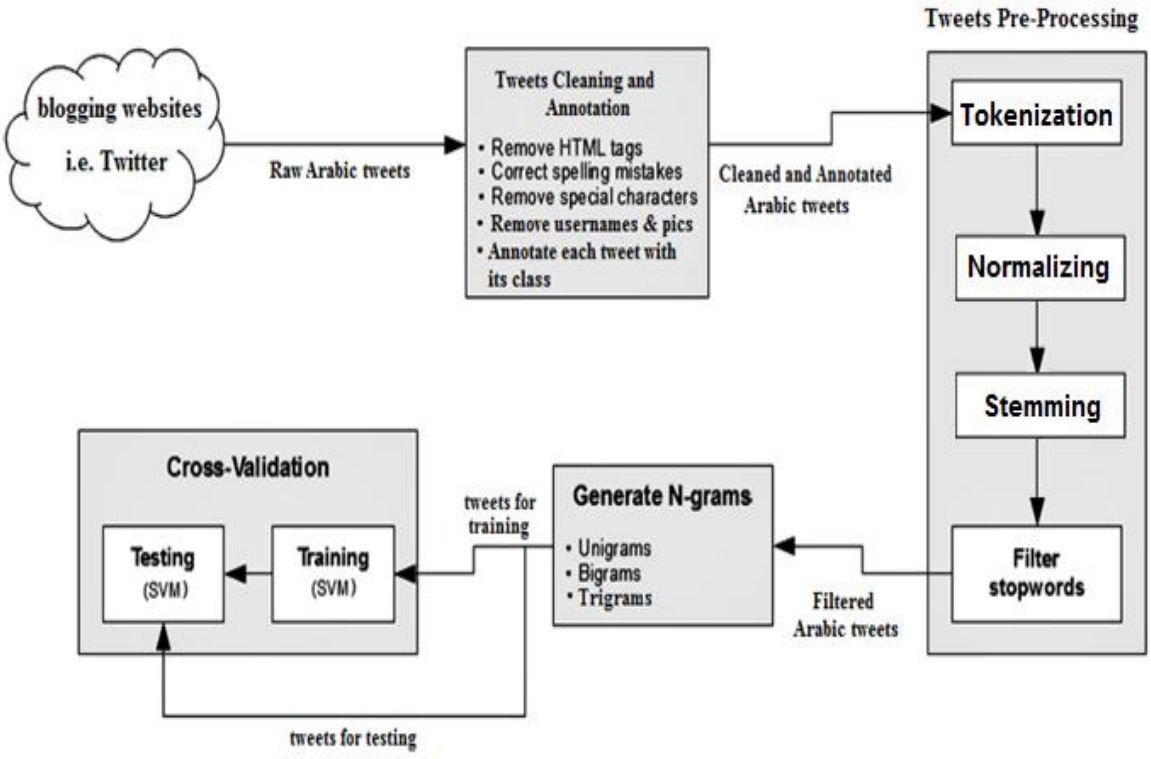

Figure 4-1: The ML Sentence-Level Sentiment Analysis Model

The model goes through the following phases:

4.1.1 Getting Data from Twitter (Arabic Tweets)

Although Arabic is considered one of the top 10 languages most used on the Internet according to the InternetWorld State[12] rank it is considered as a poor content language over the web unlike English with very few web pages specializing in Arabic reviews (Elhawary & Elfeky, 2010). We have searched for a source which communicates real opinions and at the same time the opinions are written in Arabic. For this reason, we have used Twitter's API[13] to get the required tweets. By setting the language to Arabic (lang=ar), we are now able to get Arabic tweets. Also, it was very important to get a large set of Arabic sentences in order for the classifier to be trained and be able to further classify any new supplied sentence. Twitter was one of the main sources for getting vast amounts of data as we got more than 20,000 tweets from different news topics.

4.1.2 Tweets Annotation and Cleaning

From the 20,000 tweets retrieved from Twitter, we have annotated 4,800 tweets consisting of 1,600 positive, 1,600 negative and 1,600 neutral tweets to be our training corpus. Table 4.2 shows samples of the annotated tweets. Two raters were used to determine the sentiment of the tweets. They had a high degree of agreement in

[12] http://wordnet.princeton.edu
[13] http://search.twitter.com/search.atom?lang=ar&rpp=100&page={0}&q={1}

their classification of the tweets, and for those tweets that disagreed; a third rater was used to determine its final sentiment. We have chosen the tweets that hold only one opinion, not sarcastic, and subjective. After annotating the tweets, we then went into the process of removing all user-names, pictures, hash tags, URLs and all non-Arabic words from the tweets to be easily manipulated and dealt with. Also, we have investigated the effect of the corpus size on the performance of the classifier.

Positive	إنت راجل محترم ..... انا صوتى لن يخرج من دائرة تضمك مع العظيم أبو الفتوح ياريت تتحدوا على رئيس و نائب.....جبهة لا تقهر You are a respectable person … I will definitely vote for you and Abu El Fotoh I hope you can challenge for president and vice president…unbeatable front
Negative	حرام عليك اللى عملته فينا وفى نفسك والله مصر ام الدنيا حرام عليك قوى This is over what you have done to us and to yourself Egypt is mother of the world this is really over

Table 4-1: Samples of Positive and Negative Tweets

4.1.3 Tweets Pre-Processing

Before extracting the feature vectors, preprocessing of the tweets is required. This involves tokenization (identifying the individual words and reducing the typographical variation), and then applying the proposed preprocessing mechanism (normalization, stemming, stop words removal) on the cleaned tweets. Tokenization

is easily carried out using one of the functions available in the NLTK[14] library. After that each process in the preprocessing mechanism is applied accumulatively to produce at the end normalized, stemmed tweets with the stop words removed.

4.1.4 Feature Extraction and Feature Vector

The feature vectors applied to the classifier consisted of the term frequency, as we are using statistical machine learning (Lee et al., 2002). Also, the different n-gram models were studied to analyze their influence on the classification problem. That is why we have chosen to work with unigrams, bigrams and trigrams as our work is on word/Phrase level sentiment analysis (Khreisata, 2009). Unigrams are considered the simplest features to extract and they provide good courage for the data, while bigrams and trigrams provide the ability to capture any negation or sentiment expression patterns. Therefore, the process starts by extracting all the unigrams, bigrams, and trigrams in the 4,800 annotated tweets. It is important to note that different features will be investigated including negation.

Then for each of these candidate features, its frequency in the 20,000 tweets was calculated, creating a dictionary for all the candidate features with their corresponding frequencies. Finally for each Tweet, if any of these candidate features was present in it, then this candidate feature frequency was fetched from the

[14] http://www.nltk.org/Home

dictionary and it was placed in the feature vector representing this tweet. Thus, for each tweet the following feature vector was constructed using term frequency:

({word1:frequency1, word2:frequency2 …}, "polarity")

We have used the frequency of the term in the 20,000 tweets to give more weight to those terms that appear more frequent in the corpus because these terms represent words and language patterns that are more used by the Arabic bloggers.

4.1.5 Training and Testing Classifiers

In this step, we have put the tweets into a format understandable by the classifier for maximum throughput. This involved generating the feature vectors and saving them in a sparse ARFF file which is the input file for the classifier. We chose the Weka suite software (Hall et al.., 2009) to run the classifier as it provides several ML algorithms such as SVM, NB and others. It also provides a number of test options, such as cross validation, test set and percentage split.

4.2 The SO Methodology for Sentence-level Sentiment Classification

The methodology used to build the SO classifier consists mainly of 3 steps: 1) using sentiment annotated tweets to extract the sentiment words, count the occurrences of each sentiment words in both positive and negative tweets and give each word two weights based on its number of occurrence in the positive and negative tweets; 2) preprocessing the tweets (normalizing, stemming and removing the stop words); and 3) classifying the 4,800 tweets (1,600 positive, 1,600 negative and 1,600 neutral) as

positive, negative or neutral using the sentiment word found in the tweet, and building a confusion matrix for the tweets classified as positive, another matrix for the tweets classified as negative, and another matrix for the tweets classified as neutral to measure the accuracy of classification.

4.2.1 Building the list of Sentiment Words and Smiley Faces

Given the limited work done for Arabic text in the field of sentiment analysis, especially for the Egyptian dialect, we had first to start by manually building two lists: one for the most occurring positive sentiment words, and one for the most occurring negative sentiment words. Then for each word in these lists a weight is given to it based on its frequency in positive tweets and its frequency in negative tweets. It is believed that as the lexicon size increases with different possible forms for sentiment words, the performance of the classification increases as more sentiment words will be recognized and used in calculating the semantic score. As for the smiley faces, there exists a list of the most well-known and used smiley faces, together with their polarities.

4.2.2 Tweets Pre-Processing

The preprocessing processes (normalizing, stemming, and stop words removal) were applied in the same order as in the ML approach. Both the tweets and the sentiment words list were processed.

4.2.3 Classifying the Test Set of Tweets

To determine the class of each tweet, a cumulative *score* was calculated using the sentiment words and smiley faces in the tweet to determine its class. For each sentiment word present, its score was added to the total in the following way:

$$score = \sum_{i=1}^{n} (w_{pi} + w_{ni})$$

Equation 4-1: The Semantic Score for Each Phrase

where w_{pi} is the positive weight of the word, w_{ni} is the negative weight of the word, and they are calculated based on the number of times this word appeared in the positive tweets, and the number of times this word appeared in the negative tweets. As for the smiley faces, their polarities were calculated in the same way as the sentiment words and their weights were added to the total score at the end.

The weights assigned to the sentiment words were used to determine how close it is to positive "1" or to negative "-1" or neutral "0". The final value of the score (score > 0, score < 0 or score =0) determines polarity of the whole tweet. It is important to note that the neutral class is defined as "no sentiment words were found or both numbers of positive and negative sentiment words are equal". Since, in this stage we are dealing with three classes, three binary classifiers were built: 1) positive classifier, 2) negative classifier, and 3) neutral classifier. Thus,

for each class a classifier was built determining whether the tweet belongs to its corresponding class, or it belongs to the class named "other". Then, the accuracy, the precision, the recall, and the F-measure of each classifier were calculated, which were averaged at the end to reach a final unified classifier.

