Strategies for Sentiment Analysis and Classification of Non English Tweets

by

Fahad Almuqhim

A Project Report Submitted
in
Partial Fulfillment of the
Requirements for the Degree of
Master of Science
in
Computer Science

Supervised by

Dr. Rajendra K. Raj

Department of Computer Science

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York

April 2016
Abstract

Strategies for Sentiment Analysis and Classification of Non English Tweets

Fahad Almuqhim

Supervising Professor: Dr. Rajendra K. Raj

The increased use of social network services has led researchers to study several activities in these services. Nowadays, Sentiment Analysis is considered as one of the common tasks in data mining and text classification. This would help in several areas of research, such as politics and marketing. People from all around the world use Twitter, which is one of the most popular microblogging services, thus a very large number of non-English tweets are sent every day, including opinions about news, political decisions, or any other trending topics. This project presents an approach of classifying the sentiment of non-English tweets. The Arabic language is used as a case study to apply the presented strategies. Three classification models were implemented in this project: a model extracted using only the term-document matrix, a model extract using the term-document matrix and the sentiment scores of the documents, and a two-stage classification approach. The results of each model are analyzed and discussed in detail.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>2</td>
</tr>
<tr>
<td>1.1.1 Data Mining</td>
<td>2</td>
</tr>
<tr>
<td>1.1.2 Classification Algorithms</td>
<td>2</td>
</tr>
<tr>
<td>1.1.3 Text Classification</td>
<td>3</td>
</tr>
<tr>
<td>1.1.4 Sentiment Analysis</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Problem Statement</td>
<td>6</td>
</tr>
<tr>
<td>1.3 Related Work</td>
<td>6</td>
</tr>
<tr>
<td>1.4 Hypothesis</td>
<td>7</td>
</tr>
<tr>
<td>2 Project Design</td>
<td>9</td>
</tr>
<tr>
<td>2.1 Problem Statement</td>
<td>10</td>
</tr>
<tr>
<td>2.2 Data Collection</td>
<td>10</td>
</tr>
<tr>
<td>2.3 Data Cleaning and Preprocessing</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Classification Algorithm</td>
<td>13</td>
</tr>
<tr>
<td>3 Implementation</td>
<td>14</td>
</tr>
<tr>
<td>3.1 Data Cleaning and Preprocessing</td>
<td>14</td>
</tr>
<tr>
<td>3.2 Calculate Sentiment Scores</td>
<td>19</td>
</tr>
<tr>
<td>3.3 Classification Algorithm</td>
<td>21</td>
</tr>
<tr>
<td>4 Analysis</td>
<td>25</td>
</tr>
<tr>
<td>4.1 Model 1: Cleaned tweets with no sentiment scores</td>
<td>26</td>
</tr>
<tr>
<td>4.2 Model 2: Cleaned tweets using sentiment scores</td>
<td>27</td>
</tr>
<tr>
<td>4.3 Model 3: Cleaned tweets using sentiment scores classifying using a two-stage approach</td>
<td>27</td>
</tr>
<tr>
<td>4.4 Comparison between a single-stage approach and a two-stage approach</td>
<td>28</td>
</tr>
</tbody>
</table>
5 Conclusions .................................................. 30
  5.1 Current Status ............................................. 30
  5.2 Future Work ................................................. 30
  5.3 Conclusion .................................................. 31

Bibliography ................................................. 32
List of Tables

3.1 The number of tweets of each class in the dataset ........................... 23
4.1 The results of the first model that uses tweets with no sentiment scores ... 27
4.2 The results of the second model that uses tweets and the sentiment scores 27
4.3 The results of the third model that uses tweets and the sentiment scores classifying using a two-stage classification ............................ 28
4.4 The results of the t test for the two models using their accuracies ......... 29
4.5 The confusion matrix of one of the iteration in the second stage of the two-stage approach ......................................................... 29
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The flowchart of the proposed project</td>
<td>10</td>
</tr>
<tr>
<td>3.1</td>
<td>The steps of the cleaning and preprocessing process</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>The changes of a tweet after each step in the cleaning process</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>A snapshot of the sentiment lexicon in Arabic</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>A snapshot of our dataset after adding sentiment scores</td>
<td>20</td>
</tr>
<tr>
<td>3.5</td>
<td>The flowchart of the classification model that uses sentiment scores to ex-</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>tract support features</td>
<td></td>
</tr>
<tr>
<td>3.6</td>
<td>How a given tweet can be classified in the two-stage approach</td>
<td>23</td>
</tr>
<tr>
<td>3.7</td>
<td>The flowchart of the two-stage approach</td>
<td>24</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

With the rise of social media services, there is a need to take advantage of their use. An enormous number of online posts need to be analyzed and understood to extract useful data, including non-English posts. Nowadays, Sentiment Analysis is considered as one of the common tasks in data mining and text classification, which is a process of identifying the opinion or the emotion of a given text. This process would help in several areas, such as politics and marketing. There are many online services, however, Twitter is one of the most popular microblogging websites that allows users to post their opinions with a limit of 140 characters per post (i.e. per tweet). Twitter handles a huge amount of data every day. These tweets may include opinions about news, political decisions, or any other trending topics. Studying these tweets would help to understand the opinions of these users.

