Journal of Theoretical and Applied Information Technology
15th June 2020. Vol.98. No 11
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

HATE SPEECH CLASSIFICATION IN ARABIC TWEETS

[1]SHAIMA AL-KHALIFA, [2]IBRAHIM ALJARAH, [3]MOHAMMAD A. M. ABUSHARIAH

[1]Department of Computer Science, King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan

[2]Department of Information Technology, King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan

[3]Department of Computer Information Systems, King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan

E-mail: [1]shaimaPhd@gmail.com, [2]i.aljarah@ju.edu.jo, [3]m.abushariah@ju.edu.jo

ABSTRACT

The rapid growth of using social media has produced some serious undesirable outcomes such as hate speech. Over the recent years, the amount of hate speech has widely spread and incredibly increased. Thus, there is a need to detect hateful content that may lead to violent actions and criminal activities. While most of the previous research focus on the detection of hate speech in English language and other languages, Arabic language is less emphasized and require more attention by the research community. This paper aims to present our work for detecting hate speech over Twitter platform as one of the main Online Social Networks (OSN) based on Arabic language. A dataset of 3000 tweets is collected in this work, which was experimented using Random Forest (RF), Naive Bayes (NB), and Support Vector Machine (SVM) as classification algorithms. In addition, feature extraction is conducted using Bag of Word (BoW) and Term Frequency–Inverse Document Frequency (TF-IDF). Based on our experimental results, SVM maintained consistently high performance and outperformed other classifiers, and TF-IDF outperformed BoW, which consequently achieved the highest accuracy.

Keywords: *Hate speech, Twitter, Random Forest, Naive Bayes, Support Vector Machine.*

1. INTRODUCTION

Online Social Networks (OSN) have become essential in our daily life activities, where users can post and share their opinions, feelings, reviews, and sentiments on any worldwide event, incident, product, and many more. OSN data has a textual orientation; therefore, text mining and Natural Language Processing (NLP) techniques and methods play major role in information retrieval, opinions extraction, sentiment analysis, and many more. Sentiment Analysis (SA) or opinion mining is referred to as the process of mining the opinions, reviews, and emotions from text or tweets and other OSN data forms with the help of NLP, data mining, and machine learning tools, techniques, and algorithms. SA has many tasks such as polarity classification that classifies the sentiment into different classes usually positive or negative, and sometimes neutral [1]. Sentiment classification is performed using three main approaches, namely, machine learning, semantic orientation, and a hybrid approach [2]. The machine learning approach involves the use of supervised machine learning algorithms for learning and testing data using classifiers such as Support Vector Machine (SVM), Random Forest (RF), Naive Bayes (NB). The semantic orientation approach is an unsupervised approach, where the sentiment lexicon and the linguistic rules are used based on rule-based classifiers [3]. The hybrid approach can be viewed as a combination of machine learning and semantic approach.

SA can be involved in solving OSN critical problems like cyber hate speech, which is one of the most popular problems in OSN such as YouTube, Facebook, Twitter, and others. It refers to the use of hateful content against individuals and groups with an intention of bringing harm and raise violence toward them. It can be triggered by provoking events that arise anger and hate based on race, gender, or religion [4].

Hate speech can be viewed as a classification problem that needs to build a machine to learn and determine whether the text contains hate words or otherwise. In addition, it requires NLP techniques for text normalization.

Arabic language is a widely spoken language for more than 330 million people in 22 countries [5]. It is widely and increasingly used in OSN nowadays. It is reported that the percentage of Facebook users from the Arab region is around 8.4% of all Facebook users, which means more than 150 million Facebook users are Arabs [6]. Laws, rules, and regulations against the use of hateful content in OSN may reduce and limit hate speech growth in the Arab world.

In the research field, hate speech detection is a hot topic, where most of the previous research attempts are about the detection of hate speech in English text. However, it is realized that the number of research efforts that have been conducted in the Arabic language are limited, which motivated us to explore the challenges to detect hate speech in Arabic OSN.

Dealing with Arabic language is a challenging process due to the limited resources and availability of NLP tools. In addition, the process of collecting data sets from OSN such as Twitter platform is very challenging, whereby a huge number and volume of tweets can be posted online in a matter of seconds. It is essential to detect whether the posted data contain or not hate speech. The data sets should be initially gathered and pre-processed, and trained to extract and classify their features.

In addition, to analyze Arabic sentiments, a number of challenges arise. Firstly, Arabic letters have different orthographical shapes based on their position (beginning, middle, and ending) in a word [7]. For instance, the letter "ك" /k/ is written as "كـ" at the beginning of the Arabic word "كتاب" that means "Book" in English language. It is also written as "ـكـ" in the middle of the Arabic word "مكتبة" that means "Library" in English language, and is written as "ك" at the end of the Arabic word "ملاك" that means "Angel" in English language.

Secondly, Arabic language does not contain capital letters, which is essential in text mining and helps in feature extraction [8].

Thirdly, the users tend to repeat letters to express emotions [9] such as the Arabic word "جداااا" that means "soooo" in English language, and the word "هههههه" that represents laughing in English using "hahaha".

Fourthly, Arabic is a morphologically rich language [10], where one root may have multiple derivational and different forms. For instance, the Arabic root word "كتب" that means "Write" in English, can have multiple and different forms with different meanings such as the Arabic word "يكتب" that means "He Writes", "أكتب" that means "I Write", "نكتب" that means "We Write", "مكتبة" that means "Library" in English language, and many other forms of the same Arabic root word "كتب".

Fifthly, the use of negation can change the meaning of the sentence [11]. For instance, the Arabic sentence "لم تعجبني الكاميرا" that means "I did not like the camera" in English language. The Arabic word "لم" that means "Not" in English language has changed the meaning of the sentence and made the sentiment of the sentence as negative.

Sixthly, Arabic language has diacritics that appear on its letters and text [12], and the word meaning could change based on their existence or absence. For instance, the Arabic word "ذهبَ" means "Went" in English, the last letter "ب" /b/ has the diacritic " ـَ " /a/, while the same word can be written with a different diacritic "ذهبْ" and resulted in a change to the meaning to be "Gold" in English, where the last letter "ب" /b/ has diacritic " ـْ " "Sukoon" that indicates silence.

Our research deals with the detection of hate speech in Arabic tweets in general, against race, gender, nation, and religion. In our work, we firstly collected a data set of 45,000 tweets that were then sampled to 3000 tweets in this work particularly, and have been classified into two classes, namely: not hate, and hate. We finally applied popular classifiers on the data set such as SVM, NB, and RF. In addition, the experimental results are compared against each other to discover the classifier that outperforms other classifiers in terms of accuracy, precision, and recall measures.

The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 provides a background of NLP's most popular techniques and machine learning algorithms. Section 4 presents our proposed research methodology. Furthermore, Section 5 provides our experimental results. Finally, Section 6 highlights the conclusions and future work.

2. RELATED WORK

Recently, the interest in detecting hate speech has rapidly increased. It has attracted the attention of many researchers, trying to develop models and methods to detect hateful content and extracting hate features. Some of these previous researches have required sentiment analysis, NLP techniques, and machine learning approaches to detect hate speech. In this section, we briefly discuss some of these studies.

The researchers in [13] used SVM as the classifier to detected hate speech on the Web. Unigrams technique is used in [14] to detect tweets with hateful content against blacks. Moreover, the lexicon-based approach is used for hate speech detection in [15].

The attempt to build a probabilistic classifier was conducted in [16] using a word embedding approach, where the website comments are classified into hate or not hate comment. Some of the previous research works have focused on the detection of hate speech in Twitter platform such as [17] and [18] and applied binary classification algorithms in their work such as logistic regression (LR), Random Forest (RF), Decision Tree (DT), and SVM.

