Sentiment Classification and Analysis on Arabic using Semi-Supervised Approach

[1]Khalid Bashir Bajwa, [2]Waqas Nawaz, [3]Qaiser Abbas, [4]AbdulAziz Almuzaini and [5]Rafi Ahmad Khan (Corresponding Author)
[1235]Assistant Professor, [4]Lecturer
[12345] Faculty of Computer and Information System, Islamic University of Madinah, Madinah, Kingdom of Saudi Arabia
[5]Department of Management Studies, University of Kashmir, India

Abstract: Classification and analysis of Arabic text in terms of social media analysis is presented here in this work. This work, keeping in view the morphological and syntactical difficulties in Arabic language, focuses on negative and positive sentiment candidates for polarity as a first step towards a major goal of building a complete framework for political sentiment analysis. There are many works for sentiment classification and analysis in literature, which can be categorized into supervised, unsupervised, and hybrid. However, majority of these methods either lack in availability of sufficient size dataset for learning or suffer from Arabic language complications in terms of processing diverse nature of text. To overcome this issue, we introduce a semi-supervised approach for sentiment classification and analysis in Arabic language. Our approach is based on the concept of word embedding model to improve the performance of classification even with small size seed data. It has the capability to model the words in a large vector space where similar words are expected to occur in close proximity. Once the data is mapped to vector space model, then we utilize various classifiers to learn the patterns in the lexicon and predict the classification as positive or negative for unknown similar words. Classifiers such as Stochastic Gradient Descent and SVM are trained and tested with the specified 80% and 20% data ratio. Our approach yields around 80% accuracy through intermediate experiments. It has been observed that besides the common problem of lexicon size, this is attributed mainly to the quality of word embedding available for the Arabic language.

Index Terms **- Sentiment Analysis, Classification, Tweets, Polarity, Arabic Language, Lexicon, Text Processing**

I. INTRODUCTION

Arabic is a central Semitic language of Arab world and has more than 300 million native speakers [1]. Arabic has a vast history among regions of Middle East and has different dialects including the language of Quran (Holy book of God). Arabic has a complex and unusual morphology (i.e. method of constructing words from a basic root). Arabic has a non-concatenative "root-and-pattern" morphology: A root consists of a set of bare consonants (usually three), which are fitted into a discontinuous pattern to form words. For example, the word for 'I wrote' is constructed by combining the root k-t-b 'write' with the pattern -a-a-tu 'I Xed' to form katabtu 'I wrote'. Other verbs meaning 'I Xed' will typically have the same pattern but with different consonants, e.g. qaratu 'I read', akaltu 'I ate', dhahabtu 'I went', etc. Other complex and unusual patterns are also possible in Arabic. These complications make Arabic language difficult to analyze and process.

Sentiment Analysis is the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral. A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level. Whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, "beyond polarity" sentiment classification looks, for instance, at emotional states such as "angry", "sad", and "happy". A thorough investigation of the available literature revealed that the works were mainly concentrated on dealing with specific Sentiment Analysis tasks. To this end, the researchers developed three different approaches, namely supervised (or corpus-based), unsupervised (or sometimes referred as lexicon-based) and hybrid, refer to these recent surveys for details [2, 3]. The results that these studies achieved are interesting but divergent [3].

The Arabic-specific challenges are mainly caused by Arabic morphological complexity, limited resources and dialects. While the general linguistic issues include polarity fuzziness, polarity strength, implicit sentiment, sarcasm, spam, review quality and domain dependence [4, 5]. To overcome the issue of morphological complexity, pre-processing step needs to be performed with care. Lexicon and candidate lexical item s are converted to their root forms to get rid of morphological issues. Thus the final lexicon has candidate sentiments along with their root forms. Similarly, the results are immature when polarity has only negative and positive counts [5]. An attempt is made to dim the effect of fuzziness in polarity by introducing a neutral count along with the negative and positive counts.