Due to the complexity of the Arabic language, it is chosen for our research as a case study of applying strategies for classifying sentiment analysis. Moreover, the huge amount of Arabic data on Twitter provides the motivation for dealing with this language. According to the Dubai School of Government [6], in 2014, there were 5.7M Arabic active users in Twitter, with 42% from Saudi Arabia. These users were posting 17M tweets per day. Thus, the increased use of social networking services in Arabic countries leads researchers to apply sentiment analysis to classify where a given text is positive, negative or neutral. This classification would identify the opinion of a trending topic from different areas in the Arabic world, such as politics, sports, education etc. For example, with a trending topic about a new governmental decision, this process would understand the feedback on this decision.
There are several published papers that discuss sentiment analysis for English; however, there is a lack of classifying sentiment for other languages, especially for Arabic. In this project, a strategy to classify sentiment analysis will be discussed. This classification strategy will be developed to classify opinions from Arabic text. This project will help to accurately classify given tweets as positive, negative or neutral. It can be seen that this classification model may be used for other languages to help in several applications in many countries from economy to industry. While industrialists could understand the needs of their customers to improve their services, the stock market could be predicted by analyzing the feedback of the users on stock market companies.

1.1 Background

1.1.1 Data Mining

Data mining is a process to analyze a huge data to extract useful information. Many applications can be developed in this field. The process of data mining is to analyze a large amount of data to find useful correlations and connections [16]. An example of extracted useful information is that an online user’s activities could predict the user’s interests. In this example, the prediction is done using big data consisting of other users’ activities and their interests. There are several data mining algorithms, which include classification algorithms, regression algorithms and association algorithms. Our project problem is a classification problem since it predicts an opinion of a given text.

1.1.2 Classification Algorithms

The classification algorithm, known also as statistical classification, is to predict a category of a given input based on its attributes. This classification is considered as supervised learning. It usually has a method called classifier that classifies a new instance using a trained model by already categorized instances [16]. Many packages and libraries were developed in order to implement the classification algorithms which include Support Vector Machine
(SVM), Naive Bayes, and Logistic Regression. Support Vector Machine is used in this project since it is a common classification algorithm for text classification and sentiment analysis problems.

1.1.3 Text Classification

Text classification is a process to predict the category of a given document. Each document can belong to one or more categories. This task can be implemented using a machine learning technique, which is to train a model using already categorized documents to predict unseen documents [11]. The applications of this process would include sentiment analysis, spam filtering, and readability assessment. The task of text classification starts by transforming the document from text into a required representation for the machine learning algorithm [11]. The representation that is used by SVM is a sparse matrix of the document-term matrix. This matrix is computed using the frequency of terms in a collection of documents. The rows in the matrix consist of the documents in the collection, and the columns consist of terms. There are several schemes to define a document-term matrix. A common way is to use TF-IDF, which is term frequency inverse document frequency.

1.1.4 Sentiment Analysis

Sentiment analysis is a text classification problem, and it is a task to identify the opinion of a given text based on already classified text. While the opinions correspond to the categories, the texts correspond to documents. Sentiment analysis is tasked to extract the sentiment using Natural Language Processing (NLP) [14]. It can answer the question of whether a given text is positive, negative or neutral. Two approaches can be used for sentiment analysis. While the first approach is the machine learning approach, the second approach is the lexicon-based approach. These two approaches will be discussed in the next two subsections.
Machine Learning Approach

The machine learning approach relies on pre-annotated tweets. The annotation could be positive, negative or natural. Then, this annotated dataset will be used for building the classification model. When there are no annotated tweets, the only way is to label the tweets manually. There are many ideas to build a tool for reviewing tweets manually. For example, a two-stage process should be developed, which means that every tweet is reviewed twice by two different native speakers. If the two opinions (positive, negative, or natural) are equal, the tweet is annotated and ready to be used for the study. If the opinions are not the same, the tweet will go back to be reviewed again or another native speaker could break the tie. However, this process may take time, and it needs people who are able to review each tweet manually. There is also a paid service provided by CrowdFlower\(^1\). They allow to submit the dataset. Then contributors who are native speakers of the submitted dataset’s language will review the data and annotate them manually. There are some published papers which use datasets of manually annotated Arabic tweets, such as the paper by Duwairi et al. [8] and that by Nabil et al. [12].

The difficulties in this approach is to collect tweets from trending topics, and then clean these collected data from duplicate (e.g. retweets), and reply posts (e.g. mentions). Then, these tweets will be manually reviewed to be rated as positive, negative, and neutral. After the annotation process, the dataset should be cleaned to be processed in the classification model. Cleaning steps for Arabic texts would be similar to other languages; however, there are some difficult processes since it is considered to be a morphologically rich language [15]. Users in social media websites use different dialects, idioms and compound phrases. In order to deal with different dialects, the dialects should be mapped together since they may have the same meaning. Thus, this mapping would help to ensure good performance in the classification method. Because of the lack of availability of Arabic dialect lexicon, developers should build their own lexicon to deal with this concern. However, this lexicon can be extracted from the annotated dataset. Pak et al. [13] suggests building the

\(^1\)http://www.crowdflower.com
sentiment lexicon from labeled text. When a lexicon of a language is not available, the lists of positive and negative words can be retrieved from an annotated dataset. While words with high frequency in positive text are considered in the positive dictionary, words with high frequency in the negative texts are considered in the negative dictionary. After this extraction, this list can be used as a seed list, adding synonyms of the extracted dialect words. This list could be used to map between the dialect words.

The collected data should contain opinions of different subjects, such as politics, sports, and business. Due to the limitation of Twitter API, only tweets that are less than seven days old can be queried. Thus, this limitation can stop researchers from reviewing trending topic history. Also, collected data may contain spam tweets that could affect the classification. These tweets will then be manually reviewed to be rated as positive, negative, and neutral. This process requires plenty of time. Due to people’s different opinions, it cannot be certain as to whether the sentiment of a given tweet is positive, negative or neutral. However, this method could be improved by relying on three or more persons’ opinions.