A study of hate speech detection is conducted in [19] using Twitter and Whisper against black people. The researchers in [20] have conducted a study of hate speech detection on Twitter platform, where sexism and racism were identified in English tweets using the n-gram model. In the work of [21], a word embedding approach for hate speech detection was applied.

Furthermore, researchers in [22] have employed the SVM classifier for detecting hate speech in English tweets, and a survey on hate speech detection using NLP was investigated in [23].

Based on the prior attempts for hate speech detection as found in literature, it can be concluded that Arabic language is not well studied and investigated specifically for hate speech detection.

Regarding previous researches on Arabic OSN content, they have mainly focused on either detecting offensive tweets such as [24], [25] and [26], or detecting irony tweets such as [27], [28] and [29], which are different from hate speech detection. On the other hand, Weber et al. [30] employed a quantitative and data-driven analysis to identify the phenomena of secular versus Islamist polarization in Egypt. The data set was collected from Twitter platform for both English and Arabic languages. A list based approach for detecting abusive language in Arabic tweets was developed in [31], where a list of obscene words that are used for detecting abusive tweets was made.

In addition, De Smedt et al. [32] proposed a model for automatic detection of online jihadist hate speech on Twitter using machine learning and NLP techniques. They have run qualitative and quantitative analysis on a corpus of 45K tweets. The work of [33] is devoted for detecting hate speech against religion in Arabic Twitter.

Albadi et al. [34] incorporated Gated Recurrent Units (GRU) neural networks with handcrafted features for religious hate detection on Twitter. The authors in [35] and [36] have detected anti-Shia hate speech on Twitter. Al-Hassan and Al-Dossari [37] presented a survey on previous research efforts pertaining to the detection of hate and offensive language in Arabic. Chowdhury et al. [38] focused on the detection of religious hate speech in Arabic tweets and proposed a model that combines Arabic word embedding and social network graphs.

Haddad et al. [39] built a Tunisian hate speech and abusive data set (T-HSAB). Mulki et al. [40] introduced a Levantine Twitter data set for hate speech and abusive language (L-HSAB).

Additionally, the majority of research works focus on English language. However, some researchers have presented methods to detect hate content in other languages such as profanity detection in Chinese [41], racism detection in Dutch [42], German [43], Indian [44], Indonesian [45], and Italian [46].

The most similar approach to ours as presented in this paper is the one by the researchers in [17] and [18]. We used the same machine learning classification algorithms including SVM, NB, and RF. However, in their approach, they used two additional algorithms: DT, and Bayesian Logistic Regression (BLR). On the other hand, in our work two methods of feature extraction are used including BoW and TF-IDF, whereas BoW alone is used in [17] and TF-IDF alone is used in [18].

3. RESEARCH BACKGROUND

3.1 NLP Techniques

NLP techniques for text pre-processing include many processes that help to transform and extract features from the text, which can serve as an input for machine learning classification algorithms. The most common processes are tokenization, stop word removal, and feature extraction.

Tokenization is one of the NLP essential tasks that can be defined as the process of breaking up a text/string into parts such as words or sentences that are called tokens. Tokens can be considered as the input for another process (transform text vectors), which are separated by white space, punctuation marks, and line breaks. For instance, the document is split into paragraphs, paragraphs into sentences, and sentences into words.

For example, the paragraph “Welcome to our class. Our lecture is about tokenization.”, can be tokenized into sentence one “Welcome to our class.”, and sentence two “Our lecture is about tokenization.”. Furthermore, sentences can be tokenized into words including “Welcome”, “to”, “our”, “class”, “.”, “Our”, “lecture”, “is”, “about”, “tokenization”, and “.”.

The concept of stop-word removal has been presented first by Hans Lauhan in 1957. He suggested that the words in the text are either a keyword or non-keyword terms. Stop words are meaningless words that have been used repeatedly and usually are written as part of sentences. They can be pronouns, conjunctions and prepositions such as "he", "and", "to", and many others. Removing them could help to reduce text dimensionality and improve performance of machine learning algorithms.

Therefore, the tokens of the aforementioned sentences are reduced after removing punctuation marks and stop words, which will be "Welcome", "class", "lecture", and "tokenization".

In addition, the tokens are transformed into a feature vector: [1,1,1,1], which is based on the number of times the word is repeated in the text. This transformation process is called text vectorization for features extraction, which plays a crucial role in machine learning classification algorithms that use the feature vector as their input.

The two commonly used NLP techniques for feature extraction include BoW and TF-IDF. The BoW is one of the simplest feature representation techniques that is used in NLP, where the BoW model can be described as a bag where each word is collected and stored in it. The order of the words is not important and each word is assigned a weight according to its frequency in a document and in different documents [47]. It counts the number of times the word occurs in a document and saves it in a vector.

For example, given two documents, where D1 is: {"good friends are loyal people"} and D2 is: {"real friends are needed"}.

The dictionary of words contains: 1: 'good', 2: 'friends', 3: 'are', 4:' loyal', 5: 'people', 6: 'real', and 7: 'needed'. Documents D1 and D2 are represented as 7-element vector, D1: [1,1,1,1,1,0,0] and D2: [0,1,1,0,0,1,1].

Similar to BoW model, TF–IDF does not take into consideration the order of words in a document but differs in the method of calculating the word frequency. It includes two steps, where the first step includes calculating the Term Frequency (TF) and the second step includes computing the Inverse Document Frequency (IDF).

TF is the number of times the term *t* appears in the document divided by the total number of terms in the document, which can be calculated as in Equation (1).

Inverse Document Frequency (IDF) identifies the frequency of term *t* that occurs in all documents, which can be calculated using Equation (2).

The IDF measures whether a term is common or rare in a given document corpus. The more common the term is used across the document corpus, the lower the IDF and the less important the term becomes.

TF–IDF is the result of multiplying TF by IDF, as calculated in Equation (3).

$$TF = \frac{n_t}{n} \tag{1}$$

$$IDF = \log 2 \frac{N}{N_t} \tag{2}$$

$$TF\text{-}IDF = \frac{n_t}{n} \cdot \log 2 \frac{N}{N_t} \tag{3}$$

Where:
n_t = the number of occurrences of term *t*
n = the total number of terms
N = the total number of documents
N_t = the number of documents containing term *t*

3.2 Machine Learning Classification Algorithms

3.2.1 SVM

SVM was introduced by Vapnik [48], which is a supervised machine learning algorithm that finds the optimal hyperplane by splitting the training data set into two classes. The effective elements in the training data set can be addressed as support vectors only, which are employed in decision making [49] and [50] as if the testing data set belongs to a certain class. The hyperplane function is defined in Equation (4) subject to Equation (5), and the formula that helps to find the set of effective elements is defined in Equation (6) as follows:

$$\frac{1}{2} W^T W + C \sum_i \xi_i \tag{4}$$

$$\{(x_i, y_i)\}, y_i (W^T x_i + b) \geq 1 - \xi_i \tag{5}$$

$$f(x) = sign(W^T x + b) \tag{6}$$

Where:
W = weight vector
C = loss function
ξ_i= misclassification vector *i*
x_i = train vector *i*
y_i= class train vector *i*
b = bias vector

$f(x)$ = the score function
x = the test vector

3.2.2 NB

NB is a well-known probabilistic algorithm that is based on Bayes theorem, which is widely used in text classification as an efficient and simplest classifier.