These days Lexicon are usually available online and majority of them are very small in size for Arabic Language. To make it clean, pre-processing is performed first and then filtering of candidate sentiments is carried out along with the conversion towards their root forms. At the end of this phase, a final refined lexicon is produced, which is then used to count the polarity (negative, positive and neutral) from the text. The text data set contains the feedback of public about different products and our goal is to find the polarity of sentences through sentence classifier. The text data/corpus is divided into standard division including 80% training data and 20% test data. The classification accuracy of the classifiers is low through lexicons in these approaches because of limited number of lexicons for training. The reason is that there are many words which are not available in the lexicon and hence their sentiment cannot be determined. A trivial solution to this problem would be to manually label huge amounts of training data and

extract a lexicon based on the training data, which is somewhat done in [6]. A better alternate approach is required to do this tedious task.

Our work in this study tends to be hybrid in nature, since we extend an existing lexicon through corpus based word embedding model. We use an Arabic gold standard lexicon for sentiment analysis having positive and negative words. An interesting idea developed recently in [7] is to use word embedding which represent words as vectors. The embedding tends to represent similar words with vectors which are close together in the vector space. Using this approach, we can extend our lexicon to the words which we have not seen before but are part of the embedding. The next phase is training the classifier. Various classifiers at sentence level are used to classify the polarity of sentences lying in the text corpus of social data. It is non-trivial to decide best classifier therefore we determine through experiments. The classifier is expected to be trained with the 80% training data, which has the words along with their polarity counts and classification. Finally, 20% test data is to be applied on the classifier and results are being evaluated. Rest of the article is organized as follows. We discuss related studies in Sec. II and the methodology of our solution to aforementioned problem is outlined in Sec III. The results and conclusion follow in Sec. IV and V.

II. LITERATURE REVIEW

Sentiment analysis is a well-studied topic in literature and discusses various aspects of analyzing textual data to extract useful information [8]. The reader may refer to the relevant surveys [2, 3, 8] for more details. However, this work emphasis on analyzing textual data in Arabic language, therefore, we discuss various existing approaches to process and analyze Arabic text for sentiment analysis. There are two approaches, lexicon-based and corpus-based, commonly used for sentiment analysis in Arabic context [6].

Lexicon-based approaches

Lexicon-based approach is relatively simple and straight forward because it takes help from predefined words (or lexicons). The sentiment of those lexicons are directly extracted from dictionary as positive or negative. The lexicon is developed either manually or automatically [9]. The key issue with lexicon-based approach is to deal with expansion of a limited sized lexicon to a broad vocabulary. In addition, complex structure of the Arabic language with various dialects also present a problem. A lexicon for one dialect of Arabic may not work properly for other dialects of the same language. Another critical aspect, in terms of processing and evaluating Arabic language, is the difference of the language used on social media platform compared with the standard written language. This further complicates the sentiment analysis task. Many studies in literature highlight this aspect and suggested alternate approaches.

Corpus based approaches

Corpus-based approach, which is also known as supervised approach, relies on various machine learning classifiers including Naive Bayes (NB), Support Vector Machines (SVM), Decision Tree, K-Nearest Neighbors (KNN) etc. on annotated dataset. This problem is somewhat similar to text classification problem [9], where we usually prepare a pair of datasets one for training and the other for testing. The training dataset needs to be annotated manually in terms of sentiment polarity, i.e. positive or negative. The classifiers usually learn and develop a model from the training data and then predict the polarity of testing data. We quantify the performance of such approaches by analyzing the errors made by the classifier. Therefore, it is essential to use a huge corpus for better accuracy.

Hybrid approaches

In order to overcome the issues and challenges with lexicon-based and corpus-based approaches, researcher have also suggested hybrid approaches for sentiment analysis [10]. A comparative analysis is also carried out for both lexicon-based and corpus-based approaches in [6]. A manually annotated dataset is developed initially in this study and then developed a lexicon which is further compared with corpus based approaches.

In contrary to all of the above mentioned techniques, we introduce a semi supervised approach to extend the existing limited lexicon through word embedding model, i.e. AraVec [7].

