Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 198 (2022) 269–275

Procedia
Computer Science

www.elsevier.com/locate/procedia

International Workshop on Computational Intelligence and Cybersecurity in Emergent Networks (CICEN 2021)
November 1-4, 2021, Leuven, Belgium

Comparative study of Arabic text classification using feature vectorization methods

Tarik Sabri [a*], Omar El Beggar [a], Mohamed Kissi [a]

[a] *Laboratory LIM, Department of Computer Science, Faculty of Sciences and Technology, University Hassan II Casablanca, B.P. 146, Mohammedia, 20650, Morocco*

Abstract

Arabic Text Classification (ATC) also known Arabic text categorization is the task of assigning categories to Arabic documents based on their contents. It is mostly used for sentiment analysis, detecting trends in customer feedback, spam detection and topic labeling. This paper presents an empirical study of five classification models using two Arabic datasets cnn_arabic and osac_uft8. These algorithms are Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), K-Nearest Neighbor (KNN) and Logistic Regression (LR). Three feature vectorization methods were applied to convert text into numeric vectors word count, Terms Frequency-Inverse Document Frequency (TF-IDF) and word embedding using word2vec. For the applied feature vectorization techniques, the experiment shows that the classifiers SVM and LR score the highest performance followed by RF, KNN and DT. Besides, the experiment shows that feature vectorization methods and dataset size have high impact on the performance of the algorithms RF, KNN and DT, while SVM and LR maintain stable outcomes.

Keywords: Arabic text classification; feature vectorization; word count; TF-IDF; word embedding; word2vec;

[*] Corresponding author. Tel.: +212 6 64 85 26 06
E-mail address: sabritarik@gmail.com

10.1016/j.procs.2021.12.239

1. Introduction

Text categorization is the assignment of documents to one or more predefined labels depending on their content. To do this, the text must be converted into numerical vectors using feature vectorization methods such as word count, TF-IDF and word2vec [1].

Word count is a simple mode of extracting features from documents, it is one of the most used ways to transform text into a series of numbers [2]. TF-IDF measure is used to determine in what proportions certain words in a text document can be evaluated in relation to the rest of the text, it is obtained by multiplying two terms namely TF and IDF [3]. These methods generate vectors of very large dimensionality. However, there are other ways to convert text into numbers [4]. Generally, word embedding is a representation of a term as a vector of numeric values [5]. The word2vec is a method for creating word embeddings; it takes as input words from a large dataset and learns to assign their vector representation. These vectors are chosen based on the similarity cosine method, which indicates the semantic similarity between the input words or features [6].

In this paper, we will compare five classifiers: SVM, DT, RF, KNN and LR using three feature vectorization methods: word count, TF-IDF and word2vec, based on two Arabic datasets: cnn_arabic includes 5070 documents and osac_uft8 with 22.429 documents. The experiment results revealed higher performance of LR model followed by SVM for both datasets.

The rest of this paper is organized as follows. Section 2 presents the related works about feature vectorization methods. Section 3 outlines the experiment results and the discussion. Section 4 is devoted to the conclusion and forthcoming works.

2. Related works

Most of the classification of texts researches is designed for English and other languages such as German, Italian and Spanish. However, works on the classification of Arabic language remain limited. Among those works, several recent researches have been proposed.

Aliwy and Ameer in [7] presented and compared five algorithms which are DT, SVM, KNN, NB and hidden Markov model (HMM). They surveyed the improvement which was done for each many researches. They also described each algorithm separately and studied the modifications made to the same algorithm. This study showed that modification of learner and feature selection can help for increasing the accuracy of the algorithm.

In [8], the authors presented a improved method for Arabic text categorization that uses the Chi-square feature selection (ImpCHI) to enhance the classification performance. The experiments revealed that the combination of ImpCHI and SVM outperforms the other approaches in terms of accuracy, F-measure and recall. The best F-measure value obtained exceeds 90% when the number of features equal to 900.