Where given data set of parameters/features represented by vector X, that $X = \{x_1, x_2, \ldots, x_n\}$ with n features and C as class variable, given that the features x and class C are independent and do not affect each other. These are formulated in Equation (7) and Equation (8) as follows:

$$P(C|X) = \frac{P(X|C)\,P(C)}{P(X)} \quad (7)$$

$$P(C|X) = P(X_1|C)\cdot P(X_2|C)\cdot \ldots \cdot P(X_n|C)\cdot P(C) \quad (8)$$

Where:
P(C|X) = the posterior probability of class C given attribute feature X

P(X|C) = the likelihood that the probability of attribute feature X is given the class C

P(C) = the prior probability of the class C

P(X) = the prior probability of attribute feature X

3.2.3 RF

RF was introduced by Leo Beriman and Adele Culter in 2001 [51], which is an ensemble machine learning algorithm that trains classifiers first then combine their results in a voting process [33]. RF can be considered as one of the best classification algorithms, which has high classification accuracy and ability to classify a large amount of data set like OSN data [34].

RF consists of multiple combinations of a different set of decision trees that are trained by a different set of the training set, with the use of bootstrap and random feature selection, which are used in order to create different training set from the original one. Bootstrapping is the process in which the training data set is trained and continuously evaluated to improve classifier's performance [79].

Every tree has its own decision, where the final decision in the frost is taken based on the maximum voted class. Due to RF hierarchical structure, the feature selection process can be learned and done automatically [80].

4. PROPOSED METHODOLOGY

This section describes the techniques and methodological steps that are used in our approach. The proposed methodology can be summarized into five phases, namely: 1) data collection and annotation, 2) data preprocessing, 3) feature extraction, 4) classification, and 5) evaluation, as illustrated in Figure 1.

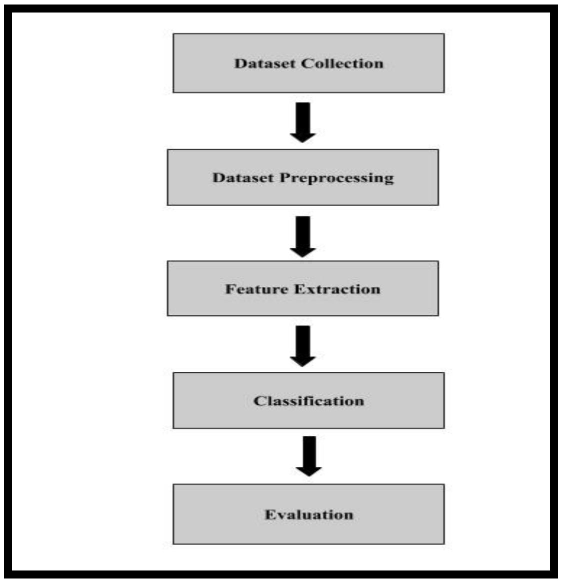

Figure 1: Model Schema

Firstly, we collected the data from Twitter platform. Secondly, the collected data must be pre-processed. Thirdly, we transformed the data into a feature vectors by employing techniques like BoW and TF-IDF in order to extract the words/features.

Fourthly, we trained and tested the classifiers. Fifthly, we compared the results using evaluation metrics such as precision, recall, F-measure, and accuracy.

4.1 Data Collection and Annotation

In this stage, we describe the data collection and annotation processes briefly. First, the data set is retrieved from Twitter platform. Consequently, the collected data goes through the annotation process to be labeled as either "hate" or "not hate", which are stored in a (.csv) file. Figure 2 illustrates the data collection and annotation processes.

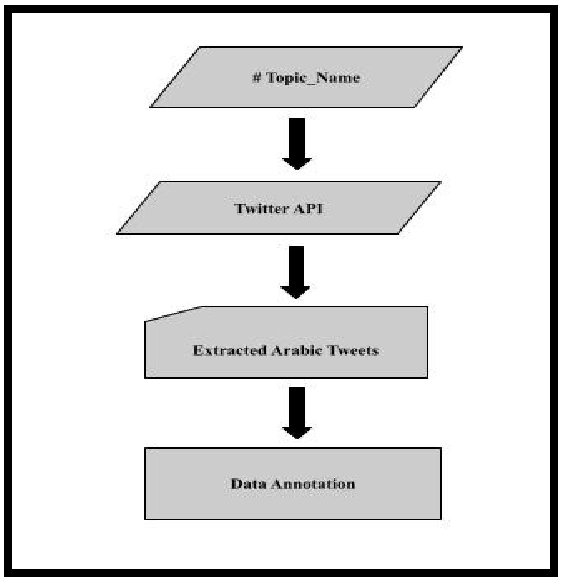

Figure 2: The Data Collection Process

To the best of our knowledge, there is no benchmark hate speech data set in Arabic language. Therefore, we created our data set that contains tweets as collected from Twitter platform, which are about different topics, such as the current Arabs political events such as a hashtag like "جمال_خاشقجي" in English "Jamal Khashoggi", "العراق" in English "Iraq", and other hashtags about the Middle East current events.

We used the Twitter Application Programming Interface (API) to collect tweets, which enabled us to retrieve a stream of tweets of over 45,000 tweets, within the time duration from 14/12/2018 to 26/12/2018.

The time duration has been about two weeks following the event in order to collect the maximum number of tweets, because the public interest is raised shortly after event occurrence and is decreased fast after two weeks. In addition, a large number of different responses will be posted during two weeks of that event [52].

Only 3,000 of them have been sampled to be annotated and pre-processed. Table 1 shows the number of tweets and samples, which are used later to train and test the classifiers.

We have collected over 45,000 tweets and randomly have sampled over 3,000 of them after eliminating the following redundant tweets: tweets contain ads, many of the tweets have been actually ads that promote products or services. Irrelevant tweets, where a few tweets have been out of the scope of our topic. Short tweets, where several tweets are very short to understand them or label their class. Repeated tweets, where there is a large number of duplicated tweets that may have occurred a result of being retweeted. Tweets written in other languages that use Arabic scripts such as Kurdish and Persian are also deleted.

Table 1: Hashtags and Collected Tweets

Hashtag	Number of Tweets	Number Of Selected Tweets
العراق	38,017	1,772
جمال _خاشقجي	2163	413
الفساد_ السياسين	2955	417
مقاطعة	2259	398

These tweets have two classes: hate or clean (not hate) as shown in Table 2.

Table 2: Classes and Definitions

Class	Definition
Hate	Contains hate words
Not Hate	does not contain hate words (clean)

After the collected data set is sampled to 3,000 tweets, each tweet must be labeled to either "hate" class or "not hate" class, where the hate class is given label 1 and the not hate (clean) class is given label 0, as shown in Table 3.

Table 3: Tweets Examples

Class	Label	Tweet Example
Hate	1	مقاطعة بلد الدعارة والمثلية واجب قومي على كل عربي ومسلم Boycott the country of porn and homosexuality is a national duty of every Arab and Muslim.
Not Hate	0	خاشقجي ينافس ترامب وبوتين على لقب "شخصية العام" حسب تصنيف مجلة "تايم" Khashoggi competes with Trump and Putin for the title "Personality of the Year" according to the "Time" magazine rankings

There are several annotation methods, where some of the researchers used the hate base website (https://hatebase.org) for annotation purposes to extract and capture hate words from their database. Other researchers have employed the popular crowd flower for tweet online annotation [52] and [18], which provide paid online service, where the annotators and tweets can be determined, and the results from the annotators could be accepted or rejected. The other annotation method can be done manually, which requires unbiased annotators who volunteer to annotate the tweets. The annotators can be the researchers themselves [20] and [53], or volunteers [15].

In our work, the annotation has been done manually as we carefully selected three annotators for tweets annotation who possess fair experience of using social media. The selected annotators are Arabic native speakers with different educational backgrounds, gender, and age. They were given definitions with examples and guidelines of hate speech, and examined the selected tweets to decide on the hate or not hate (clean) tweets.

