A Comparative Evaluation of Word Embeddings Techniques for Twitter Sentiment Analysis

Ibrahim KAIBI
AICSM laboratory,
Mohammed First University
Nador, Morocco
i.kaibi@ump.ac.ma

El Habib NFAOUI
LIIAN laboratory,
Sidi Mohamed Ben Abdellah University
Fez, Morocco
elhabib.nfaoui@usmba.ac.ma

Hassan SATORI
AICSM laboratory,
LIIAN laboratory,
Sidi Mohamed Ben Abdellah University
Fez, Morocco
hsatori@yahoo.com

***Abstract*—The writer's opinion is important information, which is becoming increasingly desirable with the increasing volume of user-generated content on the Web. That's why made sentiment analysis an important tool for extracting this information, which can usually be positive, negative or neutral. To target sentiment classification problem, the standard approach used is the binary classification, considering the sentiment (or the polarity), positive or negative. The classification result depends on the text representation and then the extracted features used to train the classifier. Word embeddings techniques have emerged as a prospect for generating word representation for different text mining tasks, especially sentiment analysis. In this paper, we focus on the comparison of three commonly used word embeddings techniques (Word2vec, Fasttext and Glove) on Twitter datasets for Sentiment Analysis, employing six popular machine learning algorithms, namely, GaussianNB, LinearSVC, NuSVC, LogisticRegression, SGD and RandomForest. We find that Fasttext representation used with NuSVC, which is a type of SVM classifier, outperforms the other combinations in accuracy.**

Keywords— Sentiment Analysis (SA); Word embedding; Arabic text

I. INTRODUCTION:

The use of the word embedding achieves a very important advancement in several domains related to natural language processing (NLP) such as text mining, sentiment analysis, topic segmentation and recommendation [1][2][3], as well as computer vision [4][5]. Several types of models have been proposed to generate vector representations of words [6], called also distributed word representation, word embedding idea is inspired from distributional hypothesis [7], which is that words used in the same contexts have a high proportion of similar meanings. In general, the idea is to project a set of words in a continuous space to obtain a rich feature vector for each word , most of developed models are based on artificial neural networks to build these vectors [6][8], but there is also count based models such as LSA [9] and GloVe [10].

Sentiment Analysis (SA) has important position with fields of research in which word embedding is applied, especially in features extraction task [11][12], the quality of vectors in terms of semantics and syntax make it very common in this kind of task [13]. Also known as Opinion Mining, sentiment analysis refers to the use of many techniques and approaches of several areas, such as machine learning, natural language processing and data mining for the extraction and identification of subjective information from text. [14].

One of the most important modern orientations in SA is the use of Deep learning techniques associated with word embeddings [14], in this paper, we experimentally compare three commonly used word embeddings models (Word2vec, Fasttext and Glove) on Arabic Twitter datasets for SA feature extraction task, employing six well-known machine learning algorithms, namely, GaussianNB, LinearSVC, NuSVC, LogisticRegression, SGD and RandomForest. The main goal of our work is to determine the most efficient combination of word embeddings models and classification algorithms, for SA feature extraction task and sentiment classification, respectively. In this work we focus on the binary classification of sentiment (positive or negative) in Arabic text.

We recall that Word2vec is the most popular word embedding model created by Mikolov et al [6], based on two neural network model architectures (Skip-gram and CBOW) for learning distributed representations of words. Skip-gram takes the word as input and predict surrounding words as output, conversely CBOW takes context words and predicts the word. FastText is an extension of the Skip-gram model [15], where word representations are increased using ngrams of characters. Each ngram character has its own associated vector, and the vector representation of the word is created by summing ngrams vectors appearing in this word [15]. GloVe, for Global Vectors, is a weighted least squares model that trains on global word-word co-occurrence counts, the model is based on a matrix co-occurrence generation and factorization, and it produces a word vector space with meaningful substructure.

the rest of the sections are organized as follows. We discuss some related work in Section 2; Methodology and comparison process described in Section 3; In section 4 we discuss experiment results comparison, and in section 5 we conclude the paper.

II. RELATED WORKS

Currently, word embedding is commonly used to address some tasks of sentiment analysis process. In this section We recall some of the previous works that use word embedding in Sentiment Analysis. [2] the authors compared three word embedding models, Wor2Vec, GloVe and LSA. The purpose of their work was to determine which model is the most effective for learning representations of words vectors ensuring the semantics of words for the Arabic and English languages, their work is focused on topic segmentation, and for such kind of tasks, they showed that GloVe and Word2Vec are more efficient than LSA for both languages, and Wod2Vec

978-1-5386-7850-3/19/$31.00 ©2019 IEEE

presents the best representation of word vectors with small dimensional semantic.