Elnagar et al. in [9] proposed two new large corpuses for Arabic text categorization collected from news portals (SANAD and NADiA) [9]. The results showed solid performance of all models on SANAD corpus with a minimum accuracy of 91.18%, as for NADiA, attention-GRU achieved the highest overall accuracy of 88.68%.

Overall, compared to our approach, the previous works did not interested in studying the impact of either feature vectorization methods or dataset size on the performance of the ATC models.

3. Experiment results and discussion

This section presents an empirical study of three feature vectorization methods: TF-IDF, word count and word2vec. We used two benchmark corpuses to study, compare and evaluate these methods. Once feature vectorizations are calculated, five machine learning algorithms: SVM, DT, RF, KNN and LR will be used. The results are established on the basis of the three statistical formulas such as precision, recall and F-measure.

3.1. Datasets

The dataset "cnn_arabic" is a collection of Arabic texts, it consists of 5.070 documents. The dataset contains six labels such as entertainment, business, scitech, sport, middle east and world. The Arabic dataset "osac_uft8" contains

22.429 documents and divided into ten categories: economic, history, education_family_woman, religion, sport, health, astronomy, law, stories and food. Both datasets are available at [10] and free for researchers. Table 1 shows the number of documents in each category:

Table 1. Documents distribution among categories

Dateset “cnn_arabic”	
Category	Documents
Business	836
Entertainment	474
Middle east	**1.462**
Scitech	526
Sport	762
World	1.010
Dataset total number	5.070

Dataset “osac_uft8”	
Category	Documents
Economic	3.102
History	3.233
Education Family Woman	**3.608**
Religion	3.171
Sport	2.419
Health	2.296
Astronomy	557
Law	944
Stories	726
Food	2.373
Dataset total number	22.429

3.2. Pre-processing

Text pre-processing is an important phase for natural language processing. It is required to transform the text into an understandable format, so that machine learning algorithms can be applied to it [11]. For this reason, we performed the following tasks as a text pre-processing of the studied datasets:

- Remove non-arabic words.
- Remove non-arabic letters: numbers, punctuation and symbols (. , ; / | \ * % $ = + …).
- Remove all Arabic stop words such as ('من','at'), ('على','on').
- Remove all diacritics of the Arabic words (◌ً◌ٌ◌ِ◌ٍ◌َ◌ْ◌ُ).
- Normalize certain letters to one form. For example, the normalization of 'آ' (alif al mad), 'ئ' (hemza on yaa), "إ أ" (alif with hemza on top or bottom), 'ؤ' (hemza on wew) to 'ا' (alif).

3.3. Feature vectorization

Feature vectorization is a mechanism for transforming text into numerical feature vectors that can be used for machine learning. In this work we used three feature vectorization methods as previously described in [12].

- word count:

word count is a simple way that represents the occurrence of words within a document. This process is often referred to as vectorization. Table 2 shows an example of word count method.

Table 2. An example of word count.

Documents	w1	w2	w3	w4	w5	w6
d1	0	0	3	2	1	0
d2	3	0	0	3	2	1
d3	0	3	2	1	1	1
d4	1	1	2	0	0	1

Table 2 represents a word count matrix. The dataset consists of four documents d1, d2, d3 and d4. For instance, the value 3 in (d2,w4) indicates that word4 appears thrice in d2. Meanwhile, the value 0 in (d4,w5) indicates that word5 does not appear in document d4.

- TF-IDF:

The *tfidf* vectorizer denotes term frequency tf and inverse document frequency idf. The mathematical formula for this measure is defined as [13]:

$$tfidf(t,d,D) = tf(t,d) \times idf(t,D) \tag{1}$$

t : terms, d : document, D : collection of documents.

In fact, the Term Frequency tf measures the number of times each word appeared in each document, tf is defined as:

$$tf(t,D_i) = \frac{count(t)}{|D_i|} \tag{2}$$

Where $count(t)$ represents the number of occurrences of the term t, and $|D_i|$ is the number of all words in the document D_i.