At the end of the annotation task, if there is a conflict in annotating any tweet, they can reach the final decision based on the voting process that helps to reach to an agreement for each tweet.

Finally, after the annotation process has been finished, we found that 47% of tweets are labeled as hate, and 53% of tweets are labeled as not hate (clean). When this step is finished, the labeled tweets have to be saved in a (.csv) file in order to be pre-processed in the next step.

4.2 Dataset Preprocessing

In this phase, the dataset goes into four stages as shown in Figure 3.

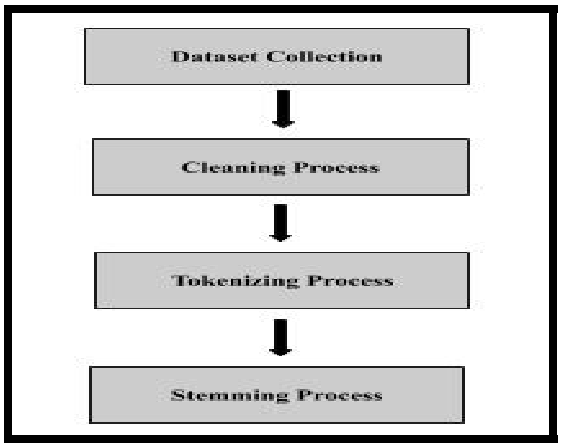

Figure 3: Dataset Preprocessing Operations

The dataset has to be cleaned from special characters, emoji, and numbers, which is then split into tokens or words, and consequently the words have to be stemmed to their roots.

4.2.1 Cleaning

Arabic text pre-processing is a very critical and crucial step. The output of this step can be used in the classification process later. Tweets may have many repetition letters, special characters, and emoji that must be removed. Therefore, the cleaning process works sequentially as follows:

Firstly, punctuation marks such as full stop “.”, question mark “?”, and comma “,”, must be removed.

Secondly, repetition letters in a word such as the Arabic word “جدااااا” that means “soooo” in English must be removed too. The result after deleting the repeated letters must be the Arabic word “جدا” that means “so” in English.

Thirdly, special characters such as the parenthesis, URLs, #, @, /, must be removed because they are considered as meaningless and redundant data.

Fourthly, non-Arabic words must be removed too.

Fifthly, Arabic and English numbers should be removed.

Sixthly, Emoji are also removed from the dataset.

Seventhly, stop words are removed. The idea of removing stop words has been presented by Hans peter Luhan in (1957), where stop words are common words in any text document that can be articles, pronouns, and prepositions. These words do not give any meaning to the text, thus they should be eliminated, which helps to reduce the text data size [54], and enhance the performance of the classification algorithm. Examples of Arabic stop words can be the word “هو” that means “he” in English, and the word “أنت” that means “you” in English. In our work, we used the “stopwords” tool within the Natural Language Toolkit (NLTK) for removing stop words.

4.2.2 Tokenizing

Tokenizing is the process of splitting the text into tokens or words. Each tweet will be tokenized (transformed into words/tokens). Many researchers contributed to the tokenizing process for Arabic language such as the Buck Walter tokenizer, which is based on the morphological approach [55]. In addition, Larkey and Connel have developed a tokenizer that depends on information retrieval [56].

Deib et al. used part of speech tagging [57]. Nulker and Shiber [58] used a diacritization method for tokenization. Attia [59] developed a rule-based tokenizer that performs the tokenization process in two stages including a preprocessing stage for white space removal and a post-processing stage for token filtering.

In our work, we used the “word_tokenize” tool within NLTK. For example: the Arabic sentence

"العراق بلد جميل" that means "Iraq is a beautiful country" in English, will be tokenized into three Arabic words, including: "العراق", "بلد", and "جميل".

4.2.3 Stemming

Stemming is a process of reducing tokens or words to their root by removing their affix (infix, suffix, and prefix), such that multiple tense representations of a word are reduced to one. For example, the Arabic word "رافضة" that means "rejectionist" in English, and "مرفوض" that means "rejected" in English, will be stemmed into "رفض" that means "reject" in English.

There are three approaches for stemming, namely: 1) the root base stemmer approach, 2) the light stemmer approach, and 3) the statistical stemmer approach [60]. The root base approach removes all suffixes and prefixes of the words and finds the root of the Arabic surface word. The light stemmer approach deals with removing the most frequent prefixes and suffixes, ignores the linguistic root of the Arabic word surface [61], and might change the word form. On the other hand, the statistical approach uses the similarity comparison to group a set of relative words [62].

Khoja stemmer [63] is an early attempt of stemming in Arabic language, which is a root based stemmer. It normalizes the word, removes numbers, punctuation, and diacritics, removes conjunctions and articles, and removes the suffixes and prefixes. Finally, it extracts the word to its root by matching it with various patterns, then uses a dictionary to validate the root. Khoja stemmer sometimes does not remove all suffixes and prefixes and requires updates for the new words in order to be stemmed correctly, which can be treated as a weakness.

Larkey et al. [64] developed the Light stemmer that first removes non-Arabic letters, punctuation, and diacritics, then removes suffixes and prefixes based on specific conditions. Unlike Khoja stemmer, it does not deal with the irregular plural, infixes, and patterns [65]. From accuracy perspective, Khoja stemmer is better.

Taghva et al. [66] built Information Science Research Institute's (ISRI) stemmer. They used a similar method to Khoja stemmer by normalizing the word at the beginning then removing only prefixes and suffixes of the word with rules, including: suffix will be removed only if the result is a word with two or more letters, and the prefix will be removed when the result is a word with three or more letters. However, it does not use the root dictionary, and does not handle the irregular plural, which may result incorrect stemming of words.

N-gram approach is proposed in [67] and [68], which extracts the word's root based on similarity comparison, where the words with similar characters have a high rate of N-gram value (bi-gram, tri-gram). After the comparison process is done, the word root will be found.

Tashaphyne stemmer [69], is similar to the ISRI stemmer, which tries to stem a word to its root with minimal representation. However, the only difference is that Tashaphyne stemmer tries to balance between the root and the removed suffixes and prefixes [70], uses two lists of suffixes and prefixes to stem the word, and stems the word to either a root or a light stem [71].

Abainia et al. [72] designed a new Arabic Light Stemmer (ARLStem) that removes the word prefixes, suffixes, and infixes. For example, the Arabic word "إسلامية" that means "Islamic" in English, is stemmed using ARLStem into "إسلامي", where only the last letter "ة" "Taa Marbutah" /t/ is deleted. The same Arabic word is stemmed into "سلم" using ISRI, where the letter "ا" "Alef" /a:/ is considered as a prefix and an infix is deleted, and the last two letters "ي" "Ya" /j/ and "ة" "Taa Marbutah" /t/ are considered as suffixes are also deleted. On the other hand, the word is stemmed into "إسلام" using Tashaphyne stemmer, which removed the last two letters "ي" "Ya" /j/ and "ة" "Taa Marbutah" /t/.

In our work, we used three different Arabic light stemmers, namely: 1) ARLStem stemmer, 2) ISRI stemmer, and 3) Tashaphyne stemmer.

Finally, once the collected dataset successfully undergoes cleaning, tokenizing, and stemming processes, they must be transformed to feature vectors using TF-IDF method for term weighting in the feature extraction phase, which is discussed in the next section.

4.3 Feature Extraction

Machine learning classification algorithms require a proper presentation of the tweets, where each tweet must be transformed into a feature vector that contains only the distinct words. The feature vector acts as an input to the classifier, which indicates that the good feature vector leads to better performance during the classification phase.