The dataset that were collected by Nabil et al. [12] is available for academic use. Thus, this dataset will be used in this project to build an effective tool for sentiment analysis of Arabic tweets. This dataset will be preprocessed and cleaned by different steps and ideas to improve the accuracy of the classification.

**Lexicon-Based Approach**

The lexicon-based approach relies on lists of words that consist of positive and negative words and phrases. The sentiment score of a given text is determined by using this lexicon. Some steps should be taken while determining the sentiment score. For example, it cannot be said that a given sentence is positive because of a positive adjective since sometimes a positive adjective follows a negative symbol, which changes the opinion of the sentence. In Arabic, persons’ names may mean a positive word in the language, which would affect the meaning of the sentence. For example, ”Saeed” is an Arabic name which means ”happy”, so this name may affect the sentiment score of the sentence. To solve this issue, an algorithm of extracting Arabic names should be developed to remove Arabic names before determining the sentiment score of the sentence [9]. There are some letters in Arabic that
are written in several ways for the same letter, such as “ل ل ل”. These different ways should be replaced with the base form of this letter, which is in this case “ل”. This procedure is called normalization for Arabic words.

The difficulties of this approach would include the lack of available sentiment lexicon for the targeted language. In Arabic, a large sentiment lexicon was collected by Badaro et al. [7]. This lexicon is available for academic use, and it will be used in our project to improve the classification model of the machine learning approach. This lexicon has sentiment for words in the Modern Standard Arabic. Thus, the difficulties would also include building dictionaries of dialect words, idioms and compound phrases.

1.2 Problem Statement

Data mining is a process to analyze a huge data to extract useful information. Many applications can be developed in this field. The area of the project is to address several strategies for sentiment analysis. The Arabic language will be used as a case study to apply these strategies. A tool that analyzes Arabic text to identify the opinion of the text will be implemented and tested in this project. The reason for using Arabic language is that Arabic language is considered a morphologically rich language which makes it very difficult to automatically work with [15]. These difficulties generate several challenges in dealing with Arabic text, which include different Arabic dialects, informal sentence structure, idioms and compound phrases. These challenges can be found in other languages as well. Thus, this project will address these challenges in detail and present strategies for dealing with them in general to generate an approach for solving sentiment problem of any language.

1.3 Related Work

There are several published papers that have been done for sentiment analysis of Arabic text. While some of them used the machine learning approach, other were used the lexicon-based approach.
In the work of Farra et al. [10], three dictionaries were used for sentiment analysis. These dictionaries consist of positive, negative and neutral words. The sentiment analysis is extracted by stemming the words of the sentence, and then determining the polarity of the sentence based on the dictionaries. If a word in the sentence is not found in any of the dictionaries, the user will be asked to classify this word. Then the word will be added to one of the dictionaries based on its classification by the user.

The work by Duwairi et al. [8] was done by developing a machine learning approach. Their work started by collecting tweets from different subjects on Twitter. Then, they annotated the tweets manually as positive, negative and neutral. They collected and annotated more than 25K tweets. After cleaning the dataset, a classification model was built in different classification algorithms, which include Support Vector Machine and Nave Bayes. The results were 71.68% for SVM and 76.78% for NB.

In the paper by Nabil et al. [12], 10K tweets were collected and annotated manually. They collected the data first using trending topics, then the data were annotated by the three authors. If a tweet has two or more of the same labels from the authors, it will be moved to the dataset, otherwise the tweet will be deleted. There are several methods in the machine learning approach used in this paper. The best recall that they got was 69% by SVM. However, this paper did not mention any preprocessing or cleaning steps.

Taking a chance to see what other languages are facing in sentiment analysis, Pak et al. [13] tried to identify the sentiment of French reviews of video games. They first extracted sentiment lexicon in France by using already labeled tweets as positive or negative. They extracted the most frequent words in these labeled tweets, and then the words were added to the corresponding class in the lexicon. This lexicon was used for sentiment analysis of video game reviews in France and the accuracy was 71.59%.

1.4 Hypothesis

There are an enormous number of posts on the Internet. As a data mining perspective, this content should be studied to be summarized into useful knowledge. The aim of this project
is to present a way for classifying sentiment analysis of non-English posts. The project will deal with Arabic language as a case study. This classification could help to understand the opinions and emotions of a given text or topic. The implementation of this classification model will have some steps and challenges that will be addressed as a base strategy of classifying sentiment analysis for non-English posts. The result of the sentiment analysis should help in many areas, including business and economy.

Several published papers could identify sentiment analysis of given texts in different languages. However, the hypothesis underlying this project is to present a way of implementing a text classification for sentiment analysis of any particular language. Also, the sentiment lexicon will be used to extract the feature vectors of the classification model in the machine learning approach. This use of the lexicon would improve the accuracy of the classification.
Chapter 2

Project Design

In sentiment analysis, the implementation should start with a good way of collecting data. Since there are two approaches can be used for sentiment analysis, the data collection differs from each approach. When the machine learning approach is chosen, the data should consist of reviews written in the selected language, and they are annotated as positive, negative or neutral. The reviews are better if they consist of different topics, such as politics, sports, and economy. These topics could help to see the users’ opinions of various views. Thus, it could help to identify the emotion of any given text. On Twitter, there is a free API to retrieve tweets that match a given criteria\(^1\). However, there is a limitation when collecting the data using their free API. Tweets that were posted more than seven days ago cannot be retrieved. Thus, old trending topics cannot be studied for sentiment analysis unless there is an enterprise access to regain old tweets. When the dataset is not labeled correctly for sentiment analysis purposes, the annotation process should be done by native speakers of the reviews’ language. Each review should be labeled by three or more different speakers to increase the correction of the annotation process. It would take time to examine all the dataset using this process. There is an enterprise company called CrowdFlower\(^2\), which can collect and label the data by contributor. They also allow submitting any collected data to be labeled by them. In our case, the dataset may contain tweets, and the task is to annotate each tweet as positive, negative, or neutral.