On the other hand, the Inverse Document Frequency idf is used to determine whether a term is rare or common across a dataset. Common words have less value as opposed to ones that occur rarely. The idf is defined as follows:

$$idf(t,D) = \log\frac{|D|}{1+|\{d \in D : t \in d\}|} \tag{3}$$

$|D|$: size of the document space, $|\{d \in D : t \in d\}|$: represents the number of times the term has appeared in the document d.

- word2vec:

Word2vec is a set of algorithms to generate word embeddings as numerical vectors. The general idea is to use the context of adjacent words and identify similar words based on their representation in the vector space. Two models to obtain word embeddings: The CBOW and skip-gram model [14]. Indeed, the CBOW takes the input context words to predict a target word, while the skip-gram model uses a target word to predict the context. The models are described in detail in [15]. Unlike word count and TF-IDF, word2vec creates one vector per word. It is very useful to explore documents and identify the content.

3.4. Performance metrics

Precision, F-measure and recall are three crucial metrics, they represent the most useful and widely used methods for evaluating text classifiers [1].

- Precision P measures the number of positive class:

$$P = \frac{T_p}{T_p + F_p} \tag{4}$$

T_p : the number of true positives, F_p : the number of false positives.

- Recall R measures the performance of a model to predict all the positive instances:

$$R = \frac{T_p}{T_p + F_n} \tag{5}$$

F_n : the number of false negatives.

- F-measure F_1 is calculated using the harmonic mean of precision and recall as follows:

$$F_1 = 2 \times \frac{P \times R}{P + R} \tag{6}$$

3.5. Discussion

Both datasets are tested using a random split method, where the two-thirds of data are used for training and one-third for testing. Tables 3 and 4 show the number of documents for each category in both datasets. We tested all methods in this paper on a 64-bit PC i7 6th Generation, RAM 16 Go, Hard disk SSD 512 Go, touring on Windows10.

Table 3. Number of documents for each category on dataset "cnn_arabic".

Categories	Training documents	Testing documents	Total number
business	561	275	836
entertainment	337	137	474
middle_east	**1.033**	**429**	**1.462**
scitech	388	138	526
sport	528	234	762
world	702	308	1.010
Total	3.549	1.521	5.070

Table 4. Number of documents for each category on dataset "osac_uft8".

Categories	Training documents	Testing documents	Total number
Economic	2.198	904	3.102
History	2.244	989	3.233
Education, Family and Woman	**2.480**	**1.128**	**3.608**
Religion	2.218	953	3.171
Sport	1.708	711	2.419
Health	1.598	698	2.296
Astronomy	401	156	557
Law	664	280	944
Stories	500	226	726
Food	1.689	684	2.373
Total	15.700	6.729	22.429

Table 5 shows the precision, recall, and F-measure for each model using the dataset "cnn_arabic". The LR classifier produced the best result of 93.65% noted in recall using TF-IDF followed by SVM using word count with 93.19% scored in recall. However, the KNN classifier produced the worst result of 73.11%. Furthermore, DT classifier produced results between 74.77% and 80.24%, while the RF classifier performed below the average with scores range from 81.24% and 91.53%.

Table 5. Performance evaluation of ML classifiers using TF-IDF, word count and word2vec on "cnn_arabic".

	Feature vectorization methods								
classifiers	TF-IDF			word count			word2vec		
	P	R	F1	P	R	F1	P	R	F1
SVM	92.98	93.00	92.97	92.74	93.19	92.93	**91.68**	91.42	**91.49**
DT	74.77	75.09	74.89	75.64	76.20	75.87	80.24	79.91	80.02
RF	81.24	89.05	82.88	82.19	90.09	83.98	91.31	91.53	90.94
KNN	73.89	83.03	73.42	73.34	82.43	73.11	90.86	91.33	91.08
LR	**93.20**	**93.65**	**93.41**	**92.79**	**93.42**	**93.08**	91.31	**91.74**	**91.49**
Average	83.22	86.76	83.51	83.34	87.07	83.79	89.08	89.19	89.00

The results for the dataset "osac_uft8" showed a very significant increase compared to the first dataset, the LR model still scored the best performance compared to SVM with a TF-IDF of 98.99% noted in recall, we also notice a significant improvement for the DT, RF and KNN models. Table 6 illustrates the different results of the second dataset.