For each class classifier, a confusion table was built displaying the number of correct and incorrect classifications made by the classifier compared with the actual classifications in the test data. The structure of the table is as follows: the rows present the number of actual classifications in the test data, and the columns present the number of predicted classifications made by the classifier. Thus, since we are only dealing with two classes, the size of the table will be 2-by-2.

4.3 Proposed Hybrid Approach

To take advantage of the benefits of each approach (ML and SO), we are proposing a hybrid approach for sentence-level sentiment analysis which combines both approaches. This approach involves building a classifier using the unigrams, bigrams, and trigrams with the optimum thresholds previously determined, together with adding a new feature for the SO score which sums the weights of all the sentiment words and smiley faces present in the tweet. Also unigrams, bigrams, and trigrams which are members in the sentiment features list their frequencies are multiplied by a factor (1/Net_Weight) to boost up their importance in the tweets. So now the feature vectors contain an attribute for the SO score as shown:

({word1:frequency1, senti_word2: factor*frequency2 …}, *SO*: score, "polarity")

It is important to note that negation is considered when it comes to calculating the SO score and the value of the sentiment features. If the tweet contains any of the negation words listed in Appendix D, the weight of the sentiment words following this negation word is multiplied by "-1" to switch its value.

4.4 Evaluation Measures

One method for comparing the performance of the classifying algorithms is the cross-validation method. It is a statistical method which splits the data into two parts: one part is used to train the model, while the other part is used to test the model (Manning & Schutze, 1999). The k-fold cross-validation form is considered the basic form of cross-validation, in which the data is first divided into k equally sized parts. Afterwards, k iterations of training and testing are carried out in way ensuring that in each iteration a different part of the data is used for testing, whereas the remaining k-1 parts are used for learning. In the experiments performed, 10-fold cross-validation (k=10) has been used to evaluate the classifiers' performance.

On the other hand, in most of the sentiment classification problems, three measures of classification effectiveness were widely used; which are: 1) the accuracy, 2) precision and 3) recall, which are explained in detail in (Sebastiani, 2002). We have used those three measures in addition to the F- Measure (F-score) to measure the accuracy of the test as it considers both the precision and the recall of the test in computing the score. These measures will help us to evaluate the performance of the

proposed prototype and measure the effectiveness of the suggested features set. Finally, within each algorithm (ML and SO) different models are created, and tests of significances are performed (Tan et al., 2005). These tests are used to evaluate the relative performance or the true difference between the models within each algorithm.

The measures are:

a) Accuracy:

The percentage of the correctly classified objects used to calculate the accuracy of a classifier is calculated as follows:

$$A = \frac{TP + TN}{TP + TN + FP + FN}$$

Equation 4-2: The Accuracy Equation

where:

TP (true positives) denotes the number of positively-labeled test sentences that were correctly classified as positive;

TN (true negatives) denotes the number of negatively-labeled test sentences that were correctly classified as negative;

FP (false positives) denotes the number of negatively-labeled test sentences that were incorrectly classified as positive; and

FN (false negatives) denotes the number of positively-labeled test sentences that were incorrectly classified as negative.

b) Precision:

The class's precision defines the probability that if a random sentence should be classified with this class, then this is the correct decision. Precision for the positive class for instance is calculated as follows:

$$P = \frac{TP}{TP + FP}$$

Equation 4-3: The Precision Equation

c) Recall:

The class's recall defines the probability that if a random sentence should be classified with this class, then this is the taken decision. Recall for the positive class for instance is calculated as follows:

$$R = \frac{TP}{TP + FN}$$

Equation 4-4: The Recall Equation

d) F-Measure:

The class's F- Measure defines the harmonic mean or the weighted average for both the precision and recall obtained. We have used the F1 measure, so that both the recall and the precision are evenly weighted. F- Measure is calculated as follows:

$$F = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

Equation 4-5: The F-measure Equation

e) Statistical Significance of Learned Models:

Comparing the performance for two models:

$$d_t = d \pm Z_{\alpha/2} \hat{\sigma}_d$$

Equation 4-6: The True Difference

where:

a) d is the absolute difference in the error rate and it is calculated as follows:

$$d = | e1 - e2|$$

Equation 4-7: Absolute Difference

where e1 and e2 are the error rates of the models

b) $Z_{\alpha/2}$ is the Z value of the level of confidence, which is the approximate value of the percentile point of the normal distribution used in probability and statistics (Rees, 1987). For large data sets, accuracy has a normal distribution. As shown in figure 4-2, the confidence level is equal to 1-α where α is the accepted error, and $Z_{\alpha/2}$ and $Z_{1-\alpha/2}$ are the upper and the lower bounds obtained from a standard distribution at confidence level 1-α (Tan et al., 2005). Given that normal distribution is symmetric around Z=0, thus $Z_{\alpha/2} = Z_{1-\alpha/2}$. Using the probability table, the Z value can be obtained for any confidence level. Table 4-2 shows the values of $Z_{\alpha/2}$ at different confidence levels.

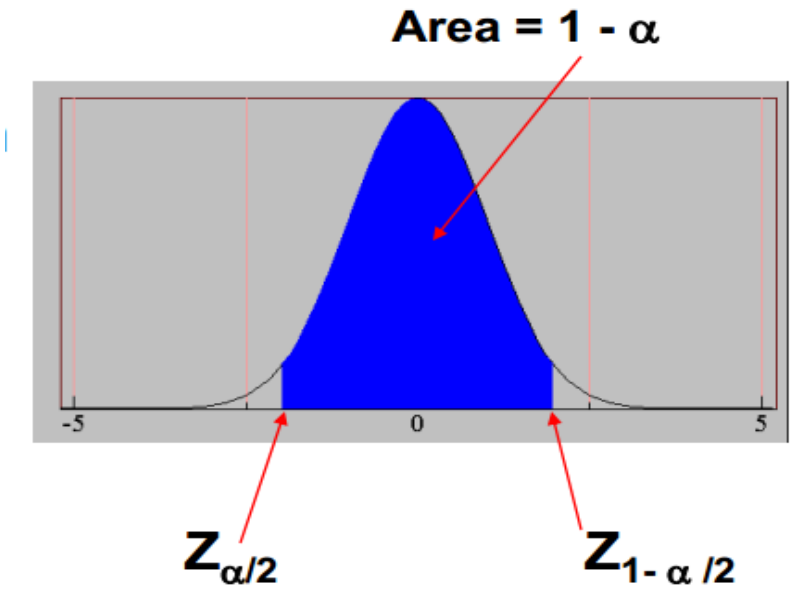

Figure 4-2: The Normal Distribution of the Accuracy

$1-\alpha$	0.99	0.98	0.95	0.9	0.8	0.7	0.5
$Z_{\alpha/2}$	2.58	2.33	1.96	1.65	1.28	1.04	0.67

Table 4-2: The Z Values at Different Confidence Levels

c) $\hat{\sigma}_d$ is the variance:

Given the assumption that n1 and n2 are sufficiently large, then e1 and e2 are approximated using normal distribution. Thus, d, error rate, is also normally distributed with mean d_t, and variance σ_d^2. The variance of d can be calculated as follows

$$\sigma_d^2 \approx \hat{\sigma}_d^2 = \frac{e1(1-e1)}{n1} + \frac{e2(1-e2)}{n2}$$

Equation 4-8: Variance for Two Models

where n1 and n2 are the size of the models, and e1(1-e1)/n1 and e2(1-e2)/n2 are the variance of the error rates.

The result of the significant test produces a range of values. If the zero value is in this range, then the difference between the models or the algorithms is not statistically significant. Otherwise, the difference is statistically significant at the specified confidence level.

CHAPTER 5

EXPERIMENTAL RESULTS AND EVALUATION

Although the ML approach is used extensively in the sentiment analysis process throughout the literature, it is still very important to test the SO approach with respect to our case which is dealing with the Egyptian dialect. This will be achieved in 6 steps: 1) suggesting a mechanism for preprocessing the tweets (normalization, stemming and stop-words removal) before processing them to reduce the noisy and unstructured nature of the text; 2) following our proposed methodology for the ML approach in section 4.1 which includes determining the optimum size of the corpus for training, together with the set of features to be used; 3) following our proposed methodologies for the SO approach in section 4.2 which includes building a semantic lexicon for extracting sentiment words, as well as calculating the semantic orientation; 4) comparing the classification results of each methodology; 5) combining those methodologies for the aim of creating a hybrid approach following the method in section 4.3 to benefit from the advantages of both approaches in classifying the sentiment of tweets written in Egyptian dialect; and finally 6) introducing a simple method for negation detection.

This chapter is organized as follows: section 5.1 compares the performance of ML classifiers and presents baseline results for the proposed ML approach, while section 5.2 defines baseline results for the proposed SO approach. Sections 5.3 and 5.4 show the effect of preprocessing on the performance of both ML and SO approaches. For the ML

approach, section 5.5 demonstrates the effect of corpus size on the accuracy of SVM classifier, whereas section 5.6 studies the optimum thresholds for the N-grams features used. For the SO approach, section 5.7 displays the results of using various sentiment words list on different corpus sizes. Finally, section 5.8 shows the results obtained after combining both the ML and SO approaches, and section 5.9 illustrates the results after considering negation on the SO and the hybrid approach.

5.1 Experiment-A: Comparing ML Classifiers Without Preprocessing

- **Objective**: The objective of this experiment is to compare the performance of the ML classification methods (SVM, Naive Bayes, Bayesian Network, Multilayer Perceptron, and Radial Basis Function Network) using unigram as features.