In [16] they built Wor2ec model for features extraction of Arabic tweets, so they use text collections that is available publicly to build a corpus, with both Modern Standard Arabic content and a dialectal one. They also included the full text of the Qur'an. For the classification phase they chose a set of six classifiers (Linear SVC, Random Forest, Gaussian Naive Bayes, NuSVC, Logistic Regression, SGDClassifie). They compared the work with those of [17][18] in terms of subjectivity classification and their word embedding method achieved a significant performance.

In 2017 Al-Smadi et al. [11], studying Arabic Hotels' reviews using aspect-based sentiment analysis. To this end, they implemented two approaches, support vector machine (SVM) and deep recurrent neural network (RNN), and the two algorithms are trained with five features (syntactic, word, lexical, semantic and morphological). The results of the evaluation show that the SVM performs better than the deep RNN in three tasks, aspect sentiment polarity identification, aspect opinion target expression extraction and aspect category identification. The word embedding vectors used to training RNN were extracted using word2vec model.

III. METHODOLOGY AND COMPARISON PROCESS

Figure 1 shows the main steps used for making the experimental comparison. After text preprocessing step, we build the vocabulary and we train the three models Word2vec, Fasttext and GloVe for representing words by real numbers vectors. Then, we compute the tweet vector representation using two methods: simple average and TF-IDF weighted average of word vectors composing this tweet. These vectors carry the important syntactic and semantic information to enhance the training of machine learning algorithms used in sentiment classification step. We mention that for the building of the vocabularies, we used the largest freely accessible Arab corpus OSAC [19], which is collected from several websites.

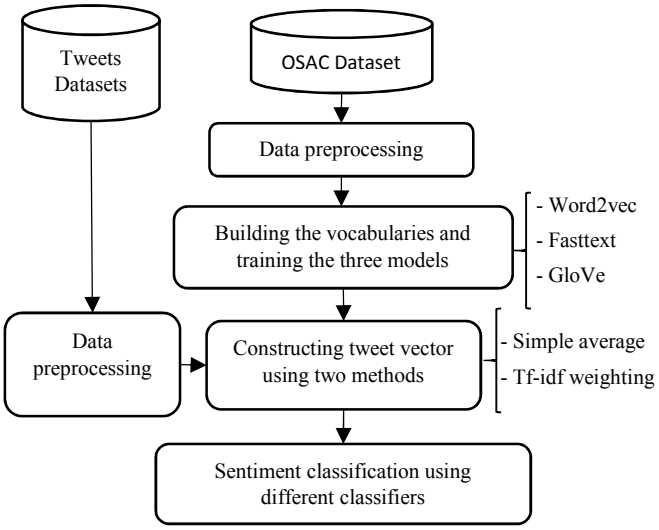

Fig. 1: Steps performed for comparison

A. Data collection

The first step of our work consists of collecting data from several sources. For this end, we have chosen to build our dataset by aggregating four datasets already published for research activities. Each dataset is composed of a well-defined number of tweets belonging to different areas. The collected data is used to train classifiers, for this task, one of the three labels ('1', '0', '- 1') is attributed to the tweets, based on the source data, the label 1 for the tweets that contains the positive opinion, 0 for negative tweets and -1 for the rest, Table 1 shows more details about the datasets:

TABLE I. DATASETS USED TO GENERATE WORD EMBEDDING MODELS AND SENTIMENT CLASSIFICATION

	Dataset	Description
Data for classification task	Arabic Gold Standard Twitter Data for Sentiment Analysis [18]	Tweets annotated manually using three labels (negative, neutral and positive), the dataset content 6514 Arabic entry.
	ASTD: Arabic Sentiment Tweets Dataset [20]	A set of 10,000 Arabic tweets labelled as : subjective mixed, subjective positive ,subjective negative and objective.
	Twitter Data set for Arabic Sentiment Analysis [21]	1000 negative and 1000 positive tweets collected by using a tweet crawler
	Twitter corpus of movies in Arabic Language [22]	1800 tweets about arabic movies which are "الفيل الأزرق", "صنع فى مصر", "الحرب العالمية التالتة" and "جوازة ميري" .
	Total number of tweets : 20320	
Data for building word embeddings models	Open Source Arabic Corpora (OSAC) [19]	Total number of documents: 22432 Total number of words after cleaning process: 16.67 million 11 categories of document: (Religious and Fatwas, Education & Family, History, Stories, Astronomy, Economics, Sports, Entertainments, Heath, Cooking Recipes).