Table 6. Performance evaluation of ML classifiers using TF-IDF, word count and word2vec on "osac_uft8".

	Feature vectorization methods								
classifiers	TF-IDF			word count			word2vec		
	P	R	F1	P	R	F1	P	R	F1
SVM	98.24	98.35	98.29	98.29	98.19	98.24	**98.11**	98.08	98.09
DT	97.50	97.29	97.39	97.49	97.25	97.36	89.89	90.55	90.20
RF	97.19	98.03	97.54	97.48	**98.23**	97.78	95.94	97.43	96.62
KNN	87.15	94.05	89.40	87.51	94.05	89.80	96.18	96.30	96.23
LR	**98.60**	**98.99**	**98.79**	**98.78**	98.05	**98.91**	98.07	**98.12**	**98.10**
Average	95.74	97.34	96.28	95.91	97.15	96.42	95.64	96.10	95.85

4. Conclusion and future work

In this work, we compared the performances of various classification algorithms using two Arabic datasets "cnn_arabic" with 5.070 documents and "osac_uft8" with 22.429 documents. The feature vectorization methods used are: TF-IDF, word count and word2vec. We preprocessed the datasets by normalizing certain Arabic letters and removing stop words. The results of these experiments show that SVM and LR classifiers have the best performance followed by RF, KNN and DT. We also observed that feature vectorization methods and dataset size have significant impacts on the performances of the models.

This paper provides many experiments and comparisons of five models using two Arabic datasets. The idea can be extended to combine feature vectorization methods and use other Arabic datasets to test the performance of each classifier.

References

[1] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes and D. Brown, "Text classification algorithms: A survey," *Information,* vol. 10, p. 150, 2019.

[2] Y. HaCohen-Kerner, D. Miller and Y. Yigal, "The influence of preprocessing on text classification using a bag-of-words representation," *PloS one,* vol. 15, p. e0232525, 2020.

[3] Z. Zhu, J. Liang, D. Li, H. Yu and G. Liu, "Hot topic detection based on a refined TF-IDF algorithm," *IEEE access,* vol. 7, p. 26996–27007, 2019.

[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 Computer Science,* vol. 117, p. 256–265, 2017.

[5] F. Enríquez, J. A. Troyano and T. López-Solaz, "An approach to the use of word embeddings in an opinion classification task," *Expert Systems with Applications,* vol. 66, p. 1–6, 2016.

[6] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781,* 2013.

[7] A. H. Aliwy and E. A. Ameer, "Comparative study of five text classification algorithms with their improvements," *International Journal of Applied Engineering Research,* vol. 12, p. 4309–4319, 2017.

[8] S. Bahassine, A. Madani, M. Al-Sarem and M. Kissi, "Feature selection using an improved Chi-square for Arabic text classification," *Journal of King Saud University-Computer and Information Sciences,* vol. 32, p. 225–231, 2020.

[9] A. Elnagar, R. Al-Debsi and O. Einea, "Arabic text classification using deep learning models," *Information Processing & Management,* vol. 57, p. 102121, 2020.

[10] M. K. Saad and W. Ashour, "OSAC: Open Source Arabic Corpora".

[11] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Information processing & management,* vol. 50, p. 104–112, 2014.

[12] J. Gao, K. Liu, B. Wang, D. Wang and X. Zhang, "Improving deep forest by ensemble pruning based on feature vectorization and quantum walks," *Soft Computing,* vol. 25, p. 2057–2068, 2021.

[13] C.-z. Liu, Y.-x. Sheng, Z.-q. Wei and Y.-Q. Yang, "Research of text classification based on improved TF-IDF algorithm," in *2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE)*, 2018.

[14] H. Liu, "Sentiment analysis of citations using word2vec," *arXiv preprint arXiv:1704.00177,* 2017.

[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013.