- **Method**: The training data set consists of 500 positive tweets and 500 negative tweets without preprocessing, and then the ML classifiers are tested using 10-fold cross validation method. It is important to note that the performance measures of both the positive and the negative classifiers were first calculated using the average of the 10-fold validations, then these measures were averaged to produce the numbers presented in the tables.

- **Results:** The results are shown in Table 5.1:

	SVM	NB	RBFN	Bayesian Network	MLP
Accuracy	**0.740**	0.691	0.675	0.567	0.499
Precision	**0.740**	0.692	0.679	0.567	0.499
Recall	**0.740**	0.691	0.675	0.567	0.499
F-Measure	**0.740**	0.691	0.673	0.567	0.478

Table 5-1: ML Results without Preprocessing

- **Discussion**

Comparing the results of the ML classifiers, it is clear that SVM has better results than other classifiers in almost all the evaluation measures. The improvement between the accuracy results of the top two models is almost 4.9% for SVM. The same goes with the precision, recall and the F-measure. Moreover, the results obtained by the SVM have been shown to be highly effective in sentiment analysis outperforming the results obtained by the other ML classifiers. SVM was applied successfully in several sentiment analysis tasks because of its principle advantages which include: "First, they are robust in high dimensional spaces; second, any feature is relevant; third, they are robust when there is a sparse set of samples; and, finally, most text categorization problems are linearly separable" (Rushdi-Saleh et al., 2011). By comparing the results obtained by

SVM in sentiment analysis in general, it is noticeable that SVM overcomes other machine learning techniques.

On the other hand, an important observation is that different forms for some feature were derived by the noisy nature of the text. The number of features used was 6622 which is considered a high dimensional feature space. This number of features might lead to distortions in the features space, decreasing the rate of the learning scheme and over-fitting of the data.

5.2 Experiment-B: Sentiment Classification Using SO Approach Without Preprocessing

- **Objective:** The objective of this experiment is to investigate the performance of the SO classifier.

- **Method:** The data set consists of 500 positive tweets and 500 negative tweets without preprocessing both the tweets and the sentiment words. The size of the sentiment words list used was 5,000 words.

- **Result:** The results are shown in table 5.2:

	Positive	Negative	Average
Accuracy	0.725	0.653	0.689
Precision	0.768	0.714	0.741
Recall	0.725	0.653	0.689
F-measure	0.746	0.682	0.714

Table 5-2: SO Results without Preprocessing

- **Discussion:**

By looking at the results calculated in table 5-2, it was clear that there were big numbers of tweets that were incorrectly classified. This behavior is caused by three problems: 1) the tweet originally contains no sentiment words, 2) the sentiment word in the tweet is not present in the lists, 3) the sentiment word in the tweet is written in a different form from the one stored in the list like "أشكر - thanks" and "اشكر – thanks". In meaning they are the same but here the prefix makes them two different words. One of the possible solutions to solve these problems is to consider different inflected forms of the sentiment words in the tweet as they might help in determining the tweet's semantic orientation. Another possible solution could be preprocessing the tweets and the sentiment words list in order to be able to extract the sentiment words and classify the tweets. However, looking at the performance measures calculated for the classified tweets, it was noticeable that they were somehow close to those of the ML approach. This fact results from the unstructured and the noise in the text which decreases the performance of the classifier to identify the class of the tweet.

5.3 Experiment-C: The Impact of Preprocessing on ML Classifiers

- **Objective:** The objective of this experiment is to test the effect of the preprocessing on the two machine learning algorithms (SVM and NB) which previously produced the highest performance.

- **Method:** We have applied our 3 stages accumulatively meaning that we will start by normalizing tweets, then stemming them, and finally the stop words will be removed from these stemmed tweets. Four experiments were carried out: 1) after applying the normalizer, 2) after applying our implemented stemmer, 3) after applying light stemmer[15], and 4) after removing the stop words. In each stage 10-fold cross validation method was used which averages the average of the 10-fold validations for each class.

- **Results:** The results are shown in tables 5-3, 5-4, 5-5 and 5-6

Performance Measures								
	Accuracy		**Precision**		**Recall**		**F-Measure**	
	NB	SVM	NB	SVM	NB	SVM	NB	SVM
Unigrams	0.695	**0.756**	0.696	**0.756**	0.695	**0.756**	0.695	**0.756**

Table 5-3: NB and SVM results using normalized tweets

[15] http://pypi.python.org/pypi/Tashaphyne/

Performance Measures								
	Accuracy		**Precision**		**Recall**		**F-Measure**	
	NB	SVM	NB	SVM	NB	SVM	NB	SVM
Unigrams	0.746	**0.774**	0.748	**0.774**	0.746	**0.774**	0.745	**0.774**

Table 5-4: NB and SVM results using stemmed tweets (1)

Performance Measures								
	Accuracy		**Precision**		**Recall**		**F-Measure**	
	NB	SVM	NB	SVM	NB	SVM	NB	SVM
Unigrams	0.73	**0.738**	0.731	**0.739**	0.73	**0.738**	0.731	**0.738**

Table 5-5: NB and SVM results using stemmed tweets (2)

Performance Measures								
	Accuracy		**Precision**		**Recall**		**F-Measure**	
	NB	SVM	NB	SVM	NB	SVM	NB	SVM
Unigrams	0.735	**0.777**	0.737	**0.777**	0.735	**0.777**	0.734	**0.777**

Table 5-6: NB and SVM results after stop words removal

Table 5-3, 5-4, 5-5, 5-6 show the results obtained after applying each process accumulatively. Tables 5-4 and 5-5 show the result of applying two stemmers: 1) our implemented stemmer, and 2) light stemmer. It is important to note that the performance measures of both the positive and the negative classifiers were first calculated using the average of the 10-fold validations, then these measures were averaged to produce the numbers presented in the tables.

- **Discussion:**

Comparing the results of NB and SVM, better results were produced after applying the preprocessing stages. The improvement between the best accuracy results before and after applying preprocessing for the NB is almost 4.4% and for the SVM is almost 3.7%. The same goes with the precision, recall and the F-measure. This behavior results from the fact that preprocessing usually tries to reduce the noise in the text, thus eliminating part of the distortions in the features space. Also an important observation is that the number of features was reduced dramatically from 6622 features in case of unigrams before applying preprocessing to 3220 features after applying preprocessing. That is because the more steps we apply from the preprocessing stage, the more related features converge together reducing the problem of features over-fitting and increasing the rate of the learning scheme.

We have tested our implemented stemmer against one of the light stemmers available. Analyzing the results in tables 5-4 and 5-5, it is noticeable that our implemented stemmer produces better results because of the dialect specific issues that we have addressed in our implementation. For example, the word "علشان" and "عشان" both forms of the words are right and they mean "because". In our stemmer we have included them in the irregular list so they won't be stemmed; however, in the light stemmer they will be stemmed to "علش-not a word" and "عش-hut" which are completely two different words.

Analyzing the results produced after stop words removal stage, it was clear that our developed stop words list needs to be further investigated. The results in table 5-6 showed that in case of SVM the performance increased by only 0.3%, while in case of NB the performance even decreased by 1.1%, which means that there are some other stop words that still need to be removed, or we have removed necessary words that shouldn't have been removed

In spite of the fact that preprocessing greatly reduces the noisy and unstructured nature of the text, yet still the noisy nature plays a major role in decreasing the performance of the experiment. For example, the word " وفي - loyal" after applying stemming will become the particle word "في - in" which is a stop word and will be removed from the tweet, thus the sentiment of the tweet will be lost. If we looked at these two tweets for example:

ألف مبروك لشعب البحرين **الوفي** الأحكام الصادره ضد أجرمو بحق الوطن والمواطنين

بعت دماء شباب مصر وحسبي الله ونعم الوكيل فيك وأولادك واحفادك **وفي** أمرك تبع أوامرك

they both contain the word "وفي", in which the first tweet "وفي" means "loyal" implying positive sentiment; whereas, in the second tweet "وفي" means "and in". Thus, in this case it will be wrongly increasing the positive weight of the tweet. Then, after stemming, the word would become "في" a stop word and it will be removed. Thus, the sentiment of the tweet will be changed.

5.4 Experiment-D: The Impact of Preprocessing on SO Classifier

- **Objective:** The objective of this experiment is to test the effect of the preprocessing on the SO performance.

- **Method:** 3 experiments were carried out accumulatively one at each stage with the preprocessing applied to both the sentiment words and the tweets. Before the experiments, we removed stop words since their removal should not have any impact on enhancing the results but will accelerate the classification process. In the first experiment we normalized both the tweets and the sentiment words, and then in the second experiment both were also stemmed. We didn't test the effect of stop words removal on SO performance as there is no intersection between the sentiment words and the stop words. Thus, removing the stop words won't affect the performance of the SO; it is only the sentiment words which affect it.

- **Results:** The results are shown in tables 5-7, 5-8, and 5-9:

	Positive	Negative	Average
Accuracy	0.728	0.658	0.693
Precision	0.767	0.711	0.739
Recall	0.728	0.658	0.693
F-measure	0.747	0.683	0.715

Table 5-7: SO results using normalized tweets

	Positive	Negative	Average
Accuracy	0.760	0.758	0.759
Precision	0.761	0.770	0.765
Recall	0.760	0.758	0.759
F-measure	0.760	0.764	0.762

Table 5-8: SO results using stemmed tweets (1)

	Positive	Negative	Average
Accuracy	0.753	0.755	0.754
Precision	0.758	0.763	0.760
Recall	0.753	0.755	0.754
F-measure	0.755	0.759	0. 757

Table 5-9: SO results using stemmed tweets (2)

Tables 5-7, 5-8, 5-9 calculate the performance results for the classification of the binary classifiers at each stage of the preprocessing. Tables 5-8 and 5-9 test the result of applying two stemmers: 1) our implemented stemmer, and 2) light stemmer.