\(^1\)https://dev.twitter.com
\(^2\)http://www.crowdflower.com
2.1 Problem Statement

The wide use of social network services leads researchers to identify the sentiment of the posted documents. This identification would help in many areas from politics to the economy. This project will discuss some strategies for classifying sentiment of posted tweets on Twitter. The approach of this project can be applied to classify the emotions of tweets in different languages. The Arabic language is used as a case study to apply the proposed approach. The flowchart of our project can be seen in Figure 2.1.

2.2 Data Collection

In this project, we are using the Arabic language as a case study for classifying sentiment analysis of given tweets. The data that we are using were collected and annotated by Nabil et al. [12]. Each tweet is labeled as positive, negative, neutral or objective. This data has 10K tweets; however, half of this dataset is under the objective class. For this study, we
may focus on subjective tweets only, so objective tweets may be disregarded.

2.3 Data Cleaning and Preprocessing

In the data preprocessing and cleaning, we believe that some typical steps should be implemented for various languages, such as removing characters that are not in the alphabet of the particular language. To analyze any language text, first, any characters that are written in different scripts should be replaced with a base form script because some scripts have different hexadecimal numbers which will affect the classification process. Then, any characters that are not in the language should be deleted, such as URLs and symbols (e.g., -, #, and $). Also, emoticons are not considered in any official language, so, these emoticons should be replaced with whether meaningful words in the selected language or text that describe the emoticons. Then, the text normalization would take a place to normalize the letters by removing diacritics, and take the base form of each letter if possible, such as converting letters to the lower case form. When the text is normalized, a tool that analyzes each word to check if the word is correct or misspelled. Also, it could check if the word belongs to the standard dictionary of the language or the dialect dictionary. This list may also contain words written without using white spaces since users in Twitter sometimes write the tweets without using white spaces if possible because of the limitation of only 140 characters per tweet on Twitter. Then, a tool should be implemented to analyze this list of words that are not in the standard dictionary to extract meaningful words, such as splitting combined words. In this process, a seek dialect list could also be obtained. This list could be expanded by adding synonyms of dialect words. After that, the stop words, which are words that are most common words in the languages, should be removed while keeping negative symbols. When the stop words are removed, the dataset should be ready for building the classification model.

In our project, as we are dealing with the Arabic language, the previously explained steps were implemented to preprocessing and cleaning our dataset. The steps are run in the following order:
Replacing characters and emoticons: In the Arabic language, some characters can be written using different scripts which are represented in the computer systems by using separate hexadecimal. In this process, these characters should be replaced with a base form of the character. Then, each emoticon is superseded by described text in Arabic.

Text normalization: The process of normalizing the Arabic text includes replacing similar letters with a base letter, for example, replace ”أ , أ , أ” by ”أ”. It also removes diacritics from the words since diacritics do not change the sentiment of the word in the Arabic language, for example, eliminate the diacritics from ”مَسْتَع” to have ”مَسْتَع” which means ”enjoyable” in English. Then, the normalization can remove tatweel from the words, which is a way to expand the word in the writing style, for example, it changes ”مَسْتَع” to ”مَسْتَع”. This process is done by using a Python library called ”Tashaphyne”, and it is implemented by Taha Zerrouki [5].

Extract Arabic symbols and conjunctions: In the Arabic language, some conjunctions and symbols can be written connected to the word without using a space. For example, ”and” is written following the word with no space in between.

Removing any non-Arabic character: each tweet goes through a regular expression to make sure that only Arabic characters are kept, for example, removing URLs and symbols (e.g., -, #, and $).

Extract words that are not in Modern Standard Arabic: a tool called ”AraMorph” [2] implemented by Tim Buckwalter was used to extract words that are not in Modern Standard Arabic. These words could be misspelled words, dialect words, combined words (i.e., two words connected without space). This list is analyzed to extract
meaningful words, such as correcting misspell a word, and splitting combined words. The dialect words are used to build a sentiment lexicon for dialect words. The sentiment of the words is assigned using the frequency of each word in each class, which means that words appearing in positive tweets more than negative tweets is considered as a positive word and vice versa. This list is used to build a dictionary of dialect words by manually adding synonyms of found words. This dictionary can be used to add synonyms in the tweets before building the classification model.

- Extract the support features of the classification algorithm using both the term-document matrix and a sentiment score of the tweet: This extraction would improve the accuracy of the classification since the sentiment score may cover words that have not been trained by the classification model. The sentiment score is calculated based on a sentiment lexicon.

- Remove Arabic stop words: The Arabic stop words were removed and the tweets are ready for building the classification model.