The most common methods for feature extraction (representation) are TF-IDF, BoW, Information Gain (GI), Ratio Gain (RG), and Chi Sequence Statistic (CHI) [73], which enhance the classifier's accuracy.

Both BoW and TF-IDF were used for extracting features in our work. The BoW is similar to TF-IDF, which counts the number of times that the word appears in the text, saves the result in a feature vector, and does not care about the length of the text nor the text order.

TF measures the term/word importance and if the term occurs many times in a tweet, it might be important. However, in IDF if the term/word does not occur a lot in other tweets, it must be considered a rare term/word of a tweet. Thus, a high IDF means the word/term frequency in the tweet is low.

4.4 Classification

The datasets have been collected, tweets have been preprocessed, and have been converted to feature vectors. Feature extraction methods are applied to convert tweets to feature vectors. They can be considered as an input of the classifier, whereby the feature vectors are used to train our classifiers that conduct the classification based on the feature values (training step), and they are tested whether they have hateful content or not (testing step).

Generally, the classification process for each tweet finds the class that the tweet belongs to, uses a binary class for sentiment polarity, namely: hate or not hate (clean). The classification approaches can either be rule-based or learning-based. The learning-based approach uses machine learning supervised classifiers such as SVM, RF, NB, and LR.

SVM is the most powerful and widely used classification algorithm. It is proposed by Vapnik in 1995 as a supervised machine learning, which works as a hyperplane separator between two classes, and tries to maximize the distance between hyperplane margins and the data instances.

Previous studies [17], [21], [22], [74] and [75] have used machine learning classification methods, and others have used evolutionary algorithms like Particle Swarm Optimization (PSO) and Cuckoo search [76], [77] and [78].

In our work, SVM, NB, and RF have been selected to classify tweets, with a training dataset equals to 70% and a testing dataset equals to 30%. The performance of the aforementioned classifiers are evaluated in the evaluation phase that is presented in the next section.

4.5 Evaluation

There are different methods to measure classifiers' performance and efficiency such as precision, recall, F-measure, and accuracy. Precision refers to the fraction of retrieved instances (tweets) that are relevant, which is formulated in Equation (9).

Recall is the fraction of relevant instances (tweets) that are retrieved as shown in Equation (10).

Accuracy refers to the fraction of correct instances (tweets) that have been classified from actual classes as shown in Equation (11).

F- Measure is the mean of precision and recall, as defined in Equation (12).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{9}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{10}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$\text{F} - \text{Measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

Where:

True Positive (TP): refers to a set of tweets that have been classified correctly to the hate category.

False Positive (FP): refers to a set of tweets that have been classified incorrectly and have been said to be related to the hate category incorrectly.

True Negative (TN): refers to a set of tweets that have not been classified into the hate category and they are actually not hate.

False Negative (FN): refers to a set of tweets that have not been classified correctly and have been said to be non-related to the hate category but they are actually hate tweets.

The confusion matrix is used to represent the classifier's performance on the testing dataset. It is a two-dimensional matrix, where each row describes the instances (tweets) in the actual class, and each column describes the instances (tweets) in predicted class, as shown in Table 4.

Table 4: Confusion Matrix for Tweet Classes

	Predicted Hate	Predicted Not Hate
Actual Hate	TP	FN
Actual Not Hate	FP	TN

5. EXPERIMENTAL SETUP AND RESULTS

In this section, all the experimental results are discussed in detail.

5.1 Experimental Setup

Our experiments are illustrated in this step, where a personal computer for running the experiments had been used, with Intel(R) Core(TM) i5-4210U CPU@ 1.70GHz 2.40 GHz processor, 64–bit Windows 8.1.

We also used Python 2.7.12 as the programming language and the library of scikit-learn library in our model implementation.

For our approach, we used a dataset that consists of Arabic tweets, which are classified into two classes, namely: hate or not hate (clean). Given label value equal to 1 in case the class was hate, and 0 if it was not hate (clean). The result of the annotation process of 3000 tweets indicates that 53% of the tweets were not hate and 47% were hate speech.

NLTK library was employed in the data prepossessing step, the "stopwords" tool was used for removing Arabic stop words, and the tokenizer "Word_Tokenize" was used to split the Arabic tweets into tokens/words.

In addition, three different Arabic light stemmers were applied to stem the words to their roots and reduce the data size. The stemmers include ARLStem, ISRI, and Tashaphyne.

TF-IDF and BoW methods were applied to our dataset to extract words/features. The dataset was divided into two sets of training and testing with a percentage of 70% for training and 30% for testing. We also used the same percentage for all classifiers for the purpose of fair comparison between them.

We used the training set to build and train the SVM, NB, and RF classifiers, and evaluated their test results and performance using precision, recall, F-measure, and accuracy.

5.2 Experimental Results and Analysis

In our work, we conducted four different sets of experiments to study the impact of the following: classification algorithm, light stemming, and feature extraction methods on Arabic tweets.

These four experiments were implemented on the same dataset of 3000 tweets. As stated earlier, the tweets were preprocessed, special characters, and the stop-words were removed, then the tweets were divided into tokens/words.

The first experiment was conducted without the applying any stemming method, the second experiment used an Arabic light stemmer called ARLStem, the third experiment used the ISRI stemmer, and the fourth experiment used the Tashpahyne stemmer.

The results from the aforementioned experiments were carried out using three different classification algorithms: RF, NB, and SVM, considering the TF-IDF and BoW as feature extraction methods.

All experimental results were reported in Tables 5, 6, 7, and 8, in terms of precision, recall, F-measure, and accuracy.

Table 5 shows the experimental results of the first experiment that was conducted without using any stemming method, which produced a large number of words/features of about 22,913 words, due to the absence of any stemmer, which normally helps in the features size reduction. Regarding classifiers' performance, it is noticed that SVM obtained the highest results compared to the two other classifiers using the TF-IDF method. The accuracy of SVM classifier was higher than NB and RF classifiers that is equal to 0.83, and the F-measure was also higher that is about 0.80.

Using the BoW as a feature extraction method for both RF and NB classifiers obtained higher experimental results and better performance compared to TF-IDF. RF classifier obtained 0.89, 0.61, 0.72 and 0.78 for precision, recall, F-measure, and accuracy, respectively. On the other hand, NB classifier obtained 0.72, 0.75, 0.74, and 0.75 for precision, recall, F-measure, and accuracy, respectively.

Therefore, the highest precision was achieved when RF classifier used BoW as a feature extraction method and obtained 0.89.

SVM classifier achieved a precision value equal to 0.87 with the use of TF-IDF, and the highest recall was achieved when NB classifier was applied along with the TF-IDF method, which was equal to 0.76.

The experimental results of the second experiment are shown in Table 6, which were obtained using ARLStem stemmer. The words/features are reduced to 10,865 words. With respect to the classifiers' performance results, SVM classifier obtained the highest accuracy, recall, and F-measure values using the TF-IDF method, which were equal to 0.84, 0.76, and 0.81, respectively.

In addition, it was reported that the highest precision value of about 0.88 was obtained when RF classifier was used along with the TF-IDF method.

Furthermore, when RF and NB classifiers used the BoW method, they had higher experimental results compared to TF-IDF. RF classifier obtained

0.85, 0.76, 0.80, and 0.83 for precision, recall, F-measure, and accuracy, respectively. On the other hand, NB classifier obtained 0.72, 0.65, 0.68, and 0.72 for precision, recall, F-measure, and accuracy, respectively. Therefore, RF classifier achieved higher experimental results than NB classifier using the BoW method. However, both RF and SVM classifiers achieved the same recall that is equal to 0.76 when both of them applied the TF-IDF method for feature extraction.