- **Discussion:**

Regarding the effect of the preprocessing on the SO performance, we can note that there was an improvement of 6.5% in the accuracy and the recall, while there was an improvement of 2% in precision and 5% in the F-measure. That is because in SO it is only the form of the sentiment words which affect the performance. Thus, after preprocessing, the sentiment words in the tweets were almost converted to the same form of the sentiment words in the lists and they

were easily extracted. However, sentiment words represent a very small percentage of the words in the tweet, and also not all tweets contain sentiment words. Hence, building more comprehensive lists of sentiment words could be considered a possible solution to further enhance the performance.

Analyzing the results in tables 5-8 and 5-9, it is noticeable that both stemmers produce almost the same results with very minor changes. This behavior is somehow expected as the stemming of most of the sentiment words is expected to be the same because there are less dialect specific sentiment words.

Comparing the results of the positive and the negative binary classifiers, it is clear that the performance of the positive classifier is improving over the performance of the negative classifier until we applied the stemmer they started to become very close. This behavior reflects the fact that the positive tweets are less noisy than the negative tweets; therefore, with minimal preprocessing (just normalizing) it has almost reached the best result.

5.5 Experiment-E: The Effect of the Corpus Size on the ML Approach

- **Objective:** The objective of the experiment is to determine the optimum size of the training corpus to be used in the ML approach.

- **Method:** We have tried different corpus sizes in which 600 indicates that 200 positive tweets, 200 negative tweets and 200 neutral tweets were used, and so on. In each step we have added 200 tweets from each class until we have reached a

corpus of size 4800 consisting of 1600 positive tweets, 1600 negative tweets and 1600 neutral tweets.

- **Results:** The result is shown in figure 5-1:

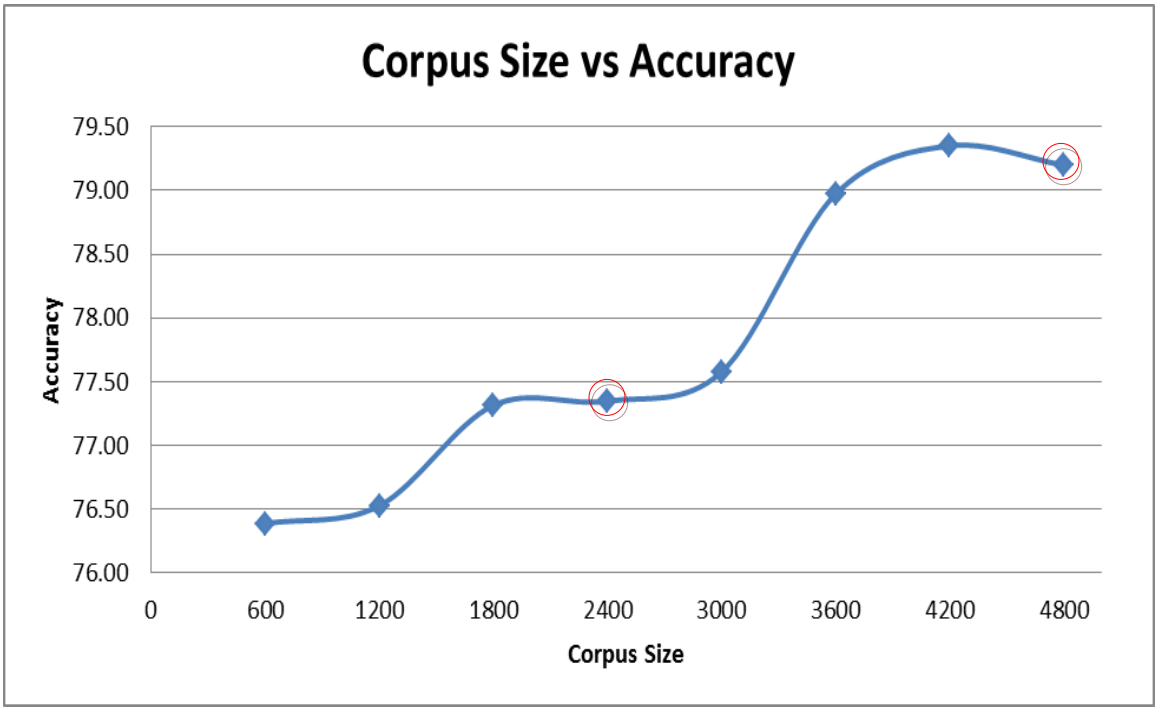

Figure 5-1: The Performance Using Different Corpus Sizes

- **Discussion:**

It is generally known that the size of the annotated corpus has a direct relation to the accuracy of the model generated; hence, the bigger the size of the corpus, the more accurate the model will be. However, it is rarely described how big the size of the corpus needs to be since factors like time, limited resources and cost appear to dominate decision-making about corpus size. Since the work

done on the Egyptian dialect is very limited, we had to figure out the optimum size of the training corpus to be used.

It is clear from the graph 5-1 that as the size of the corpus increases, the better the performance. However at the corpus size 2400, the accuracy remained constant, and at the corpus size 4800 the curve started to go downwards. Analyzing these 1200 added tweets, we have discovered that they were very sparse and short tweets about outdated topics like Gamal Abd el Nasser, Magles Askary, with their features rarely used in other tweets. That is why they distorted the learning process and caused overfitting problem.

Comparing the performance of 600 model with the 4800 model, we have performed a test of significance. To test if the performance difference is statistically significant, we have calculated the true difference (d_t) between the two models, assuming that the accuracy has a normal distribution:

So:

1) d is calculated as follows:

$$d = |0.236 - 0.208| = 0.028$$

2) At 95% confidence level, $Z_{\alpha/2} = 1.96$.

3) $\hat{\sigma}_d$ is calculated as follows:

$$\hat{\sigma}_d^2 = \frac{0.236(1-0.236)}{600} + \frac{0.208(1-0.208)}{4800} = 0.000335$$

Thus:

$$d_t = 0.028 \pm 1.96 \times \sqrt{0.000335} = 0.028 \pm 0.0359$$

The interval contains 0. Thus, the difference may not be statistically significant, which means that we need more tweets to create bigger model that could produce more statistically significant results at a 95% confidence level.

5.6 Experiment-F: The Optimum Thresholds for the N-grams Features in ML Approach

- **Objective:** The objective of the experiment is to determine the optimum threshold for each N-gram (unigrams, bigrams, and trigrams) feature in the ML approach using SVM classifier.

- **Method:** Using our new big corpus (4800 tweet) after applying the three preprocessing stages (normalization, stemming, and stop words removal), we have tried different frequency thresholds for each n-gram model separately as features. In which 0 indicates that all the n-grams were used, 1 indicates that n-grams greater than 1 were used, and so on. Following the observations we got from this step, we then went into the process of trying various combinations of the different n-gram models to further improve the performance of the classification process.

- **Results:** The results are shown in figure 5-2 and 5-3, and table 5-10:

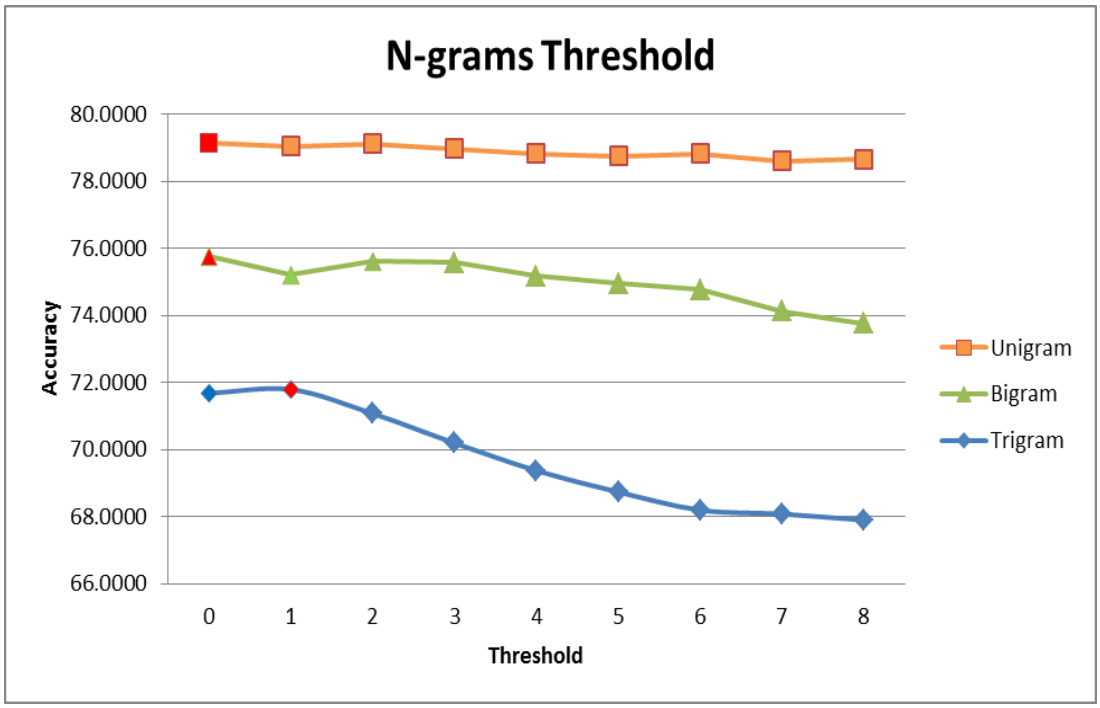

Figure 5-2: Different Frequency Thresholds for Each N-gram Model Separately

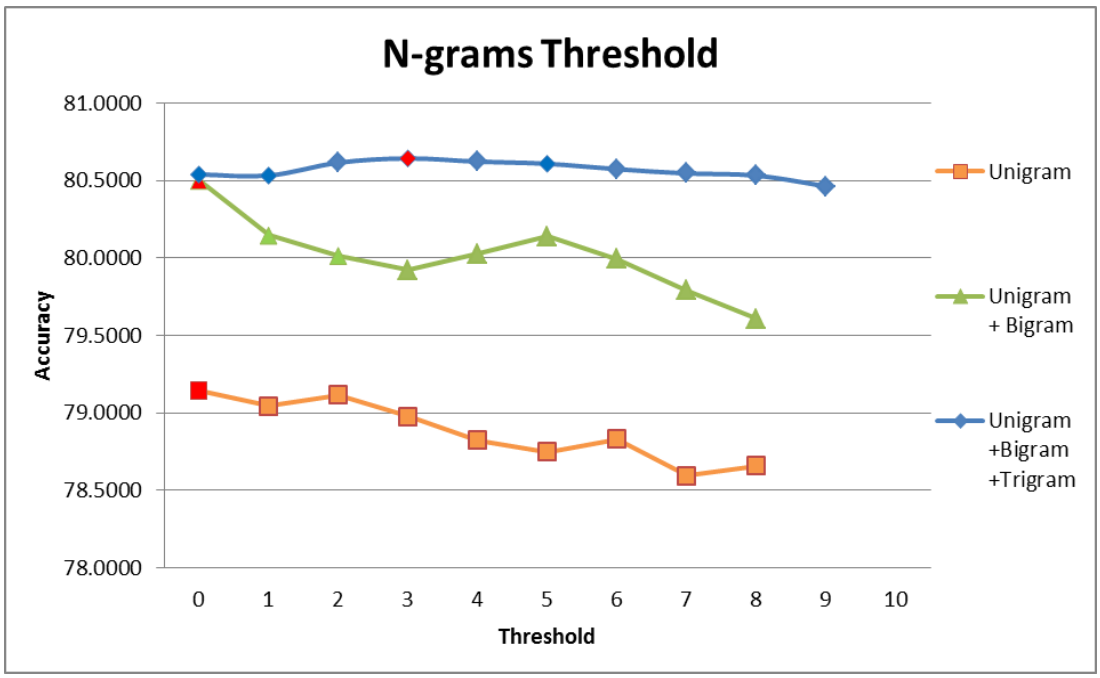

Figure 5-3: Different Frequency Thresholds for Combined N-gram Model

	Positive	Negative	Neutral	Average
Accuracy	0.819	0.761	0.796	0.792
Precision	0.819	0.756	0.794	0.790
Recall	0.819	0.761	0.796	0.792
F-measure	0.819	0.757	0.795	0.790

a. Unigrams

	Positive	Negative	Neutral	Average
Accuracy	0.839	0.776	0.800	0.805
Precision	0.838	0.770	0.796	0.801
Recall	0.839	0.776	0.800	0.805
F-measure	0.838	0.770	0.797	0.802

b. Unigrams and Bigrams

	Positive	Negative	Neutral	Average
Accuracy	0.840	0.778	0.800	0.806
Precision	0.839	0.772	0.797	0.803
Recall	0.840	0.778	0.800	0.806
F-measure	0.839	0.772	0.798	0.803

c. Unigrams, Bigrams and Trigrams

Table 5-10: SVM Results for the Combined Model

Figure 5-2 shows the accuracy of using different thresholds for each N-gram model separately. Since unigrams produced best results at threshold 0, we fixed this threshold for the unigrams. Figure 5-3 shows the accuracy of various thresholds for the combined unigram and the bigram model using the optimum

threshold for the unigram model got from graph 5-2. It is observable that bigrams produced best results at threshold 0 for the combined model. Then, the accuracy of different thresholds for the combined unigram, bigram, and trigram model was calculated using the optimum thresholds for the unigram and the bigram model previously obtained. Then Table 5-10 shows the performance measure using the optimum thresholds obtianed for each N-gram model.

- **Discussion:**

It is clear from the graph 5-2 that unigrams outperform bigrams and trigrams when determining the sentiment of the tweets. The results we have obtained using the SVM classifier is very similar to those obtained by Pang et al. (2002) as they have obtained 82.9% accuracy in case of unigrams. This behavior is due to the fact that unigrams are able to provide good coverage for the data, whereas bigrams and trigrams tend to be very sparse. Therefore, it is better to combine unigrams, bigrams and trigrams as features to improve the performance of the sentiment classification problem.

On the other hand, Figure 5-3 shows that adding the bigram model to the unigram model greatly improves the performance by 1.3%. However, there were not big differences in the performance by adding the trigram model to the combined unigram and bigram model. Bigrams and Trigrams are usually added to identify any repetitive patterns or expressions which might be associated with certain class. It should be noted that we have used only the 4800 annotated tweets

to build the unigram, bigram and trigrams models. Maybe using more tweets could result in more unigrams, bigrams and trigrams, thus improving the results.

5.7 Experiment-G: The Effect of the Sentiment Words Size on the SO Approach

- **Objective:** The objective of the experiment is to test the effect of using different sentiment words list's sizes with regards to the performance of the SO approach.

- **Method:** Different test corpus sizes were used in which 600 indicates that 200 positive tweets, 200 negative tweets and 200 neutral tweets were used, and so on. All the sentiment words in the 4800 corpus were extracted, and they were divided into 5 lists. For each test corpus size, the 5 sentiment words lists were used to classify the tweets. The sentiment words in the lists were given weights based on their frequencies in the test corpus size used.

- **Results:** The result is shown in Figure 5-4:

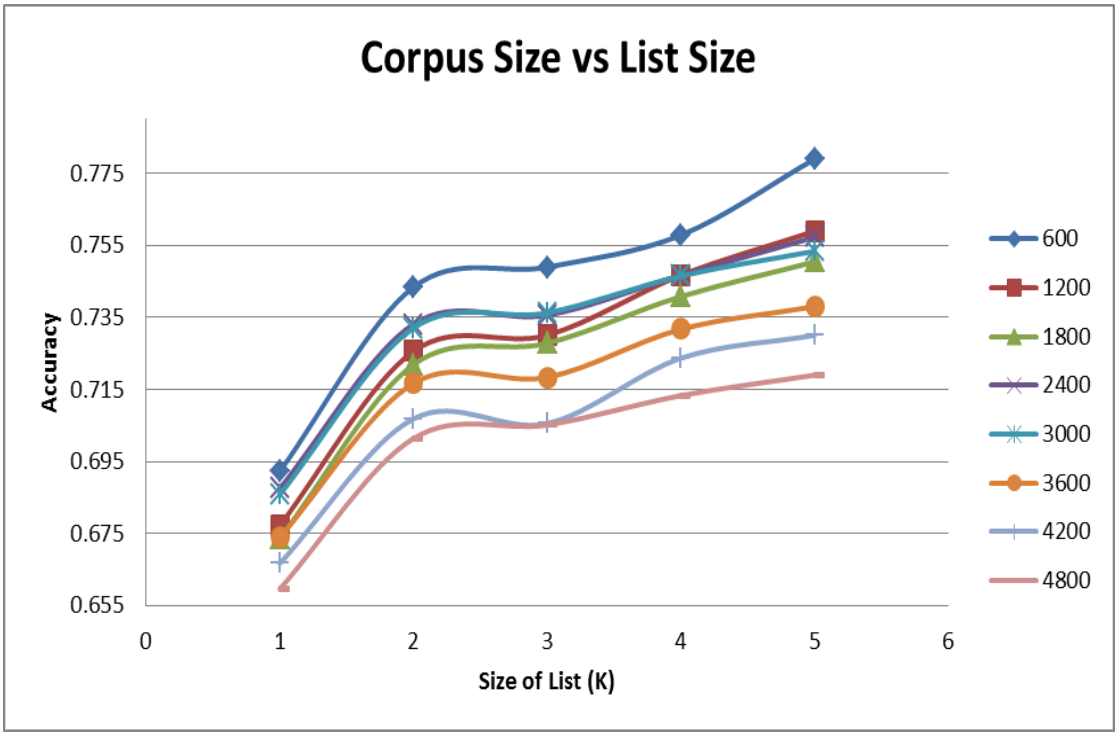

Figure 5-4: Using Different Sizes of Sentiment Words List

	Positive	Negative	Neutral	Average
Accuracy	0.765	0.715	0.676	0.719
Precision	0.739	0.685	0.622	0.682
Recall	0.754	0.694	0.603	0.683
F-measure	0.746	0.689	0.622	0.683

Table 5-11: SO Results

Figure 5-4 shows the accuracy of using different sizes for the sentiment words list for each corpus size, and Table 5-11 shows the performance measure using the most comprehensive list built for the corpus of size 4800.

- **Discussion**

It is clear from figure 5-4 that as the size of the sentiment words list increases the better the performance as more sentiment words were recognized, thus enhancing the classification process. Yet, it also reflects that we haven't reached the optimum size for the list as the curve hasn't reached saturation especially in case of the 4800 corpus, but it is kind of guidance as how large the list should be. It is important to note that, no matter what the size of the list is, it is essential to have as much as possible a comprehensive and up-to-date list which is evenly balanced including examples of all the sentiment words expected to be present in the test environment with their inflected forms if possible. Moreover, the figure shows that the smaller the size of the corpus and the bigger the size of the list, the better the results as the majority of the sentiment words will be identified, thus improving the classification process. That is why the corpus of size 600 produced better results than the corpus of size 4800, using the biggest list of words.

On the other hand, comparing the results obtained in the SO experiment with the ML experiment related to the accuracy measure, it is clear that the best result (0.806) obtained using the SVM learning algorithm in the ML approach improves on the average result (0.719) obtained using the SO approach. This improvement is almost 8.7% given that only one feature was considered in the ML experiment which is the frequency of n-grams in the corpus. Thus for ML, adding more features is expected to further improve the performance; however,

for the SO it depends only on the reliability of the sentiment words list to improve its performance.

To compare the performance of 4800 test corpus size with the smallest sentiment words list and with the largest sentiment words list, we have performed a test of significance. To test if the performance difference is statistically significant, we have calculated the true difference (d_t) between the two models, assuming that the accuracy has a normal distribution:

So:

1) d is calculated as follows:

$$d = |0.34 - 0.281| = 0.059$$

2) At 95% confidence level, $Z_{\alpha/2}$ =1.96.