### 2.4 Classification Algorithm

In the classification phase, we choose to implement a machine learning algorithm, and the model is built using the document matrix and the sentiment score of the document. This section will be explained in detail in Section 3.3.
Chapter 3

Implementation

The implementation of this project went through three stages: data cleaning and preprocessing, calculating the sentiment score for each tweet, and implementing the classification algorithm. The data collection is not mentioned in this chapter since the dataset was already collected and annotated by Nabil et al. [12]. Each tweet in the dataset is labeled as objective, positive, negative or neutral. The dataset consists of 10K tweets that are stored in a text file. Each line of the text file consists of a tweet and its label.

3.1 Data Cleaning and Preprocessing

The Java and Python programming languages were used for the cleaning and preprocessing process. These languages were chosen because of the ability to deal with the Arabic language, and the available libraries and packages for Arabic text.

Data cleaning and preprocessing was done in several steps in order to get the dataset cleaned and ready for the classification algorithm. Figure 3.1 shows all the steps and their order for the cleaning process. These steps should be run in the same order as shown in Figure 3.1.

![Figure 3.1: The steps of the cleaning and preprocessing process](image-url)
**Replace characters and emoticons**

In Arabic languages, some characters can be written using different hexadecimal representations. These characters should be replaced with the base representation in order to get a better matching in the classification process. Moreover, the emoticons express the emotion of the sentence. Thus, these emoticons should be handled and kept for sentiment analysis. In order to keep these emoticons, the emoticons are replaced with a meaningful words that describe them. For example, smiley face is replaced with 'smiley face’ in Arabic. This replacement could give the same form for any used smiley face, which would also help in the building process of the classification model.

To make these replacements, a Python script is written using some regular expressions. It basically scans all the text file of the dataset and looks for any characters that need to be replaced to the base form of this character. It also looks for any emoticons that could help in understanding the sentiment of the tweet and replaces these emoticons with meaningful words in the Arabic language.

**Normalize Arabic words**

For the normalization process, there are several cases that should be handled in order to normalize the Arabic text, which would include normalizing characters to get the base form, eliminating diacritics and removing expanding from the words.

In the Arabic language, some characters can be written using different forms. For example the letter 'A' in the Arabic language can be written in more than one way. In this case, these characters "أ،ي،يَ" should be replaced by "أ". Moreover, in the Arabic language, diacritics are often used in writing. The diacritics are eliminated since these diacritics are just written to identify the meaning of the word, and they do not affect the sentiment of the word. For example, if we have "مَسْتَحِيض", this word should be replaced by "مستحيض", which means "enjoyable" in English. Furthermore, people sometimes use tatweel, which is a way to expand the word for a writing style. For example, the word "مَسْتَحِيض" will be changed to
In order to handle the normalization process, a Python program is written using a package called "Tashaphyne" that is implemented by Taha Zerrouki [5]. This package has several methods for each of the above cases. The program scans all the tweets, and normalizes each word by calling these methods. After this program, each word in the dataset that is written using one of the explained matters will be normalized to be in a base form.

**Extract Conjunctions and Symbols**

In the Arabic language, conjunctions and symbols can be written without using a white space in between. For example, the word ""و يحب"", which means "and love" in English, can be also written as the word ""و يحب"" by using a white space between the word ""يحب"" and the conjunction ""و"". These differences can affect the matching process between the two words while in fact they are the same. To solve this issue, an Arabic analyzer in Java called "AraMorph" [2] implemented by Tim Buckwalter was used to analyze every word in each tweet, then extract and remove any unwanted conjunctions and symbols that we believe do not affect the sentiment analysis. In this case, the conjunction ""و"" will be removed from whether the word is written as ""و يحب"" or ""ويحب"". Thus, after this analysis the output of this word will be ""يحب"".

**Remove non Arabic characters**

After we analyze and replace any wanted emoticons and characters that are written using different hexadecimal representations, we can now delete any non-Arabic characters since these characters will not help at all in the classification process. In fact, they might affect the result of the classification model. In this step, characters that are not in the Arabic languages will be deleted, such as URLs and symbols (e.g., -, # and $). This step is completed using a
Python script that scans all the characters in the dataset, and deletes any character that does not belong to the Arabic language.

**Analyze wrong words**

In any language, specially in microblogging services, people usually write their posts using dialects, which means that they are not included in Modern Standard Arabic (MSA). Moreover, in the Arabic language, the form of the character can be changed depending on the position of the character in the word. In this case, every character in the Arabic word is connected to each other. However, there are some characters that do not connect to the following word. For example, if we have this word "لا يحب" , which means "does not like" in English, it is in fact "لا يحب" , but the two words "لا" and "يحب" are written connected to each other without using a white space. This example shows that if the white space is eliminated by the writer, it will not affect the form of the letter "لا" at the end of the word "لا يحب", which means "NO" in English. Thus, in this case, the reader will recognize and read the two connected words with no problem. Also, users in Twitter are always trying to write their posts with less characters to meet the limitation of 140 characters per post. To solve this issue, we run a Java program that analyzes words that are not in Modern Standard Arabic to find correct words. These words can be extracted from the dataset using the Arabic analyzer in Java called "AraMorph" [2] implemented by Tim Buckwalter. In our previous example, the word "لا يحب" is usually written and followed by another word without using a white space, so the word "لا يحب" is checked if it starts with "لا" , a white space will be added to split the word after "لا" to have "لا يحب" , then we check if the other word "يحب" is in the Modern Standard Arabic, we make sure that the correct form of the word "لا يحب" is "لا يحب".