III. METHODOLOGY

Our approach to sentiment analysis is a combination of lexicon-based and corpus-based methodologies. We utilize the concept of word embedding in our approach. The basic idea is to map the existing limited number of lexicons to a vector space where similar words are mapped to nearby space. Later, we use various classifiers to learn and detect the classification of those word embedding towards sentiment analysis. The detail of our approach is presented in the subsequent sections.

Pre-processing

We perform pre-processing on input text to cleanse text. The following steps are taken

- Removal of compound words
- Removal of special characters
- Normalization of letters, for instance replacement of ‘ﻭﻭ’ with ‘ﻭ’ etc,

Acquisition of Lexicon

We have acquired two lexicons, which include Arabic translation of the Bing Liu Lexicon (AT-BLL) and the Arabic translation of the MPQA Subjectivity Lexicon (AT-MPQA-SL). The details of both lexicons can be found in [11] and [12]. The format of the first lexicon is given in Figure 1 includes four columns. First column with English Term contains the lexical items from the Bing Liu Lexicon. Second column Arabic Translation consists of translated version of lexical items available in the previous column English

Term. Third is the Buckwalter column, containing the transliteration of Arabic Translation entries. Similarly, Sentiment is the last column with the score of lexical items in the Bing Liu Lexicon [13]. This score is +1 for positive sentiment and -1 for negative sentiment. The size of lexicon is 6789 tokens (different lexical items) from which 2006 are positive sentiment candidates and 4783 are the negative sentiment candidates. Some cleaning of the sentiment lexicon was required to remove compound words.

[English Term]	[Arabic Translation]	[Buckwalter]	[Sentiment]
abound	تكثر	tkvr	1
abundant	كثير	kvyr	1

Figure 1: Arabic translation of the Bing Liu Lexicon

Figure 2, represents the format of the second Arabic translated lexicon from MPQA Subjectivity Lexicon [14], with additions. From left to right, column type represents the strong or weak subjectivity of clue in words while the column len gives us the length of the clue in words. The columns with word1 and pos1 represent the token and part-of-speech of the clue, respectively. The column with variable stemmed1 can take yes/no (y/n) values depending on the clue available in variable word1. Next is the prior polarity, which is divided into four categories including positive, negative, neutral and both. Finally, are the two additional columns for Arabic translation and transliteration of the clue, respectively. We consider only the positive and negative sentiments in this study. The total size of the lexicon is 8199 tokens, which are then distributed into 2,718 positive tokens and 4,911 negative tokens.

type=weaksubj	len=1	word1=abandoned	pos1=adj	stemmed1=n	priorpolarity=negative	ar=مهجور	bw=mhjwr
type=weaksubj	len=1	word1=abandonment	pos1=noun	stemmed1=n	priorpolarity=negative	ar=هجر	bw=hjr
type=weaksubj	len=1	word1=abandon	pos1=verb	stemmed1=y	priorpolarity=negative	ar=التخلي عن	bw=Altxly_En

Figure 2: Arabic translation of the MPQA Subjectivity Lexicon

Word embedding model

We use these Arabic gold-standard lexicons (AT-BLL & AT-MPQA-SL) for sentiment analysis having positive and negative words only. Since, the size of the lexicons is limited to a couple of thousand, there are many words, which are not available in the lexicons, and hence their sentiment cannot be determined. One work around of this problem is to manually label huge amounts of training data and extract a lexicon based on this training data. However, this process is cumbersome and requires a lot of human effort. A workaround to this is to use a word-embedding model named AraVec [7], which represent words as vectors. The word embedding tends to represent similar words with vectors, which are close together in the vector space. Using this approach, we can extend our Arabic gold-standard lexicons to the words, which we have not seen before and our Arabic gold-standard lexicons would become the part of embedding model AraVec.

AraVec is open source and is used for word distribution representation. It is based on vector space models like word2vec, which can generate vectors for the tokens of a corpus. AraVec was developed using the Gensim tool and it was then trained on data collected from Tweets (vocabulary size 331,679), Wikipedia (vocabulary size 162,516), and World Wide Web (vocabulary size 234,961). The total number of words in corpora of AraVec is reported to be more than 3,300,000,000, however, after adding up the given vocabulary sizes of corpora, AraVec has 729,156 tokens.