B. Preprocessing

Due to the non-formal nature of tweets text, several text preprocessing operations of Arabic tweets have been applied, mainly, for word embedding models to generate a stable and non-divergent vocabulary. The same word with same meaning can be written in different ways, which generates the creation of several vectors according to the syntaxes used. Otherwise an Arabic tweet may contain non-Arabic words, and Arabic communications but written with Latin characters "ARABIZI".

For embedding models to include only words written in Arabic letters, whether it is standard or dialectal Arabic, a non-Arabic character removal operation is executed. Then, we apply normalization scripts such as "tatweel" and "tashkeel" [1] [23], Finally, each tweet is transformed into a list of words using tokenization, an example of the preprocessing steps is presented in Table 2.

[1] https://pypi.python.org/pypi/pyarabic

TABLE II. PREPROCESSING TASKS

Step	Result
Initial tweet	"هـــــذا خبررررر ســــــارٌ جدًاااا #nice"
Non-Arabic letters elimination	"هـــــذا خبررررر ســــــارٌ جدًاااا"
Normalization	"هذا خبر سار جدا"
Tokenization	"هذا", "خبر", "سار", "جدا"

C. *Features extraction*

The features extraction is one of the primordial phases in the process of sentiment analysis, we focused on comparing of the most widespread word embedding models (Word2vec, Fasttext, Glove) in this task. The same parameters are used for the three models, after several test combinations of parameters, we opted for dimensionality of the feature vectors the value 300, and for the maximum distance between the current and predicted word within a sentence the value 5, the words with total frequency lower than 2 are ignored, and we set the number of threads to train the model to the value 4.

These four common parameters (size=300, window=5, min_count=2 and threads=4) are used to build vocabulary and training each of models (word2vec, GloV and fasttext). The total number of documents used in this task is 22432, the sum of all words exceeds 16.6 million words. The results of this phase are a vocabulary composed of the words and vectors that represent these words in space of dimension 300, a count of words in vocabulary generated by the three methods is the same and equal to 251247 words. In this work, we use a genism [24] implementation of word2vec and fasttext, but for GloVe we use the official one [2], because there is no gensim library for this latter.

D. *Tweets representation*

We use two methods to represent tweet vectors. The first one is the simple average (mean) of vectors of the words that compose the tweet. In the equations, n represents the number of words that make up the tweet, and it differs from one tweet to another:

$$\mathrm{Vec_{tweet}} = \frac{\sum_{i=1}^{n} \mathrm{Vec_{word_i}}}{n} \tag{1}$$

And the second method is the mean Tfidf-weighting of the vectors, which is obtained by multiplying each vector by the Tfidf-weight of the corresponding word before calculating the average:

$$\mathrm{Vec_{tweet}} = \frac{\sum_{i=1}^{n} \mathrm{Tfidfweight}_{\mathrm{word_i}} \times \mathrm{Vec_{word_i}}}{n} \tag{2}$$

E. *Sentiment classification*

For binary sentiment classification, we use the same six classifiers employed in [16], namely Linear SVC, Random Forest, Gaussian Naive Bayes, NuSVC, Logistic Regression and SGDClassifie. We justify our choice of these classifiers because we aim make comparisons with previous work, as well as the determination of the best combination of word embedding model and classifier.

IV. EXPERIMENT, COMPARISON AND DISCUSSION

Table 3 and Table 4 show precision recall and f1-score of the six classifiers. Each line represents results using different word embeddings techniques in features extraction phase, and difference between the two tables is that Table3 represents the values resulting from the use of average of words vector to generating tweet vector, and Table 4 values is using the mean Tfidf-weighting.

The best result obtained in terms of precision, recall and f1-score is that of the combination of Fasttext model, NuSVC classifier and the mean of the tweet word vectors, that respectively equal to 84.89%, 84.29% and 83.85%.

After executing the binary classification of the tweets represented by vectors generated from the vocabularies of the three models Word2vec, Fasttext and GloVe, using two representation methods (the average and the weighted average) and testing several classifiers, the results displayed in Table 3 and Table 4 show that in terms of accuracy, recall and f1-score, the Fasttext model outperforms Wor2vec and GloVe models with all the classifiers, except in the case of GaussianNB classifier and the mean word method, Word2vec exceeds Fasttext by a very small proportion in the three measures.