Furthermore, users sometimes repeat a character more than once to emphasize their meaning. For example, the word "م(link)" , which means "mooore" in English, should be written as "م"; however, the letter "م" is repeated more than once. In this case, deleting
repeated characters is needed in order to have the base form of the words. However, some words already have repeated characters, such as “written” in English. To handle this matter, a Python program is written to delete repeated characters for words that are not in MSA. This list of words that are not in MSA is extracted also using ”AraMorph” [2]. Python is used to delete the repeated characters because of the ease of using regular expression, and the quickness of reading and analyzing the list of non-MSA words.

After the non-MSA words are analyzed, a Java program is implemented to replace non-MSA words with the new defined phrases. At the end, the dataset will be analyzed and cleaned, and it is ready for the next phase of the implementation.

Figure 3.2 shows an example of a tweet and the changes after each of the explained steps. An output of a tweet after this process would be as shown in the red rectangle of Figure 3.2.

Figure 3.2: The changes of a tweet after each step in the cleaning process
3.2 Calculate Sentiment Scores

Having a sentiment score of a review or a tweet might help to improve the results of the sentiment analysis since some words of unseen instances might have not been trained by the classification model. In order to have a good sentiment analysis, sentiment scores of each tweet are calculated. These sentiment scores consist of positive score, negative score and objective score. These sentiment scores of each tweet in the dataset are determined using a sentiment lexicon in Arabic that was collected by Badaro et al. [7]. This sentiment lexicon has more than 28K lemmas with almost 158K synsets, which means that each lemma may have a different part of speech, or different sentiment scores. Figure 3.3 shows a snapshot of the sentiment lexicon. Each line represents one word, which has the lemma of the word analyzed using Aramorph analyzer, POS, positive score, negative score. The other information is disregarded in our use. It can also be seen that each lemma has two different scores which are positive score and negative score. The objective score is calculated by \(1 - (\text{positive score} + \text{negative score})\). Moreover, the words in the lexicon are represented using lemmas not Arabic characters. For example, the word "جميل" is represented as "jamiyl" in the lexicon. In this case, analyzing Arabic words to define the lemmas of each tweet is needed. The "AraMorph" analyzer [2] is used to convert the Arabic words to lemmas before finding the sentiment scores from the lexicon. There are Arabic words that do not exist in the lexicon. In this case, these words will be disregarded, which means the sentiment scores are calculated using only the words are exist in the lexicon.

![Figure 3.3: A snapshot of the sentiment lexicon in Arabic](image)

Because of the different part of speech of each word, an Arabic POS tagger from the Stanford NLP Group is used [1], which helps to calculate the sentiment scores based on
the POS since some words can express different feelings of the writer based on their POS.

These sentiment scores will then be added to the selection features of the classification model. These scores consist of three values: positive score, negative score and objective score. Algorithm 1 shows how these scores are calculated.

### Algorithm 1: Calculate sentiment scores for each tweet

```plaintext
function getSentimentScores (T, L);

Input : A tweet T and the sentiment lexicon L
Output : Sentiment scores P, N and O, where P is the positive score, N is the negative score and O is the objective score

1. P = 0, N = 0, and O = 0;
2. while t_i \in T do
3.     lemma ← lemma of t_i by Aramorph tool;
4.     if lemma \in L then
5.         P ← P + positive_score;
6.         N ← N + negative_score;
7.         O ← O + (1 - (positive_score + positive_score));
8.     else
9.         end
10. end
```

After this process, the dataset is stored in a text file that has 10K lines. Each line consists of the tweet, label, and the three sentiment scores, which are positive score, negative score, and objective score. These scores will be added to the selection features of the classification model. Figure 3.4 shows a snapshot of the dataset after adding the sentiment scores.

Figure 3.4: A snapshot of our dataset after adding sentiment scores
3.3 Classification Algorithm

In this phase, the classification algorithms are implemented using Support Vector Machine (SVM) since it is known to give good results for text classification tasks. In this implementation, a Python program is implemented to build, train and test the classification model. The reason for using Python over other languages is fast runtime and execution, and the great package available for SVM called ”scikit-learn” [4]. We implement three different models; these three models show the improvements of the classification algorithm when one model is used over others. For all of the models we partition our dataset into two sets which are a training set that consists of 80% of the data, and a test set that consists of the rest of the data. While the training set is used to train and build the model, the test set is used to find how good the model is. Moreover, the document matrix, which is here the matrix of a tweet, is defined using TF-IDF technique, which is term frequency inverse document frequency. This matrix is built using the ”scikit-learn” package where it has a method called ”fit_transform” that converts a row document to a term-document matrix. The differences between the models are individually described in the following subsections.

Cleaned tweets without adding sentiment scores

In this model, the feature selections of the Support Vector Machine are extracted using term document matrix technique. After the term-document matrices for both the training set and test set are extracted, the SVM model will then be trained using the matrices of the training set. Then, the model will be tested and evaluated using the matrices of the test set. In short, in this model, the feature selection is extracted using only the term-document matrix of each tweet. The results of this model will be discussed in detail in Chapter 4.

Cleaned tweets with adding sentiment scores of each tweet

This model will be discussed as a single-stage approach since the classification will be processed using one stage. The only difference from the above model is that the calculated sentiment scores are added to the feature selection for SVM model. The sentiment scores
have three values: positive score, negative score and objective score. These scores are added to the support features after the term-document matrix is extracted using the method “fit_transform”. Thus, at the end the support features will include the term-document matrix that is defined using TF-IDF technique, and the three values of the sentiment score. Figure 3.5 presents the flowchart of this model. The results of this model are discussed in Chapter 4.