During the third experiment, ISRI stemmer was used and the total number of words/features was reduced to 6,641 words. Based on the experimental results as shown in Table 7, SVM classifier achieved the highest experimental results using TF-IDF method and obtained 0.82, and 0.84 for the F-measure and the accuracy, respectively. However, when SVM classifier used the BoW method, it achieved the highest recall and obtained 0.80.

In addition, RF classifier achieved the highest precision using the TF-IDF method, which was 0.88. It also achieved higher experimental results than NB classifier using both BoW and TF-IDF methods. The combination of the RF classifier with BoW obtained 0.85, 0.75, 0.80, and 0.82 for precision, recall, F-measure, and accuracy, respectively. The combination of the RF classifier with TF-IDF obtained 0.88, 0.71, 0.79, and 0.82 for precision, recall, F-measure, and accuracy, respectively.

On the other hand, NB classifier achieved higher experimental results using BoW than the TF-IDF method, where the precision was 0.68, the recall was 0.52, the F-measure was 0.59, and the accuracy was 0.66.

Finally, the experimental results of the fourth and last experiment are illustrated in Table 8, where the Tashaphyne stemmer and the total number of words/features was reduced to 6,634 words. Based on the experimental results, SVM classifier achieved higher experimental results than RF and NB classifiers using TF-IDF method and obtained 0.83, 0.76, 0.80, and 0.82 for precision, recall, F-measure, and accuracy, respectively.

On the other hand, it was noticed that when RF classifier is combined with the TF-IDF method, it obtained a precision of 0.84, which was a little higher than the precision of the SVM classifier, which obtained 0.83. However, RF classifier had better experimental results using TF-IDF as a feature extraction method. In addition, it performing better than NB classifier and obtained 0.84, 0.68, 0.75, and 0.79 for precision, recall, F-measure, and accuracy, respectively.

When NB classifier was used with TF-IDF method, it obtained 0.69, 0.62, 0.66, and 0.70 for precision, recall, F-measure, and accuracy, respectively. It also obtained almost the same results when using with the BoW method, which obtained 0.72, 0.59, 0.65, and 0.70 for precision, recall, F-measure, and accuracy, respectively.

Based on our experimental results, we can concluded that SVM classifier dominated RF and NB classifiers as classification algorithms. SVM classifier outperformed RF and NB classifiers, and obtained higher experimental results based on the evaluation metrics used in this work. In addition, for both BoW and TF-IDF as feature extraction methods, SVM classifier achieved better experimental results when combined with TF-IDF compared to BoW. Furthermore, RF classifier was the second-best classifier compared with NB classifier.

Taking into consideration both BoW and TF-IDF as feature extraction methods, it was found that TF-IDF outperformed BoW allowing SVM classifier to achieve the highest accuracy, while NB classifier has always performed better using BoW. For RF classifier, there were some cases that BoW method outperformed TF-IDF when non-stemmer and ARLStem were used. Furthermore, TF-IDF performed better than BoW when Tashaphyne stemmer was used and almost performed the same when using ISRI stemmer.

Table 5: Classifiers' Performance without Using Any Stemmer

Algorithm	Word Feature	Precision	Recall	F-Measure	Accuracy
RF	BoW	**0.89**	0.61	0.72	0.78
	TF-IDF	0.88	0.57	0.69	0.77
NB	BoW	0.72	0.75	0.74	0.75
	TF-IDF	0.69	**0.76**	0.73	0.73
SVM	BoW	0.84	0.75	0.79	0.82

	TF-IDF	0.87	0.74	**0.80**	**0.83**

Table 6: Classifiers' Performance Using ARLStem Stemmer

Algorithm	Word Feature	Precision	Recall	F-Measure	Accuracy
RF	BoW	0.85	0.76	0.80	0.83
	TF-IDF	**0.88**	0.71	0.79	0.82
NB	BoW	0.72	0.65	0.68	0.72
	TF-IDF	0.68	0.68	0.68	0.70
SVM	BoW	0.84	0.75	0.79	0.82
	TF-IDF	0.87	**0.76**	**0.81**	**0.84**

Table 7: Classifiers' Performance Using ISRI Stemmer

Algorithm	Word Feature	Precision	Recall	F-Measure	Accuracy
RF	BoW	0.85	0.75	0.80	0.82
	TF-IDF	**0.88**	0.71	0.79	0.82
NB	BoW	0.68	0.52	0.59	0.66
	TF-IDF	0.64	0.56	0.60	0.65
SVM	BoW	0.81	**0.80**	0.80	0.82
	TF-IDF	0.86	0.78	**0.82**	**0.84**

Table 8: Classifiers' Performance Using Tashaphyne Stemmer

Algorithm	Word Feature	Precision	Recall	F-Measure	Accuracy
RF	BoW	0.81	0.70	0.75	0.78
	TF-IDF	**0.84**	0.68	0.75	0.79
NB	BoW	0.72	0.59	0.65	0.70
	TF-IDF	0.69	0.62	0.66	0.70
SVM	BoW	0.81	0.76	0.78	0.80
	TF-IDF	0.83	**0.76**	**0.80**	**0.82**

Regarding the stemming methods, the experimental results showed that the best results in terms of precision, recall, F-measure, and accuracy, were achieved when ARLStem stemmer was used along with SVM as a classifier.

In addition, ISRI stemmer achieved experimental results that are similar to ARLStem when SVM classifier was used with the TF-IDF method. However, RF classifier achieved best experimental results using ARLStem stemmer, but NB classifier achieved the highest experimental results when no stemmer was used. Tashaphyne stemmer had similar results of NB classifier in both BoW and TF-IDF methods, but it reduced the features size better than the other two stemmers.

6. CONCLUSIONS AND FUTURE WORK

Recently, the use of OSN especially in the Middle East, has been evolving. OSN can be considered as the suitable platforms and atmospheres where people can express their opinions and reviews of a certain topic on OSN such as Facebook or Twitter platforms. The detection of hate speech in Arabic social media is not an easy task, which requires finding features that extract the meaning of the text, NLP techniques for text preprocessing, and machine learning classification algorithms.

In this research, we aimed to detect hate speech in Arabic tweets, which has not been widely investigated previously. We were able to create a new dataset of 45,000 tweets, which were sampled to 3000 tweets that were annotated and labeled manually into two classes, namely: hate or not hate (clean). The labeled tweets were then pre-processed to enhance the classification accuracy.

In addition, supervised classification algorithms were applied along with BoW and TF-IDF as the feature extraction methods, which produce word feature vectors that serve as an input to the classifiers.

In our work, all experimentations were conducted using RF, NB, and SVM classifiers. Based on our experimental results, SVM classifier achieved the highest accuracy.

For future work, we suggest enhancing the accuracy of the SVM classifier by combing it with evolutionary methods that could help in reducing the feature size. We also recommend the use of other feature extraction methods such as N-grams and word embedding together with RF, NB, and SVM classifiers. In addition, collecting large volume of tweets that contain hate and not hate tweets in order to enlarge our dataset for training and testing purposes, which can be distributed publicly to the research community.

REFERENCES:

[1] V. Kharde, P. Sonawane et al., "Sentiment analysis of twitter data: a survey of techniques," arXiv preprint arXiv:1601.06971, 2016.

[2] I. Aljarah, M. Habib, N. Hijazi, H. Faris, R. Qaddoura, B. Hammo, M. Abushariah, and M. Alfawareh, "Intelligent detection of hate speech in arabic social network: A machine learning approach," Journal of Information Science, p. 016555152091765, 05 2020.