In general, the results obtained by the method of representation of tweets by means of word vectors, are more perform than that obtained using tfidf-weight, this is due to the diversity of categories and domains of tweets used, so tf-idf is clearly not fit to represent tweets efficiently [25].

Comparing Word2vec and GloVe, Table 3 shows that Word2vec exceeds GloVe in all classifiers except the case of NuSVC classifiers, in which there is a small advantage of GloVe.

We note that the results of the Tf-idf method show a stability of order in terms of precision for the six classifiers, Fasttext the first one, followed by Word2vec and GloVe in the last place, and with the same representation method the combination of Word2vec and the classifier RandomForest reaches the precision 83.83% which is an important result.

TABLE III. CLASSIFIERS RESULTS FOR EACH WORD EMBEDDING MODEL USING A MEAN FOR THE TWEETS VECTOR REPRESENTATION

		GaussianNB	LinearSVC	NuSVC	LogisticRegression	SGD	RandomForest
Word2vec	Precision	67.94%	79.43%	82.29%	79.62%	74.83%	80.98%
	Recall	68.58%	79.20%	81.75%	79.54%	75.11%	80.31%
	F1-score	67.73%	78.60%	81.17%	79.04%	74.76%	79.57%
Fasttext	Precision	67.60%	81.26%	**84.89%**	82.80%	76.97%	82.89%
	Recall	68.25%	81.19%	**84.29%**	82.74%	77.10%	81.86%
	F1-score	67.46%	80.82%	**83.85%**	82.45%	77.01%	81.12%
Glove	Precision	65.22%	77.12%	82.69%	78.40%	71.61%	79.50%
	Recall	66.04%	77.10%	81.97%	78.21%	71.68%	78.54%
	F1-score	65.16%	76.47%	81.34%	77.54%	71.64%	77.51%

[2] https://nlp.stanford.edu/projects/glove/

TABLE IV. CLASSIFIERS RESULTS FOR EACH WORD EMBEDDING MODEL USING A MEAN TF-IDF WEIGHTING FOR THE TWEETS VECTOR REPRESENTATION

		GaussianNB	LinearSVC	NuSVC	LogisticRegression	SGD	RandomForest
Word2vec	Precision	67.24%	78.99%	83.20%	79.54%	73.22%	81.71%
	Recall	67.81%	78.65%	82.30%	79.20%	73.34%	81.08%
	F1-score	66.05%	77.93%	81.64%	78.53%	73.27%	80.43%
Fasttext	Precision	67.68%	80.08%	83.60%	81.17%	76.32%	**83.83%**
	Recall	67.92%	79.87%	**82.63%**	80.97%	75.77%	82.30%
	F1-score	65.59%	79.32%	**81.97%**	80.51%	75.93%	81.45%
Glove	Precision	62.95%	76.78%	81.30%	76.08%	70.76%	79.23%
	Recall	64.16%	76.66%	80.42%	76.00%	71.02%	77.99%
	F1-score	62.26%	75.90%	79.61%	75.19%	70.85%	76.78%

V. CONCLUSION

We compared the three commonly used word embeddings techniques (Word2vec, Fasttext and Glove) on Twitter datasets for Sentiment Analysis. To this end, we trained the three models on a large Arabic corpus (OSAC) to build words vectors representation for each model, using six popular machine learning algorithms, namely, GaussianNB, LinearSVC, NuSVC, LogisticRegression, SGD and RandomForest. The results show the effectiveness of Fasttext compared to other models, and we find that simple average of words vector performs better than the mean Tfidf-weighting in the case of tweets, as well as the NuSVC classifier which is based on SVM achieving the best results followed by RandomForest classifier. In brief, we emphasize that the use of Fasttext, vectors average and NuSVC for features extraction, tweet representation and sentiment classification, respectively, is the best option found.

REFERENCES

[1] Q. Li, S. Shah, X. Liu, and A. Nourbakhsh, "Data Sets: Word Embeddings Learned from Tweets and General Data," Int. AAAI Conf. Web Soc. Media, pp. 428–436, Aug. 2017.

[2] M. Naili, A. H. Chaibi, and H. H. Ben Ghezala, "Comparative study of word embedding methods in topic segmentation," Procedia Comput. Sci., vol. 112, pp. 340–349, 2017.