3) $\hat{\sigma}_d$ is calculated as follows:

$$\hat{\sigma}_d^2 = \frac{0.34(1-0.34)}{4800} + \frac{0.281(1-0.281)}{4800} = 0.000088841$$

Thus:

$$d_t = 0.059 \pm 1.96 \times \sqrt{0.000088841} = 0.059 \pm 0.01847$$

The interval doesn't contain 0, so the difference may be statistically significant between the two models at a 95% confidence level.

5.8 Experiment-H: Sentiment Classification After Combining Both ML and SO

- **Objective:** The objective is to combine both approaches for the aim of creating a hybrid approach.

- **Method:** The ML and SO approaches are combined in 2 steps: 1) Adding the SO score as feature in the feature vector built using the ML approach; 2) each sentiment word found is multiplied by the inverse of its SO weight.

- **Results**: The results are shown in tables 5-12:

	Positive	Negative	Neutral	Average
Accuracy	0.844	0.783	0.801	0.809
Precision	0.842	0.777	0.797	0.805
Recall	0.844	0.783	0.801	0.809
F-measure	0.842	0.777	0.798	0.806

Table 5-12: Combining ML and SO

Table 5-12 shows the effect of adding the SO score as a feature in the feature vector and multiplying the sentiment words found by the inverse of their weight.

- **Discussion**

It is clear from the results in table 5-12 that combining both approaches (ML and SO) by multiplying each sentiment word by the inverse of its semantic weight and adding the SO score as a new feature has clearly improved the classifier's performance in terms of the accuracy, recall and F-Measure by almost 0.3%, while precision by 0.2%. This behavior is similar to the study by Kouloumpis et al. (2011) which showed that the best performance is produced using the *n*-grams together with the sentiment features. That is mainly due to the benefits taken from each approach: 1) the ML approach associates the combination of specific features and sentiment words to specific class; and 2) the SO approach gives sentiment words more weight and calculates a score which corresponds to the class of the tweet. For example, in the tweet:

علمتنا **نتفاءل** ، ودائما ما تفتح أمامنا طاقات نور **تحيى** فينا **الأمل** من جديد ، دمت لنا بألف **خير**

The positive sentiment words present in it like "نتفاءل", "تحيى", "الأمل" and "خير" were given higher weight to boost their presence in the tweet; also the tweet was given a positive score. Therefore, the combination of these features and SO score will be interpreted by the ML classifier to correspond to positive class. Thus, we can say that combining both approaches creating a hybrid approach is beneficial to correctly classify the sentiment of the tweets.

5.9 Experiment-I: Sentiment Classification After Adding Negation to Hybrid Approach and SO approach

- **Objective:** The objective of this experiment is to measure the effect of adding negation to SO approach and hybrid approach. ML approach is not tested because the sentiment words are not recognized in the feature vector; whereas in the hybrid approach the sentiment features are recognized by the SO approach.

- **Method:** Every sentiment word found in tweet is tested if it was preceded by a negation word. In this case, the weight of the word is multiplied by -1, otherwise its weight is not changed.

- **Results:** The results are shown in tables 5-13 and 5-14:

	Positive	Negative	Neutral	Average
Accuracy	0.761	0.715	0.676	0.717
Precision	0.750	0.684	0.622	0.681
Recall	0.736	0.694	0.602	0.682
F-measure	0.743	0.689	0.612	0.681

Table 5-13: SO and Negation

	Positive	Negative	Neutral	Average
Accuracy	0.845	0.782	0.800	0.809
Precision	0.844	0.776	0.797	0.806
Recall	0.845	0.782	0.800	0.809
F-measure	0.844	0.776	0.797	0. 806

Table 5-14: Hybrid Approach and Negation

Tables 5-13 and 5-14 show the performance measures after adding negation to the SO and hybrid approach.

- **Discussion:**

It is clear from the results obtained that considering sentiment words occurring after negation terms improved the classifier's performance in some cases, while decreased the performance in other cases, but on average it improved the performance of the hybrid approach by 0.1%. This behavior was observed in one of the studies which showed that ignoring the sentiment words following either contrast or conditional words, or even modal verbs decreased the classifier's accuracy in all their datasets with an average of 1%; however, there was a slight enhancement in the recall and the precision in some of their datasets (Morsy, 2011). The number of tweets which changed after considering negation is 207 tweets, representing only 4.2% from the total number of tweets used. These 207 tweets consist of 75 positive tweets, 92 negative tweets, and 40 neutral tweets. Samples from the tweets belonging to the negative class which were

earlier incorrectly classified as positive, but then after considering negation they were correctly classified as negative are:

الوهابيين وعبيدهم لا يعرفون الكرامة وبرغم حناجرهم فهم مخنثين لا **شرف** لهم

and

طلع عاليك سمعة مش **كويس** بسبب كلمة ديماجوجية دي اللهما بلغت اللهما فشهد

These tweets contain positive sentiment words like "شرف" and "كويس" which, before considering negation, increased the positive weight of the tweets and the tweets were mistakenly classified as positive. But then after considering negation, the weights of these words were multiplied by -1, thus increasing the negative weights of the tweets and the tweets were correctly classified as negative.

On the other hand, a closer look at the changed tweets reveals that when considering negation by multiplying weight of sentiment words by -1 is not subtle enough for natural complex languages like Arabic language. Some tweets can't be processed by this straightforward negation handling method. For example the following tweet

البرادعى توقعته صح **لا لعب** بلناس و**لا ضحك** عليهم فهي مبنية على خبرة طويلة. ربنا يحفظ ك لمصر ويحفظ اهلها

was from the positive class, but after considering negation it was wrongly classified as negative. That is because it contains the common phrases "لعب بلناس" and "ضحك على الناس", which usually have negative implications in spite of their

positive sentiment carrying word. Therefore, considering negation in this tweet would not be beneficial as their semantics are ignored. Thus, we can say that semantics consideration is sometimes required to correctly identify the sentiment of such tweets.

CHAPTER 6

CONCLUSION AND FUTURE WORK

Sentiment analysis has recently become one of the growing areas of research related to text mining and natural language processing. Research in sentiment analysis for the Arabic language has been very limited compared to other languages like English whether at the sentence-level or document-level. The main objectives of this research work were to explore the different approaches used in sentiment analysis, the effect of text preprocessing on the classification's accuracy, the impact of corpus size and features on the ML classification quality, the impact of semantic lexicon size on the SO classification quality, and finally the effect of creating hybrid approach combining both ML and SO approaches. The approach followed to achieve these objectives was: 1) suggesting a mechanism for preprocessing the tweets (normalization, stemming and stop-words removal) before using them to reduce the noisy and unstructured nature of the text; 2) proposing a methodology for the ML approach which includes investigating the optimum size of the corpus for training, together with the set of features to be used; 3) proposing a methodology for the SO approach which includes building a semantic lexicon for extracting sentiment words, as well as calculating the semantic orientation; 4) comparing the classification results of each methodology; and 5) combining those methodologies for the aim of creating a hybrid approach to classify the sentiment of tweets written in Egyptian dialect, together with a relatively simple method for handling negation.

Following our proposed approach, we started by building the feature vectors, using only unigrams as features from 1000 Arabic tweets (500 positive and 500 negative) written in Egyptian dialect from Twitter as our training data, to the SVM and NB classifiers for the aim of choosing the classifier with the higher accuracy. Also, we applied these 1000 tweets to the SO classifier and it was clear that ML approach outperformed over SO approach by 5% using only one feature (frequency of unigrams). Then, we demonstrated the effect of our preprocessing mechanism on the performance of the ML and SO approach using the same 1000 tweets. We have used two stemmers; our implemented stemmer and light stemmer in both approaches (ML and SO). In the ML approach, using our implemented stemmer improved all the performance measures of the SVM classifier by almost 3.7% and the NB classifier by almost 4.4%. While in the SO approach, using our implemented stemmer caused an improvement between 2-7% for the different performance measures. On the other hand, using the light stemmer decreased all the performance measures of the SVM classifier by 1.8% and the NB classifier by 3.5% in case of the ML approach, while in the SO approach there was an improvement between 1.5-6.5% for the different performance measures. This could be explained as a result of adding Egyptian dialect prefixes, suffixes and rules for broken plurals. Afterwards, we introduced the neutral sense in our training data set and started to investigate the effect of the training data size on the accuracy of the ML classifier (SVM) and we have reached the highest accuracy of 79.35% at the size of 4200 tweets, but then it decreased at size of 4800 to 79.20% because of the overfitting problem. A test of significance was carried between the 600 model and the 4800 model, but the resulting range showed that the difference between the two models is not statistically significant.

For the 4800 training data size, we worked on identifying the best threshold for each N-gram feature used. We found that the highest results were produced at threshold 0 for unigrams, threshold 0 for bigrams, and threshold 3 for trigrams. Similarly, we explored the impact of the size of the sentiment words list on the accuracy of the SO classifier, and it was clear that the bigger the size of the list the better the result regardless of the corpus size. A test of significance was carried between the 2 approaches (ML and SO), and the resulting range showed that the difference between the two approaches is statistically significant. Subsequently, we combined both approaches (ML and SO) proposing a hybrid approach for a sentence-level sentiment classification. This proposed hybrid system has used all the identified features from the ML approach, and the sentiment lexicon from the SO approach, resulting in an accuracy and recall of 80.9%, while its precision and F-measure is 80.6% which is much better compared to other systems. Finally, a negation detection method was incorporated with the proposed hybrid system, but there wasn't much improvement in the performance results as all the measures remained constant, except for the accuracy increased by only 0.1%.