[3] H. S. Ibrahim, S. M. Abdou, and M. Gheith, "Sentiment analysis for modern standard Arabic and colloquial," arXiv preprint arXiv:1505.03105, 2015.

[4] Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on twitter using a convolution-GRU based deep neural network," in Lecture notes in computer science. Springer Verlag, 2018.

[5] A. Soudi, A. Farghaly, G. Neumann, and R. Zbib, Challenges for Arabic machine translation. John Benjamins Publishing, 2012, vol. 9.

[6] F. Salem, "Social media and the internet of things towards data-driven policymaking in the Arab world: potential, limits and concerns," The Arab Social Media Report, Dubai: MBR School of Government, vol. 7, 2017.

[7] A. Farghaly and K. Shaalan, "Arabic natural language processing: Challenges and solutions," ACM Transactions on Asian Language Information Processing (TALIP), vol. 8, no. 4, pp. 1–22, 2009.

[8] G. Alwakid, T. Osman, and T. Hughes-Roberts, "Challenges in sentiment analysis for Arabic social networks," Procedia Computer Science, vol. 117, pp. 89–100, 2017.

[9] S. O. Alhumoud, M. I. Altuwaijri, T. M. Albuhairi, W. M. Alohaideb et al., "Survey on Arabic sentiment analysis in twitter," International Science Index, vol. 9, no. 1, pp. 364–368, 2015.

[10] S. Ahmed, M. Pasquier, and G. Qadah, "Key issues in conducting sentiment analysis on Arabic social media text," in Innovations in Information Technology (IIT), 2013 9th International Conference on. IEEE, 2013, pp. 72–77.

[11] R. M. Duwairi, R. Marji, N. Sha'ban, and S. Rushaidat, "Sentiment analysis in Arabic tweets," in 2014 5th International Conference on Information and Communication Systems (ICICS). IEEE, 2014, pp. 1–6.

[12] N. Farra, E. Challita, R. A. Assi, and H. Hajj, "Sentence-level and document-level sentiment mining for Arabic texts," in 2010 IEEE international conference on data mining workshops. IEEE, 2010, pp. 1114–1119.

[13] W. Warner and J. Hirschberg, "Detecting hate speech on the world wide web," in Proceedings of the Second Workshop on Language in Social Media. Association for Computational Linguistics, 2012, pp. 19–26.

[14] I. Kwok and Y. Wang, "Locate the hate: Detecting tweets against blacks." in AAAI, 2013.

[15] N. D. Gitari, Z. Zuping, H. Damien, and J. Long, "A lexicon-based approach for hate speech detection," International Journal of Multimedia and Ubiquitous Engineering, vol. 10, no. 4, pp. 215–230, 2015.

[16] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati, "Hate speech detection with comment embeddings," in Proceedings of the 24th international conference on world wide web. ACM, 2015, pp. 29–30.

[17] P. Burnap and M. L. Williams, "Cyber hate speech on Twitter: An application of machine classification and statistical modeling for policy and decision making," Policy & Internet, vol. 7, no. 2, pp. 223–242, 2015.

[18] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," arXiv preprint arXiv:1703.04009, 2017.

[19] L. A. Silva, M. Mondal, D. Correa, F. Benevenuto, and I. Weber, "Analyzing the targets of hate in online social media." in ICWSM, 2016, pp. 687–690.

[20] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? predictive features for hate speech detection on twitter," in Proceedings of the NAACL student research workshop, 2016, pp. 88–93.

[21] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in Proceedings of the 26th International Conference on World Wide Web Companion. International World Wide Web Conferences Steering Committee, 2017, pp. 759–760.

[22] S. Malmasi and M. Zampieri, "Detecting hate speech in social media," arXiv preprint arXiv:1712.06427, 2017.

[23] A. Schmidt and M. Wiegand, "A survey on hate speech detection using natural language processing," in Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, 2017, pp. 1–10.

[24] A. Alakrot, L. Murray, and N. S. Nikolov, "Towards accurate detection of offensive language in online communication in Arabic," Procedia computer science, vol. 142, pp. 315–320, 2018.

[25] H. Mohaouchane, A. Mourhir, and N. S. Nikolov, "Detecting offensive language on Arabic social media using deep learning," in 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS). IEEE, 2019, pp. 466–471.

[26] H. Mubarak and K. Darwish, "Arabic offensive language classification on Twitter," in International Conference on Social Informatics. Springer, 2019, pp. 269–276.

[27] J. Karoui, F. B. Zitoune, and V. Moriceau, "Soukhria: Towards an irony detection system for Arabic in social media," Procedia Computer Science, vol. 117, pp. 161–168, 2017.

[28] L. Moudjari and K. Akli-Astouati, "An embedding-based approach for irony detection in Arabic tweets," in Proceedings of the [email protected] In Metha P., Rosso P., Majumder P., Mitra M.(Eds.) Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2019). CEUR Workshop Proceedings, 2019.

[29] B. Ghanem, J. Karoui, F. Benamara, V. Moriceau, and P. Rosso, "Idat at fire2019: Overview of the track on irony detection in Arabic tweets," in Proceedings of the 11th Forum for Information Retrieval Evaluation, 2019, pp. 10–13.

[30] I. Weber, V. R. K. Garimella, and A. Batayneh, "Secular vs. Islamist polarization in Egypt on Twitter," in Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. ACM, 2013, pp. 290–297.

[31] H. Mubarak, K. Darwish, and W. Magdy, "Abusive language detection on Arabic social media," in Proceedings of the First Workshop on Abusive Language Online, 2017, pp. 52–56.

[32] T. De Smedt, G. De Pauw, and P. Van Ostaeyen, "Automatic detection of online jihadist hate speech," arXiv preprint arXiv:1803.04596, 2018.

[33] N. Albadi, M. Kurdi, and S. Mishra, "Are they our brothers? analysis and detection of religious hate speech in the Arabic Twitter sphere," in 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). IEEE, 2018, pp. 69–76.

[34] N. Albadi, M. Kurdi, and S. Mishra, "Investigating the effect of combining GRU neural networks with handcrafted features for religious hatred detection on Arabic twitter space," Social Network Analysis and Mining, vol. 9, no. 1, p. 41, 2019.

[35] A. Siegel, Sectarian, "Twitter Wars: Sunni-Shia Conflict and Cooperation in the Digital Age,". Carnegie Endowment for International Peace, 2015, vol. 20.

[36] A. Siegel, J. Tucker, J. Nagler, and R. Bonneau, "Socially mediated sectarianism," Unpublished Manuscript, 2018.

[37] A. Al-Hassan and H. Al-Dossari, "Detection of hate speech in social networks: a survey on multilingual corpus," in 6th International Conference on Computer Science and Information Technology, 2019.

[38] A. Ghosh Chowdhury, A. Didolkar, R. Sawhney, and R. Ratn Shah, "Beyond hostile linguistic cues: The gravity of online milieu for hate speech detection in Arabic," in Proceedings of the 30th ACM Conference on Hypertext and Social Media, 2019, pp. 285–286.

[39] H. Haddad, H. Mulki, and A. Oueslati, "T-HSAB: A Tunisian hate speech and abusive dataset," in International Conference on Arabic

Language Processing. Springer, 2019, pp. 251–263.

[40] H. Mulki, H. Haddad, C. B. Ali, and H. Alshabani, "L- HSAB: A Levantine twitter dataset for hate speech and abusive language," in Proceedings of the Third Workshop on Abusive Language Online, 2019, pp. 111–118.

[41] H. P. Su, Z.J. Huang, H.T. Chang, and C.J. Lin, "Rephrasing profanity in Chinese text," in Proceedings of the First Workshop on Abusive Language Online, 2017, pp. 18–24.