[3] Y. Xu, J. Liu, W. Yang, and L. Huang, "Incorporating Latent Meanings of Morphological Compositions to Enhance Word Embeddings," Proc. ACL, pp. 1232–1242, 2018.

[4] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP," Procedia Comput. Sci., vol. 117, pp. 256–265, 2017.

[5] D. Li, H.-Y. Lee, J.-B. Huang, S. Wang, and M.-H. Yang, "Learning Structured Semantic Embeddings for Visual Recognition," arXiv Prepr. arXiv1706.01237, Jun. 2017.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," IEEE Trans. Neural Networks, vol. 14, no. 6, pp. 1–12, Jan. 2013.

[7] Z. S. Harris, "Distributional Structure," WORD, vol. 10, no. 2–3, pp. 146–162, 1954.

[8] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," Trans. Assoc. Comput. Linguist., vol. 5, no. 1, pp. 135–146, 2017.

[9] S. T. Dumais, "Latent semantic analysis," Annu. Rev. Inf. Sci. Technol., vol. 38, no. 1, pp. 188–230, Sep. 2005.

[10] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2015, vol. 28, no. 1–2, pp. 1532–1543.

[11] M. Al-Smadi, O. Qawasmeh, M. Al-Ayyoub, Y. Jararweh, and B. Gupta, "Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews," J. Comput. Sci., vol. 27, pp. 386–393, 2018.

[12] S. Al-Azani and E. S. M. El-Alfy, "Using Word Embedding and Ensemble Learning for Highly Imbalanced Data Sentiment Analysis in Short Arabic Text," in Procedia Computer Science, 2017, vol. 109, pp. 359–366.

[13] L. White, R. Togneri, W. Liu, and M. Bennamoun, "How Well Sentence Embeddings Capture Meaning," in Proceedings of the 20th Australasian Document Computing Symposium on ZZZ - ADCS '15, 2015, pp. 1–8.

[14] M. Al-Ayyoub, A. A. Khamaiseh, Y. Jararweh, and M. N. Al-Kabi, "A comprehensive survey of arabic sentiment analysis," Information Processing and Management, 2018.

[15] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning Word Vectors for 157 Languages," in Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018), 2018, pp. L18-1550.

[16] A. A. Altowayan and L. Tao, "Word embeddings for Arabic sentiment analysis," Proc. - 2016 IEEE Int. Conf. Big Data, Big Data 2016, pp. 3820–3825, 2016.

[17] C. Banea, R. Mihalcea, and J. Wiebe, "Multilingual Subjectivity: Are More Languages Better?," in Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), 2010, no. August, pp. 28–36.

[18] A. Mourad and K. Darwish, "Subjectivity and Sentiment Analysis of Modern Standard Arabic and Arabic Microblogs," Proc. 4th Work. Comput. Approaches to Subj. Sentim. Soc. Media Anal., no. 3, pp. 55–64, 2013.

[19] M. Saad and W. Ashour, "OSAC: Open Source Arabic Corpora," EEECS'10 6th Int. Symp. Electr. Electron. Eng. Comput. Sci., pp. 118–123, 2010.

[20] M. Nabil, M. Aly, and A. Atiya, "ASTD: Arabic Sentiment Tweets Dataset," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 2515–2519.

[21] N. a Abdulla, N. a Ahmed, M. a Shehab, and M. Al-ayyoub, "Arabic Sentiment Analysis: Lexicon-based and Corpus-based," Jordan Conf. Appl. Electr. Eng. Comput. Technol., vol. 6, no. 12, pp. 1–6, 2013.

[22] W. Medhat, "Twitter corpus of movies in Arabic Language," 2014. [Online]. Available: https://www.researchgate.net/publication/270393703_Twitter_corpus_of_movies_in_Arabic_Language.

[23] T. Zerrouki, "Pyarabic, An Arabic language library for Python," 2010. [Online]. Available: https://pypi.python.org/pypi/pyarabic.

[24] R. Rehurek and P. Sojka, "Software framework for topic modelling with large corpora," in LREC Workshop on New Challenges for NLP Frameworks, 2010, pp. 45–50.

[25] C. De Boom, S. Van Canneyt, T. Demeester, and B. Dhoedt, "Representation learning for very short texts using weighted word embedding aggregation," Pattern Recognit. Lett., vol. 80, pp. 150–156, 2016.