Tokens from Arabic Gold Standard (AGS) lexicon are input to AraVec to find and collect their respective vectors. In this way, we end up with a new Arabic lexicon with vectors (ALV) for the manually input tokens. The dimension set in AraVec for vectors representation is 300, which can be more up to N dimension but the standard inferred from empirical study is that the dimension limit 100-300 is better in producing quality result. As AraVec is based on word2vec model, its structure is explained in Figure 4 as follows. An input vector given on the left hand side of the Figure 4 is a vocabulary-sized vector, which is 331,679 for the Tweets corpus, 162,516 for the corpus of Wikipedia, and 234,961 for the WWW corpus.

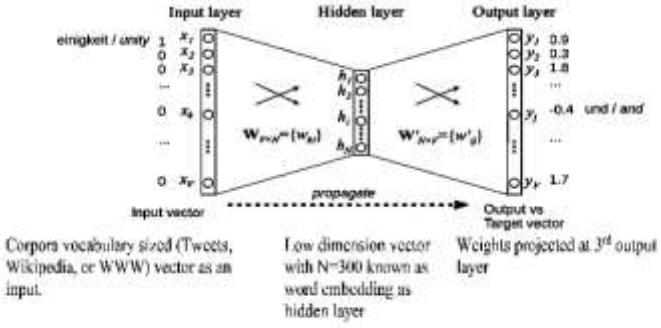

Figure 4: Structure of AraVec based on word2vec model [21].

For our experimentation, we have configured its two methodologies known as Continuous Bag of Words (CBOW) and Skip Gram [22] in AraVec. In CBOW, context words (a form of multiple words) can be given as an input vector and then the model can predict the one most probable target word at the output layer, while the Skip Gram model is the opposite of it. In it, one word is given as an input vector and it can then predict the most likely words at the output layer.

Using AraVec we map the word to vectors and extract similar words based on their similarity in the vector space. Figure 5, shows similar words to "جيد" from AraVec with similarity index (0—1) shown in the first column. As we can see most of the similar words returned have the same polarity as the input word. There are some exceptions though, for instance "سيئ".

```
0.5705305337905884 وجيد
0.529331386089325 ممتاز
0.5098779797554016 وممتاز
0.4738933742046356 سيئ
0.45213863253593445 جيدجدا
0.43476083874702454 مقبول
0.4263805150985718 حقي
0.4177446961402893 بتقدير
0.4173172116279602 محفز
0.4146695137023926 وايجابي
```

Figure 5: Similar words from

Learning phase: Selection of classifier

Due to complex nature of Arabic morphology in terms of derivation and inflection, our work is focused on sentiment analysis, which is the identification of positive and negative polarity. The AGS has this identification of positive and negative polarity discussed in Section 3.2 and on the other hand our model adopted, presented in Section 3.3 has the capability to infer similar words of a given word. This is the backbone of our work that we are going to utilize both of these ideas to predict sentiment words discussed as follows. There are different classifiers available like Max Entropy [15], SVM [16], Nave Bayes [17], Stochastic Gradient Descent (SGD) [18], etc. Every classifier has its own merits and demerits. For example, Nigam et al. [19] concluded that Max Entropy sometimes performs better than Naive Bayes but not always and only for standard text classification of features/class. It also does not care about the relationship between features. Joachims [20] advocates the supremacy of SVM classifier over the Naive Bayes and Max Entropy models for text classification if the probabilities are involved like in our case of AraVec use discussed in Section 3.3. SGD and SVM lie under the same umbrella of linear classification and are being used successfully for large scale and sparse machine learning problems for text classification. These all reasons convinced us to use SGD and SVM classifiers to evaluate our work.