Figure 3.5: The flowchart of the classification model that uses sentiment scores to extract support features
Table 3.1: The number of tweets of each class in the dataset

<table>
<thead>
<tr>
<th>No. of Tweets</th>
<th>Label</th>
<th>General Label</th>
<th>No. of tweets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1684</td>
<td>Negative</td>
<td>Subjective</td>
<td>3315</td>
</tr>
<tr>
<td>799</td>
<td>Positive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>832</td>
<td>Neutral</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6691</td>
<td>Objective</td>
<td>Objective</td>
<td>6691</td>
</tr>
</tbody>
</table>

Classifying Cleaned tweets with sentiment scores using a two-stage approach

When we look at the dataset, we find that it has four classes: objective, positive, negative, and neutral. The objective class has 6,691 tweets while the closest class is the negative class which has 1684 tweets. Due to the big difference between the numbers of instances of the classes, a two-stage classification is implemented to see if this approach will be significant over the single-stage approach. Figure 3.6 shows how the two-stage classification can classify a given tweet. It can be seen the red ellipses mean the final label of the given tweet.

![Figure 3.6: How a given tweet can be classified in the two-stage approach](image)

In the dataset, the positive, negative and neutral classes can be combined into one class called subjective. Table 3.1 shows the number of tweets for each class before and after the combination.

In this model, we build two classification models. While the first model is to classify the tweet as subjective or objective, the second model is to classify the subjective tweets from the first model as positive, negative or neutral. Figure 3.7 shows the flowchart of the two stage approach.
As presented in Figure 3.7, the dataset is first converted from having four classes to only two classes: subjective and objective. Any tweet that is labeled as positive, negative or neutral would be labeled as subjective. The first model will be trained using this dataset. The output of this model that are correctly classified as subjective will be then used to test the second model. The second model is trained using the dataset after removing objective tweets, and it will be tested by the output of the first model. The results of these two models will be discussed in detail in Chapter 4.
Chapter 4

Analysis

The purpose of this project is to provide strategies for sentiment analysis and classification of tweets. The Arabic language is used as a case study to develop these strategies. In the previous chapter, we present the implementation of how we preprocess and clean the dataset, and how we implement the classification algorithms. We also show that there are three different models that have been implemented for the sentiment analysis. In this chapter, the results of the implemented models will be analyzed and discussed in detail. A $t$ test will be used to compare between the one stage classification and the two-stage classification. This test will show which model is most significant.

For each model, the algorithm is run 100 times using a random seed to shuffle and partition the dataset into training set and test set. This randomness changes the training and test sets in every iteration, which means that the algorithms are tested using 100 combinations of the training and test sets. There are several ways to measure and evaluate the classification models. The basic and common measurement is the accuracy of the classification model, and that is determined using equation 4.1. Precision and Recall are performance measures that also used to evaluate the classification model. While Precision is used to evaluate the exactness of the model and it is computed using equation 4.3, Recall is used to evaluate the completeness of the model and equation 4.4 shows how it is computed. In order to compute the balance between Precision and Recall, the $F_1$ score is used for this purpose, and it is determined using equation 4.2. In this paper, we would only show the value of $F_1$ score since it recovers the balance between Precision and Recall.
\[ \text{Accuracy} = \frac{T_{\text{positive}} + T_{\text{negative}}}{N} \]  

(4.1)

Where:

- \(T_{\text{positive}}\): is the number of instances that are correctly classified
- \(T_{\text{negative}}\): is the number of instances that are not correctly classified
- \(N\): is the total number of instances in the test set

\[ F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \]  

(4.2)

\[ \text{Precision} = \frac{T_{\text{positive}}}{T_{\text{positive}} + F_{\text{positive}}} \]  

(4.3)

\[ \text{Recall} = \frac{T_{\text{positive}}}{T_{\text{positive}} + F_{\text{negative}}} \]  

(4.4)

In order to run the algorithms, a Macbook with a 3 GHz Intel Core i7 processor and a memory of 16 GB 1600 MHz DDR3 was used for the three models. The following sections will show a brief explanation of the results for each model.

### 4.1 Model 1: Cleaned tweets with no sentiment scores

In this model, the support features of the classification model are defined using the term-document matrix. Table 4.1 shows the results of this model after running the algorithm 100 times. The runtime of this model for all the iterations was 499.9 seconds. In Table 4.1, it can be seen that the average accuracy of this model is 68.9 with a maximum accuracy among the iterations of 70.2%. \(F_1\) score shows that this model does not perform well in terms of exactness and completeness.
Table 4.1: The results of the first model that uses tweets with no sentiment scores

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>70.2</td>
<td>67.8</td>
<td>68.9</td>
</tr>
<tr>
<td>F1-Score</td>
<td>62.01</td>
<td>58.5</td>
<td>60.3</td>
</tr>
</tbody>
</table>

Table 4.2: The results of the second model that uses tweets and the sentiment scores

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>74.5</td>
<td>72.4</td>
<td>73.3</td>
</tr>
<tr>
<td>F1-Score</td>
<td>68.3</td>
<td>65.2</td>
<td>66.6</td>
</tr>
</tbody>
</table>

4.2 Model 2: Cleaned tweets using sentiment scores

In this model, one idea for improving the sentiment analysis would be to add sentiment scores of each tweet that describe the positiveness or negativeness of the tweet. This model was also run 100 times and it performs well compared with the previous model. The runtime of this model was 532.8 seconds. Table 4.2 shows the results of this algorithm. It can be seen that this model gives an average of 73.3%, which is greater than the average of the previous model. The difference between Accuracy and $F_1$ score goes down also in this model, which means there is a slight improvement in the exactness and completeness of the model.