[42] S. Tulkens, L. Hilte, E. Lodewyckx, B. Verhoeven, and W. Daelemans, "A dictionary-based approach to racism detection in dutch social media," arXiv preprint arXiv:1608.08738, 2016.

[43] B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, and M. Wojatzki, "Measuring the reliability of hate speech annotations: The case of the European refugee crisis," arXiv preprint arXiv:1701.08118, 2017.

[44] T. Santosh and K. Aravind, "Hate speech detection in Hindi-English code-mixed social media text," in Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, 2019, pp. 310–313.

[45] N. Aulia and I. Budi, "Hate speech detection on Indonesian long text documents using machine learning approach," in Proceedings of the 2019 5th International Conference on Computing and Artificial Intelligence, 2019, pp. 164–169.

[46] F. Del Vigna12, A. Cimino23, F. Dell Orletta, M. Petrocchi, and M. Tesconi, "Hate me, hate me not: Hate speech detection on Facebook," in Proceedings of the First Italian Conference on Cybersecurity (ITASEC17), 2017, pp. 86–95.

[47] K. Soumya George and S. Joseph, "Text classification by augmenting bag of words (bow) representation with co-occurrence feature," IOSR J. Comput. Eng, vol. 16, no. 1, pp. 34–38, 2014.

[48] C. Cortes and V. Vapnik, "Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273–297, 1995.

[49] A. Moh'd Mesleh, "Support vector machines based Arabic language text classification system: feature selection comparative study," in Advances in Computer and Information Sciences and Engineering. Springer, 2008, pp. 11–16.

[50] B. Y. Pratama and R. Sarno, "Personality classification based on twitter text using Naive Bayes, KNN and SVM," in 2015 International Conference on Data and Software Engineering (ICoDSE). IEEE, 2015, pp. 170– 174.

[51] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5–32, 2001.

[52] P. Burnap and M. L. Williams, "Us and them: identifying cyber hate on Twitter across multiple protected characteristics," EPJ Data Science, vol. 5, no. 1, p. 11, 2016.

[53] R. Magu, K. Joshi, and J. Luo, "Detecting the hate code on social media," arXiv preprint arXiv:1703.05443, 2017.

[54] A. A. Kumar and S. Chandrasekhar, "Text data pre-processing and dimensionality reduction techniques for document clustering," International Journal of Engineering Research and Technology (IJERT), vol. 1, 2012.

[55] T. Buckwalter, "Buckwalter Arabic morphological analyzer version 1.0," Linguistic Data Consortium, University of Pennsylvania, 2002.

[56] L. S. Larkey and M. E. Connell, "Arabic information retrieval at umass in trec-10," in TREC, 2001.

[57] M. Diab, K. Hacioglu, and D. Jurafsky, "Automatic tagging of Arabic text: From raw text to base phrase chunks," in Proceedings of HLTNAACL 2004: Short papers. Association for Computational Linguistics, 2004, pp. 149–152.

[58] R. Nelken and S. M. Shieber, "Arabic diacritization using weighted finite-state transducers," in Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages. Association for Computational Linguistics, 2005, pp. 79–86.

[59] M. A. Attia, "Arabic tokenization system," in Proceedings of the 2007 workshop on computational approaches to Semitic languages: Common issues and resources. Association for Computational Linguistics, 2007, pp. 65–72.

[60] M. M. Syiam, Z. T. Fayed, and M. B. Habib, "An intelligent system for Arabic text categorization," International Journal of Intelligent Computing and Information Sciences, vol. 6, no. 1, pp. 1–19, 2006.

[61] A. Wahbeh, M. Al-Kabi, Q. Al-Radaideh, E. Al-Shawakfa, and I. Alsmadi, "The effect of stemming on Arabic text classification: an empirical study," International Journal of Information Retrieval Research (IJIRR), vol. 1, no. 3, pp. 54–70, 2011.

[62] M. Hadni, S. A. Ouatik, and A. Lachkar, “Effective Arabic stemmer based hybrid approach for Arabic text categorization,” International Journal of Data Mining & Knowledge Management Process, vol. 3, no. 4, p. 1, 2013.

[63] S. Khoja and R. Garside, “Stemming Arabic text,” Lancaster, UK, Computing Department, Lancaster University, 1999.

[64] L. S. Larkey, L. Ballesteros, and M. E. Connell, “Light stemming for Arabic information retrieval,” in Arabic computational morphology. Springer, 2007, pp. 221–243.

[65] M. Y. Dahab, A. Ibrahim, and R. Al-Mutawa, “A comparative study on Arabic stemmers,” International Journal of Computer Applications, vol. 125, no. 8, 2015.

[66] K. Taghva, R. Elkhoury, and J. Coombs, “Arabic stemming without a root dictionary,” in International Conference on Information Technology: Coding and Computing (ITCC’05)-Volume II, vol. 1. IEEE, 2005, pp. 152–157.

[67] L. Khreisat, “Arabic text classification using n-gram frequency statistics a comparative study.” DMIN, vol. 2006, pp. 78–82, 2006.

[68] S. H. Mustafa and Q. A. Al-Radaideh, “Using n-grams for Arabic text searching,” Journal of the American Society for Information Science and Technology, vol. 55, no. 11, pp. 1002–1007, 2004.

[69] T. Zerrouki, “Tashaphyne, Arabic light stemmer/segment,” 2010.

[70] M. El-Defrawy, Y. El-Sonbaty, and N. A. Belal, “A rule-based subject correlated Arabic stemmer,” Arabian Journal for Science and Engineering, vol. 41, no. 8, pp. 2883–2891, 2016.

[71] M. El-Defrawy, Y. El-Sonbaty, and N. Belal, “Enhancing root extractors using light stemmers,” in Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation: Posters, 2015, pp. 157–166.

[72] K. Abainia, S. Ouamour, and H. Sayoud, “A novel robust Arabic light stemmer,” Journal of Experimental & Theoretical Artificial Intelligence, vol. 29, no. 3, pp. 557–573, 2017.

[73] A. Sharma and S. Dey, “A comparative study of feature selection and machine learning techniques for sentiment analysis,” in Proceedings of the 2012 ACM research in applied computation symposium. ACM, 2012, pp. 1–7.

[74] P. Burnap and M. L. Williams, “Hate speech, machine classification and statistical modeling of information flows on twitter: Interpretation and communication for policy decision making,” 2014.

[75] Z. Waseem, “Are you a racist or am I seeing things? annotator influence on hate speech detection on twitter,” in Proceedings of the first workshop on NLP and computational social science, 2016, pp. 138–142.

[76] L. Shang, Z. Zhou, and X. Liu, “Particle swarm optimization-based feature selection in sentiment classification,” Soft Computing, vol. 20, no. 10, pp. 3821–3834, 2016.

[77] M. Mafarja, I. Aljarah, A. A. Heidari, A. I. Hammouri, H. Faris, A.- Z. Ala’M, and S. Mirjalili, “Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems,” Knowledge-Based Systems, 2017.

[78] S. Rajamohana, K. Umamaheswari, and S. V. Keerthana, “An effective hybrid cuckoo search with harmony search for review spam detection,” in Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), 2017 Third International Conference on. IEEE, 2017, pp. 524–52

[79] W. T. Aung, Y. Myanmar and K. H. M. Saw Hla, "Random forest classifier for multi-category classification of web pages," 2009 IEEE Asia-Pacific Services Computing Conference (APSCC), Singapore, 2009, pp. 372-376.

[80] J. Hartmann, J. Huppertz, C. Schamp, and M. Heitmann, “Comparing automated text classification methods,” International Journal of Research in Marketing, vol. 36, no. 1, pp. 20–38, 2019.