A classifier is trained on the gold standard lexicon by transforming the words in the lexicon to their embedding (Vectors). The classifier learns how the vectors which in effect are words, are mapped to positive and negative sentiments. Now when a sentence is provided which may have words not present in our lexicon then the sentence is first tokenized and the individual words are vectorized based on the word embedding. The pre-trained classifier on the gold standard lexicon is then used to compute the scores for each of the words in the sentence. The scores are then added up to compute the final score for the sentence. This method works great if the word embedding gives us good vector representation having similar words close together. This expands our system to predict the sentiment for those words which the lexicon never had.

The libraries for SGD and SVM are available as an open source. We have downloaded and modified these libraries as per our requirement from the SciKit Learn website. The AGS lexicon is divided into 80% training data and 20% test data. After the configuration of SGD and SVM, the training of these classifiers has been done on the AGS lexicon and AraVec is made accessible to these classifiers for the extended version of AGS. At this point, the classifiers are able to classify the polarity of input tokens in terms of positive/negative class, which is depicted in Figure 6. The positive values for tokens in Figure 6 represent positive polarity and negative values represent negative polarity.

After this stage, we are in a state to test our 20% test data. One final point, which is about the working of classifiers, is that after taking input token (single or multiple), the classifiers map the input into AGS lexicon. If found then the polarity of the input token can be identified directly from the AGS as it is available there, but what would happen in case of unknown tokens. For unknown tokens, the classifiers then can have access to already trained extended part of AGS (ten times larger than AGS) in AraVec. If the unknown token is found into the extended part of AGS (without polarity) then it means we have its vector representation within it. At this point, our classifiers use the AraVec utility of finding the similar words for the unknown token. Those similar words are then mapped again with the AGS to find the polarity, unveil the unknown token and if some similar word is found then its respective polarity is reported as can be seen in Figure 6 for some of the unknown tokens of held out data.

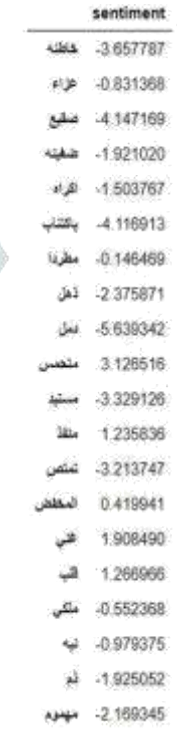

Figure 6: Result of sentiment prediction for test data. +ve values indicates a positive sentiment and –ve a negative sentiment

IV. RESULTS

The results of our experiments are depicted as under. The experimental setup involves use of skip gram and continuous bag of words model for AraVec. The results for accuracy in terms of sentiment classification on unseen data for the lexicons AT-BLL & AT-MPQA-SL are.

Table 1: Skipgram method of word embedding (AraVec)

	SGD	SVM
AT-BLL	79.5%	80.6%
AT-MPQA-SL	77.6%	79.2%

Table 2: Continuous bag of words method of word embedding (AraVec)

	SGD	SVM
AT-BLL	75.1%	76.3%
AT-MPQA-SL	74.9%	77.2%

As it is evident for Table 1 and Table 2 that the Skip gram method reports a slightly better performance when compared to the continuous bag of words method. Both the Skip gram and continuous bag of words models were trained on www data. Also, the SVM classifier gives a notch better performance in comparison to the stochastic gradient descent classifier.

V. CONCLUSION

In this paper, we presented a novel approach of extending the sentiment classifier to data previously unknown with satisfactory performance. To this end we used word embedding to transform the input words to vector space where similar words are mapped

close together. The sentiment classifier performance is mainly dependent on the quality of the word embedding used. An improvement in the embedding model can significantly improve sentiment classification performance.

VI. ACKNOWLEDGEMENT

This work is done under the grant received by Deanship of research at Islamic University of Madinah(IUM), Saudi Arabia. We would also like to thank and acknowledge the work done by AbdulAziz Almuzaini (Lecturer, Islamic University of Madinah) and Dr. Wajahat Ali Khan (Assistant Professor, Kyung Hee University, South Korea) and their close support in development and implementation of this project.