4.3 Model 3: Cleaned tweets using sentiment scores classifying using a two-stage approach

After we looked at the distributions of the tweets among the classes, we thought that classifying using a two-stage classification would give a better result than using a single-stage classification. This model was run 100 times, and the runtime was 644.3 seconds. In order to show the accuracy, and $F_1$ score of this model, the averages of the accuracies and $F_1$ scores for the two models are computed. Table 4.3 shows the averages of both the accuracy and $F_1$ score. This model performs as the best model among the other models in term of exactness and completeness.
Table 4.3: The results of the third model that uses tweets and the sentiment scores classifying using a two-stage classification

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>73.3</td>
<td>68.7</td>
<td>70.7</td>
</tr>
<tr>
<td>F1-Score</td>
<td>69.6</td>
<td>64.03</td>
<td>67.02</td>
</tr>
</tbody>
</table>

4.4 Comparison between a single-stage approach and a two-stage approach

In order to compare between these two models, a *t* test is used which uses the means of the two models to compare and show the significant mean among them. This test is usually used to compare between two independent groups to see which group performs better with a significant difference. The output of this test includes a *P* value which describes whether the difference is significant or not. An online calculator was used to compute the *t* test [3]. In our case, the two independent groups were the single-stage model and the two-stage model. The accuracies of the 100 iterations of each model were the inputs to the calculator. Table 4.4 shows the result of this test. The calculator gives a *P* value of less than 0.0001, which means that the difference is statistically significant. This finding shows that our assumption was wrong, and the single-stage classification performs better than the two-stage classification. However, when we looked deeply into the classification results and review the misclassified instances, we could see that the problem was because we were classifying instances having an overlap among them. Unlike negative words, positive words can be found in the three classes which are positive, neutral and objective. The results of the first model in the two-stage approach were good since they classified two classes. The first model gives an average accuracy of 74.91%, and it gives an average $F_1$ score of 72.5%, which means that the first model performs well in the exactness and completeness. In short, our new assumption is that if the classification problem is to classify between only two classes that do not have an overlap among them, it will give a very good result compared with our current problem.

To prof the previous assumption, another two-stage classification was implemented,
Table 4.4: The results of the t test for the two models using their accuracies

<table>
<thead>
<tr>
<th>Group</th>
<th>Single-Stage</th>
<th>Two-Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.7335</td>
<td>0.7076</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>0.00482</td>
<td>0.01154</td>
</tr>
<tr>
<td>Standard error of the mean</td>
<td>0.000482</td>
<td>0.001154</td>
</tr>
</tbody>
</table>

Table 4.5: The confusion matrix of one of the iteration in the second stage of the two-stage approach

<table>
<thead>
<tr>
<th>Label</th>
<th>Negative</th>
<th>Positive</th>
<th>Neutral</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negative</td>
<td>157</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Positive</td>
<td>11</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>Neutral</td>
<td>49</td>
<td>1</td>
<td>15</td>
</tr>
</tbody>
</table>

which classifies between two classes in the first stage, and two classes in the second stage. While the two classes in the first stage are objective, the two classes in the second stage are positive and negative. This model gives an average accuracy of 80.61%, and average $F_1$ score of 79.70%. In order to see where the false positive goes to, Table 4.5 shows the confusion matrix of the previous two-stage approach. It can be seen that most of the neutral instances are incorrectly classified as negative. Since we did not annotate the dataset, we can not make sure that the dataset is annotated correctly. Neither can we make sure that there are overlaps among the classes. However, this finding could show that there is an overlap between negative and neutral classes.
Chapter 5

Conclusions

This chapter is to conclude our work on this project. The current work is described in Section 5.1. Section 5.2 describes the possible future work that can be appended to this project. The overall conclusion of this project will be given in Section 5.3

5.1 Current Status

The goal of this project is to address some strategies for sentiment analysis and classification of tweets in any language. These strategies were developed in this project to deal with Arabic tweets. The approach of this project was to divide the sentiment analysis problem into four steps: data collection and annotation, data cleaning and preprocessing, calculating the sentiment score based on a sentiment lexicon, and building, training and test the classification model. This project addressed strategies and solutions of each step. Three different approaches were implemented. The results of these approaches were evaluated and discussed, which gives an average accuracy of 73.3% when a one-stage approach is used, and 70.7% when a two-stage approach is used.

5.2 Future Work

Several future work for this project. First, for the Arabic language, the future work would include collecting and annotating the dataset to make sure of the annotation. A sentiment lexicon for idioms and dialect words should be built and extracted. Moreover, users in Twitter use different dialects to express their feelings. Thus, a dictionary to connect between
these dialects would improve the performance of the classification model. Furthermore, in the Arabic language, some Arabic persons’ names are positive adjectives, so it would be a good idea if we could identify and remove these names so that they do not affect the classification process. Other future work would include applying our presented strategies and solutions to another language to see the performance of the sentiment algorithm. Other languages could include Spanish, French or Chinese.

5.3 Conclusion

The purpose of this project is to present and discuss several steps in how to deal with a sentiment analysis problem. The project present strategies in how to collect and annotate the dataset, clean and preprocess the dataset, train and test the classification model. Three classification models were implemented, which include classifying tweets without adding sentiment scores to the support features, classifying tweets after adding sentiment scores to the support features, and classifying the tweets using a two-stage approach. The result of this project did not turn out how we had expected it to. It shows that the one-stage approach performs better than the two-stage approach. A $t$ test was used to show that the mean of the single-stage approach is more significant than the other approaches.
Bibliography