Therefore, the main contribution of this research study is the proposal of a sentence-level sentiment classifier for the Egyptian dialect, which classifies a sentence whether a blog, review, tweet, etc… as holding an overall positive, negative or neutral sentiment with regards to the given target. This classifier uses a new feature set combining the ML and the SO features, as well as a simple method for negation detection. The results obtained by our proposed system showed better performance than other sentence-level sentiment classification systems using either ML or SO approaches.

There are different directions for extending this system. One direction could be further improving our developed stemmer by closely monitoring the performance of each applied rule, thus increasing the probability that more related words will be reduced to the same stems. Another direction for future work could be investigating how enlarging and improving the training data by incorporating hashtags, as well as positive and negative emoticons in our collection method could improve the accuracy. Both collection methods have proved to be useful (Kouloumpis et al., 2011), yet we have to determine which method results in the collection of better training data , or that the two collection methods are complementary to each other since the size and the accuracy of the training data has direct effect on the accuracy of the results produced. One more direction for future work could be building a more comprehensive list for the positive and negative sentiment words to enhance the performance of the SO approach; thus the classification quality of the hybrid approach, together with the negation detection method. Finally, given that this research work is part of a bigger system which also includes extracting hot topics, and identifying influential bloggers, one possible direction could be extracting the different topics in the sentence, then classify the sentiment of each topic separately.

REFERENCES

A. Abbasi, H. Chen, and A. Salem, "Sentiment Analysis in Multiple Languages: Feature selection for opinion classification in Web forums", *ACM Transactions on Information Systems (TOIS)*, vol. 26 (3), pp. 12, 2008.

R. Al-Shalabi, and R. Obeidat, "Improving KNN Arabic Text Classification with N-Grams Based Document Indexing", *In Proceedings of the Sixth International Conference on Informatics and Systems*, March 27-29 2008, *Cairo*, *Egypt.*

E.T. Al-Shammari, "A Novel Algorithm for Normalizing Noisy Arabic Text", *Computer Science and Information Engineering, WRI World Congress on, March 31 – April 2,* volume 4, pp. 477-482, 2009.

S. Baccianella, A. Esuli, and F. Sebastiani, "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining", *In Proceedings of the 7th conference on International Language Resources and Evaluation*, 25, 2010.

R. Duwairi, M. Al-Refai, and N. Khasawneh, "Feature Reduction Techniques for Arabic Text Categorization", *Journal of the American Society for Information Science and Technology*, vol. 60, number 11, pp. 2347–2352, 2009.

S. El-Beltagy, and A. Rafea, "An Accuracy-Enhanced Light Stemmer for Arabic Text" *ACM Transactions on Speech and Language Processing (TSLP)*, volume 7, number 2, Article 2, 22 pages, 2011.

M. Elhawary, and M. Elfeky, "Mining Arabic Business Reviews", *In Proceeding ICDMW '10 Proceedings of the 2010 IEEE International Conference on Data Mining Workshops*, IEEE Computer Society, pp. 1108-1113, 2010.

A. Esuli and F. Sebastiani, "Sentiwordnet: A publicly Available Lexical Resource for Opinion Mining", *In Proceedings of the 5th Conference on International Language Resources and Evaluation*, volume 6, pages 417–422, 2006.

N. Farra, E. Challita, R. Abou Assi, and H. Hajj, "Sentence-level and Document-level Sentiment Mining for Arabic Texts" *In Proceeding ICDMW '10 Proceedings of the 2010 IEEE International Conference on Data Mining Workshops IEEE Computer Society*, pp. 1114-1119, 2010.

D. Fradkin, and I. Muchnik, "Support Vector Machines For Classification" *Discrete Methods in Epidemiology, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 70, pp. 13–20, 2006.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten,"The WEKA Data Mining Software: An Update", *SIGKDD Explorations*, volume 11, issue 1, 2009.

A. Kennedy, and D. Inkpen, "Sentiment Classification of Movie Reviews Using Contextual Valence Shifters", *Computational Intelligence*, volume. 22, pp. 110–125, 2006.

L. Khreisata, "A Machine Learning Approach For Arabic Text Classification Using N-gram Frequency Statistics", *Journal of Informatics*, volume 3, Issue 1, pp. 72-77, 2009.

E. Kouloumpis, T. Wilson, and J. Moore, "Twitter Sentiment Analysis: The Good the Bad and the OMG!", *In Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pp. 538-541, 2011.

S. Larkey, L. Ballesteros, and M. Connell, "Light Stemming for Arabic Information Retrieval", *In Arabic Computational Morphology, A. Soudi, A. van der Bosch, and G. Neumann,* Eds, pp. 221–243. 2007.

B. Liu, "Sentiment Analysis and Subjectivity", *Handbook of Natural Language Processing*. Second Edition, (editors: N. Indurkhya and F. J. Damerau), 2010.

C. Manning, and H. Schutze, "Foundations of Statistical Natural Language Processing", *Cambridge, MIT Press. Cambridge, MA*, 1999.

Michelle de Haaff. Sentiment Analysis, Hard But Worth It!. *CustomerThink*, http://www.customerthink.com/blog/sentiment_analysis_hard_but_worth_it retrieved March 12th, 2010.

G. Miller, "Wordnet: A Lexical Database for English", *Communications of the ACM*, volume 38, issue 11. pp. 39–41, 1995.

S. Morsy, "Recognizing Contextual Valence Shifters in Document-Level Sentiment Classification", *Department of Computer Science and Engineering, The American University in Cairo (AUC)*, 2011.

J. Na, H. Sui, C. Khoo, S. Chan, and Y. Zhou, "Effectiveness of Simple Linguistic Processing in Automatic Sentiment Classification of Product Reviews", *In Conference of the International Society for Knowledge Organization (ISKO)*, pp. 49–54, 2004.

B. Pang, and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts", *In Proceedings of the ACL*. Cornell University, 2004.

B. Pang, and L. Lee, "Opinion Mining and Sentiment Analysis", *Foundation and Trends in Information Retrieval*, volume 2, pp. 1–135, 2008.

B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment Classification Using Machine Learning Techniques", *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 79–86, 2002.

D. G. Rees, *Foundations of Statistics, Why 95% confidence? Why not some other confidence level?"* CRC Press, p. 246, ISBN 0-412-28560-6, 1987.

M. Rushdi-Saleh, M. T. Martín-Valdivia, L.A. Ureña-López, and J. M. Perea-Ortega, "OCA: Opinion Corpus for Arabic", *Journal of the American Society for Information Science and Technology*, volume 62, pp. 2045–2054, 2011.

F. Sebastiani, "Machine Learning in Automated Text Categorization:, *ACM Computing. Survey*, volume 34, issue 1, pp. 1–47, 2002.

K. Shaalan, and H. Raza, "NERA: Named Entity Recognition for Arabic", *Journal of the American Society for Information Science and Technology*, volume 60, issue 8, pp. 1652–1663, 2009.

A. Shoukry, and A. Rafea, "Preprocessing Egyptian Dialect Tweets for Sentiment Mining" *In The Fourth Workshop on Computational Approaches to Arabic Script-based Languages*, pp. 47, 2012.

A. Shoukry, and A. Rafea, "Sentence-level Arabic Sentiment Analysis", *Collaboration Technologies and Systems (CTS), International Conference on*, pp.546-550, 2012.

P. Tan, M. Steinbach, and V. Kumar, "Classification: Basic Concepts, Decision Trees, and Model Evaluation", *Introduction to Data Mining*. 1 ed., volume 1, pp. 145-205. Boston: Pearson Addison Wesley, 2005.

P. Turney, "Thumbs Up or Thumbs Down?: Semantic Orientation Applied to Unsupervised Classification of Reviews", *In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pp. 417–424, Stroudsburg, PA, USA. Association for Computational Linguistics, 2002.

J. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin, "Learning Subjective Language", *Computational Linguistics*, volume 30, pp. 277–308, 2004.

T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis" *In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pp. 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics, 2005.

R. Xia, and C. Zong, "Exploring The Use of Word Relation Features for Sentiment Classification", *In Proceedings of the 23rd International Conference on Computational Linguistics*: Posters, COLING '10, pp. 1336–1344, Stroudsburg, PA, USA. Association for Computational Linguistics, 2010.

APPENDICIES

Appendix A: Samples of the tweets used in classification after removing stop words

I. Samples of positives tweets:

يامعالى رئيس الوزراء كنا نتمنى تلتقى

عاش شباب الثوره الشغل الجامد صفحه المجلس الجديده كومنتات ضد حكم العسكر

أوافق السيد عمرو موسى رؤيته للاامركزية وأعتقد الحل للكثير مشاكلنا

تحية للرئيس الراحل جمال عبد الناصر ذكراه اكيد تحيه ليه ماهي مراته

اول مرة اكتوبر بدون اختارناة اديها كمان حرية مصر تنتزع حريتها رغم الطغاة

شهداء الأمس شهداء مصر ويجب تودعهم مصر

كلنا مينا دانيال كلنا خالد سعيد لا اقصد شعار الوحدة بل اقصد نفس الشعار سيشعل الثورة جديد

مسلم مسيحي كلنا مصريين وساحمي أخوتي بدمي أيا دينهم اعتقادهم لو موافق

وفروا الوقت والاموال اعدموا شعب البحرين لتعيشوا وحدكم ويصفق لكم الغرباء

يارب اكتب الحريه لثوار المحاكمات العسكريه

II. Samples of negative tweets:

مبقاش بياكل الشويتين دول خلاص وفرهم لنفسك

جمال عبدالناصر أحترامى هوسبب مانحن وهومن رسخ فكرة القائدالملهم الديكتاتورالأوحدالذى يسعى الكل لرضاه

جمال عبدالناصر انشأ الدولة البوليسيه مصر بقيادة صلاح نصر واعوانه وقضى الاحزاب والحياة السياسية

يارب يزدهم شوطة تاخدهم هم وعسوي السلعوة

كمان اسجل اعتراضى كمان محتجه

ليه مصره أماني وجهه نظرك بس ممكن تكون صح حد ذاته غلط

آدى التطرف جهة أخرى استاذ كمال

بيتهيئلى استعداد ليوم 6 اكتوبر بدل قطع الاتصالات نبهدل الدنيا اية رئيك التحليل المغرض