REFERENCES

[1] J. Owens, The Oxford handbook of Arabic linguistics, Oxford University Press, 2013.
[2] M. El-Masri, N. Altrabsheh, H. Mansour, Successes and challenges of arabic sentiment analysis research: a literature review, Social Network Analysis and Mining 7 (1) (2017) 54.
[3] N. Boudad, R. Faizi, R. O. H. Thami, R. Chiheb, Sentiment analysis in arabic: A review of the literature, Ain Shams Engineering Journal.
[4] S. R. El-Beltagy, A. Ali, Open issues in the sentiment analysis of arabic social media: A case study, in: Innovations in information technology (iit), 2013 9th international conference on, IEEE, 2013, pp. 215-220.
[5] A. Hamdi, K. Shaban, A. Zainal, A review on challenging issues in arabic sentiment analysis, Journal of Computer Science 12 (9) (2016) 471-481. doi:10.3844/jcssp.2016.471.481.
[6] N. A. Abdulla, N. A. Ahmed, M. A. Shehab, M. Al-Ayyoub, Arabic sen-timent analysis: Lexicon-based and corpus-based, in: Applied Electrical Engineering and Computing Technologies (AEECT), 2013 IEEE Jordan Conference on, IEEE, 2013, pp. 1-6.
[7] A. B. Soliman, K. Eissa, S. R. El-Beltagy, Aravec: A set of arabic word embedding models for use in arabic nlp, Procedia Computer Science 117 (2017) 256-265.
[8] M. Korayem, D. Crandall, M. Abdul-Mageed, Subjectivity and senti-ment analysis of arabic: A survey, in: International Conference on Advanced Machine Learning Technologies and Applications, Springer, 2012, pp. 128-139.
[9] M. Taboada, J. Brooke, M. To loski, K. Voll, M. Stede, Lexicon-based methods for sentiment analysis, Computational linguistics 37 (2) (2011) 267-307.
[10] N. El-Makky, K. Nagi, A. El-Ebshihy, E. Apady, O. Hafez, S. Mostafa, S. Ibrahim, Sentiment analysis of colloquial arabic tweets, in: ASE Big-Data/SocialInformatics/PASSAT/BioMedCom 2014 Conference, Har-vard University, 2014, pp. 1-9.
[11] M. Salameh, S. Mohammad, S. Kiritchenko, Sentiment after translation: A case-study on arabic social media posts, in: Proceedings of the 2015 conference of the North American chapter of the association for compu-tational linguistics: Human language technologies, 2015, pp. 767-777.
[12] S. M. Mohammad, M. Salameh, S. Kiritchenko, How translation alters sentiment, Journal of Arti cial Intelligence Research 55 (2016) 95-130.
[13] M. Hu, B. Liu, Mining and summarizing customer reviews, in: Proceed-ings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2004, pp. 168-177.
[14] T. Wilson, J. Wiebe, P. Ho mann, Recognizing contextual polarity in phrase-level sentiment analysis, in: Proceedings of the conference on human language technology and empirical methods in natural language processing, Association for Computational Linguistics, 2005, pp. 347-354.
[15] V. Vryniotis, The importance of neutral class in sentiment analysis, Machine Learning and Statistics.
[16] M. Koppel, J. Schler, The importance of neutral examples for learning sentiment, Computational Intelligence 22 (2) (2006) 100-109.
[17] D. D. Lewis, Naive (bayes) at forty: The independence assumption in information retrieval, in: European conference on machine learning, Springer, 1998, pp. 4-15.
[18] L. Bottou, Large-scale machine learning with stochastic gradient de-scent, in: Proceedings of COMPSTAT'2010, Springer, 2010, pp. 177-186.
[19] A. McCallum, K. Nigam, et al., A comparison of event models for naive bayes text classi cation, in: AAAI-98 workshop on learning for text categorization, Vol. 752, Citeseer, 1998, pp. 41-48.
[20] T. Joachims, Text categorization with support vector machines: Learning with many relevant features, in: European conference on machine learning, Springer, 1998, pp. 137-142.
[21] X. Rong, word2vec parameter learning explained, arXiv preprint arXiv:1411.2738.
[22] T. Mikolov, K. Chen, G. Corrado, J. Dean, E cient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781