جثتي تنجح انتخابات الرئاسة يبأة انتكاسة للثورة لو حصل ونجحت يقود حملة ضدك وجدت فل

لا لو لو يوفقوا للاختيار يقوم المجلس مختارها ازاااي طيب بثق العوا ومش فاهمه ليه موافق

Appendix B: The List of Stop Words with Their Corresponding Frequency

من 4793	دي 338	كنت 154	اى 100	عنها 34
في 4379	بين 335	هي 154	كلها 95	إنه 32
على 2882	انت 333	فيها 152	التى 92	اني 32
و 2821	أنا 304	عند 151	هى 92	معك 31
فى 1934	حتى 249	التي 149	دا 91	اننا 30
يا 1560	لما 236	الذي 146	انك 86	فيهم 29
عن 1228	فيه 234	قال 145	وهو 76	د 26
مع 949	هذا 231	هذه 145	ومن 70	انتا 16
ان 877	واحد 225	قد 139	منك 69	عنك 15
هو 852	احنا 209	انه 138	نحن 68	وهى 15
علي 849	اي 205	ريتويت 138	زى 66	معا 14
ما 796	كده 197	بعض 133	أنت 64	آن 9
اللي 723	إن 196	أول 132	انهم 63	انتي 8
كل 723	او 192	إيه 125	معانا 62	وأنت 7
بعد 692	أو 191	الآن 124	حتي 60	وإن 6
ده 645	عليه 191	الان 124	وانا 56	ومع 5
اليوم 582	ف 189	أي 121	عنه 56	وعن 4
أن 569	دى 185	منذ 121	منه 53	معاكم 4
يوم 568	مين 171	؟؟ 120	إلي 49	معاكو 4
انا 560	الي 167	عليها 120	ونحن 44	معاها 4
via 544	كانت 164	له 118	وانت 44	وعليه 4
إلى 494	أمام 162	ال 115	منكم 43	وانتم 4
كان 461	زي 160	تم 108	وان 42	وانتي 4
ايه 422	يكون 159	ب 106	معاهم 40	
اللى 418	خلال 155	دة 106	معايا 39	
الى 343	ع 154	عليك 105	وأنا 38	

Appendix C: Samples from the Lists of Sentiment Words

Positive Sentiment Words

احترام	اعتذار	فخر	استقبال	براء	حب
اشكر	اعنك	فخور	اسف	برافو	حبه
تحي	اقدر	فرح	اشجع	برنس	حبي
حقيق	امل	فضل	اشرف	بطل	حترم
حري	انجاز	قدير	اصلح	تحرير	حتفل
حمي	انصر	كريم	اصلي	تزم	حره
خير	تحالف	كسب	اصيل	تضامن	حسم
رحم	تحياتي	مءمن	اطمءن	تغن	حسن
مءيد	ترحيب	محترم	اعانكم	تفق	حفظكم
بارك	تقبل	معجب	اعتدل	تقد	حل
بحب	جامد	منضم	اعجاب	تقدير	حلال
حلو	جدير	نتصر	اعز	تمبع	حلوة
سلام	حذو	واضح	اعظم	تمني	حمد
ضحك	حر	وافق	اعن	تهنء	حمية
طاهر	حفظ	وفر	افد	توفيق	حميد
غالي	حق	اثق	افن	ثبتكم	خلاص
قوي	حلم	احسن	القاء	ثقت	خلق
مجهود	زغرط	احفظ	اللة	ثقه	خلي
مخلص	سلم	احلي	الله	جد	دليل
مصلح	شجيع	اخف	امان	جدع	دمت
مناسب	شرف	اخير	امن	جزيل	ديك
وفق	شريف	ادب	انبسط	جمع	ديمقراطي
اءيد	شكر	اذهل	انتصر	جنة	راءع
احب	صح	ارحب	انجح	جنت	راجل
احترم	طيب	ارحم	انشاء	جنته	رب
اسعد	عين	ارع	بجد	حاسم	ربنا

هنء	نتفاءل	مثبر	فل	صراح	رحمت
هني	نجح	مثلي	فوز	صريح	رزق
هنيك	نحب	مجد	قاءد	صفء	رضا
واعد	نحترم	مجيد	قمر	صفق	زاد
واعي	نزيه	محرب	قيمة	ضمير	زعيم
وحشت	نسيم	محروس	كرم	عاش	سالم
ورا	نشجع	مسامحين	كفا	عبقري	سامح
ورد	نصركم	مستنين	كنان	عتذر	سريع
وطن	نضيف	مصارح	كويس	عجب	سعاد
وعي	نعم	مصداقي	لاءق	عدل	سعدة
يارب	نفع	معالي	لاقي	عدلة	سعيد
يجا	نقدر	مكافا	لتقي	عزنا	سننتتصر
يرض	نكر	مكن	لجنة	عسل	شامل
يسر	نهض	ممتاز	لله	عظيم	شجع
يسمح	نور	منقذ	لوح	عقبال	شقيق
ينر	هايل	موافق	مءيدي	عقل	شهيد
	هدن	موهوب	مبروك	غنء	صاحب
	هدي	نبي	مبني	فسيح	صدق

Negative Sentiment Words

تعذيب	بجاحة	ارعبو	مءامر	خيانة	قبض	قرف
تهد	بشو	استقيل	مات	زندق	قطع	حرق
ثاءر	بعت	اسر	مجزر	سخري	محتاج	خرب
ثقيل	بقش	اسرا	مخلوع	سرق	مشكل	دهس
جبر	بلء	اسفوخص	مستكتر	شحت	مفتري	اضراب
جث	بلاش	اسقط	مضرب	شذذ	نقد	رفض
جرحي	بلاغ	اسلام	نفخ	شرموط	ابحي	سجن
جري	بلوك	اسير	نيل	شيطان	اتعذب	ضيع
جريمة	بهيم	اشكي	وسخ	ضعف	اختلاف	طرد
جزم	تاخر	اصدم	ضد	طورء	استقال	لعب
جعنة	تباط	اصعب	ابحة	ظلم	اسف	لعن
جلاد	تحرش	اضرب	ابطاء	عار	اسوا	معرص
جهل	تحرق	اضربو	ابكي	عبء	اسود	ناقص
حثلة	تحمل	اعدم	اتحرق	عيان	اعتراض	هزم
حذري	تخف	اغب	اتخدع	عيب	انسحاب	ولع
حرار	تخوين	اغبيا	اتعدم	غباء	تربي	اتفو
حرب	ترجي	اغبياء	اتلهي	غبي	تهم	اعتصام
حرش	تردد	امو	اتهد	غلبة	جحم	بلطجي
حرض	ترويع	انتحر	اجرم	غلط	جوع	حزن
حرقو	تسب	انتكاس	اجهاض	فاسد	حاكم	حسب
حستي	تسلط	انزف	احا	فسد	حرامي	خيب
حسم	تشجر	انسحب	اخبط	فشل	حسد	ذبح
حمل	تصعيد	انقذو	اختل	فض	حشيش	زعل
حمير	تضر	انقض	اختلف	فلل	حقير	سقط
خازوق	تضيق	اهانة	اخرس	قتل	خايف	سكت
خاين	تظاهر	اهمد	اخص	قواد	خرف	ضرب
خبط	تعب	اهنه	ادع	كفاي	خطا	فتن
خبطو	تعديل	بايع	ارحم	كلب	خوف	فشخ

نشيل	مرتزق	فوضي	علوقي	صرخ	زعلان	خرا
نفاق	مزبل	فيتو	عنصر	صعب	زفت	خروف
نفجر	مزور	قبر	عنف	صمت	زننه	خزي
نفض	مستفز	قتحم	عنيفا	صنم	زهق	خسارة
نقص	مسروق	قحب	عواجيز	صهيون	سافل	خست
نمو	مشفق	قديم	عوض	صيان	سامح	خسر
نوم	مضده	قمام	غريب	ضايع	سبها	خسران
هاكر	مظاهر	قمع	غصب	ضحي	ستر	خسير
هاكرز	معكس	قنبل	غلطان	ضدك	سحل	خطة
هبط	مفقود	كارتون	غنيم	ضريب	سخن	خفي
هجص	مقيد	كارث	غور	ضلالي	سطحي	خلاف
هجوم	مكرو	كذب	غوغاءي	طاغي	سفلت	خلع
هدل	مكسور	كس	غيب	طراطير	سفيه	خمورج
هرب	ملة	كسح	فات	طز	سلبي	خول
هروب	ملت	كميم	فاش	طعم	سلط	خون
هماجي	ملعون	كمين	فاشل	طغا	سلعو	داهي
هيج	منافق	لاذع	فاض	طغيان	سم	دبابا
وحش	منفطس	للاسف	فرق	طيط	سيس	ديك
وساخ	منكر	ماساو	فرم	عاتقين	شاذ	ديكتاتوري
وضيع	موت	متعسر	فصل	عامي	شتيم	دياصور
وعد	موقوت	محتل	فضح	عبط	شعل	ذءب
وغد	ميت	محير	فقر	عبيط	شغب	ذل
يخف	ميسر	مختلف	فقير	عتذر	شل	رخيص
يدس	نتزع	مخدر	فل	عدو	شوط	رعع
يسمح	نتقم	مخلف	فلط	عرص	شوي	رميت
يطق	ندالة	مدبر	فلق	عشواءي	صارم	رهق
	نزف	مدرع	فني	عفنة	صحي	رهيب
	نسحب	مراحيض	فهم	عك	صراع	زحم

Appendix D: The Lists of Negation Words

دون

بـ غـ ير

بـ دون

لا

ما

لـ ن

مش

غـ ير

عدم

عديـ م

لـ يس

لـ م

لـ ست

لـ يست

لـ سنا