Journal of King Saud University – Computer and Information Sciences 29 (2017) 189–195

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Toward an enhanced Arabic text classification using cosine similarity and Latent Semantic Indexing

Fawaz S. Al-Anzi *, Dia AbuZeina

Department of Computer Engineering, Kuwait University, Kuwait

ARTICLE INFO

Article history:
Received 4 November 2015
Revised 28 March 2016
Accepted 2 April 2016
Available online 8 April 2016

Keywords:
Arabic text
Classification
Supervised learning
Cosine similarity
Latent Semantic Indexing

ABSTRACT

Cosine similarity is one of the most popular distance measures in text classification problems. In this paper, we used this important measure to investigate the performance of Arabic language text classification. For textual features, vector space model (VSM) is generally used as a model to represent textual information as numerical vectors. However, Latent Semantic Indexing (LSI) is a better textual representation technique as it maintains semantic information between the words. Hence, we used the singular value decomposition (SVD) method to extract textual features based on LSI. In our experiments, we conducted comparison between some of the well-known classification methods such as Naïve Bayes, *k*-Nearest Neighbors, Neural Network, Random Forest, Support Vector Machine, and classification tree. We used a corpus that contains 4,000 documents of ten topics (400 document for each topic). The corpus contains 2,127,197 words with about 139,168 unique words. The testing set contains 400 documents, 40 documents for each topics. As a weighing scheme, we used Term Frequency.Inverse Document Frequency (TF.IDF). This study reveals that the classification methods that use LSI features significantly outperform the TF.IDF-based methods. It also reveals that *k*-Nearest Neighbors (based on cosine measure) and support vector machine are the best performing classifiers.

1. Introduction

Recently, text classification (TC) for Arabic language has been widely investigated. Manning and Schütze (1999) defined text classification as the task of classifying texts into one of a pre-specified set of classes based on their contents. According to Sebastiani (2002), text classification is the activity of labeling natural language texts with thematic categories from a predefined set. With big data environment, researchers have been hard at work to address the text classification problem in this huge information era. With massive growth of text search transactions, effective algorithms are needed to satisfy efficient retrieval time and relevance constraints. In today's market, achieving user satisfaction within this astronomical growth of online data is becoming very appealing to business investment. Search engines, e.g., Google and other high traffic query processing portals, are expected to meet and satisfy today's user demands.

Supervised machine learning (ML) approaches are widely used for text classification. The most popular machine learning algorithms include Naïve Bayes (NB), *k*-Nearest Neighbor (*k*-NN), Support Vector Machines (SVM), Neural Networks (NN), Classification Trees (CT), Logistic Regression (LR), Random Forest (RF), and Maximum Entropy (ME). In addition, similarity or distance measures are used for text classification as well as the bases for some classifiers. For example, *k*-NN algorithm uses a similarity function such as Euclidean distance or cosine similarity to find neighbors, Torunoğlu et al. (2011).

In text classification problems, large feature sets are a challenge that should be handled for better performance. Therefore, utilizing feature reduction techniques are important for efficient representation of textual features. Harrag and Al-Qawasmah (2010) presented a number of dimensionality reduction techniques such as root-based stemming, light stemming, and singular valued decomposition (SVD). In this work, we use the SVD as a feature reduction technique as well as for producing semantic rich features. SVD is a linear algebra method that is used to truncate the term-document matrix that produced by Latent Semantic Indexing (LSI), a well-

* Corresponding author.
E-mail addresses: fawaz.alanzi@ku.edu.kw (F.S. Al-Anzi), abuzeina@ku.edu.kw (D. AbuZeina).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2016.04.001
1319-1578/

known indexing and retrieval method. Even though the vector space model (VSM) is widely used for textual features representation, however, it is a semantic loss while LSI-SVD is characterized by maintaining the semantic information. Rosario (2000) showed that SVD could be used to estimate the structure in word usage across the documents based on LSI that has the underlying structure in a word choice. Kantardzic (2011) indicated that LSI gives better results when using in text classification as it enables better representation of document's semantics.

This paper contains two parts. First, the LSI-SVD techniques were used to generate the textual features of a corpus that contains 4000 documents. The generated features were then used along with the cosine similarity measure to classify the testing set documents. Second, a number of classification methods were employed for a comparison purpose. The classifiers include NN, NB, *k*-NN, SVM, RF, CT, LR, and CN2 (induction rule). In the implementation, Gensim ("Gensim", 2016) and Orange tool ("Orange", 2016) were used. Gensim is a Python library for natural language processing (NLP) while Orange is an open source machine-learning tool for data visualization and analysis.

In next section, a literate review is presented. In Section 3, we present the singular value decomposition followed by the theoretical background of the cosine similarity in Section 4. The experimental setup is presented in Section 5, and the results are discussed in Section 6. Finally, the conclusion and future work are presented in Section 7.

2. Literature review

Cosine similarity measure has been widely used in pattern recognition and text classification. For example, Nguyen and Bai (2011) used cosine similarity measure for face verification. In this work, the focus will be on the cosine measure for linguistic applications. Among such applications, Silber and McCoy (2002) used cosine measure for text summarization. El Gohary et al. (2013) used cosine measure to detect the emotions in the Arabic language text. Takçı and Güngör (2012) indicated that cosine is the commonly used similarity measure in the language identification problem. Sobh et al. (2006) used cosine measure for Arabic language text summarization. Roberts et al. (2005) used cosine similarity for Arabic language concordance. Al-Kharashi and Evens (1994) used the cosine measure for indexing and retrieval processes for Arabic bibliographic data. Lin and Chen (1996) used cosine measure to extract concept descriptors (terms or keywords) from a Chinese-English bibliographic database. Elberrichi and Abidi (2012) indicated cosine similarity dominant measures in information retrieval (IR) and text classification.

For Arabic text classification domain, various feature extraction and classification methods were proposed in the literature as shown in Table 1. In this table, TF.IDF is the shorthand for Term Frequency Inverse Document Frequency, the well-known weighting scheme of text features. TF.IDF is a combination of two parts, (TF: the frequency of the word in the document, and IDF: the inverse of the frequency of the word throughout all documents). ANSI is the shorthand for American National Standards Institute.

Even LSI is a powerful feature representation for words' semantic, the literature provided in Table 1 shows that LSI has very little contribution for Arabic text classification. Therefore, an effort was made to address this deficiency by utilizing semantic information for Arabic text classification. The cosine similarity measure was chosen for classification process. The highlighted cells in Table 1 indicate that only two research works have the same scope as this research (LSI and cosine). However, the first work, i.e. Froud et al. (2013) was conducted for document clustering while our proposed research is for text classification. In addition, we used a larger data set containing 4000 documents while they used 278 documents. We also compared the results using eight well-known classifiers as well as exploring the performance of LSI using a wide range of rank approximation. Regarding the other work, i.e. Harrag and Al-Qawasmah (2010), they used NN for classification while we used cosine similarity measure.

As this research demonstrates a comparative study of the different text classification algorithms, we present a comparison between the supervised machine learning algorithms found in the literature. Table 2 shows that SVM outperforms most of the classification algorithms for Arabic language text classification. The information presented in Table 2 are arranged as the researchers, the classifiers used, the best performance classifier, and the corpus size. However, the information provided in Table 2 is not judgemental as we agree with Sebastiani (2002) that illustrated comparisons are only reliable when they are based on experiments performed by the same author under carefully controlled condi-

Table 1
Summary of Arabic text features and classifiers.

References	Features	Classifier
Syiam et al. (2006), Thabtah et al. (2008), Gharib et al. (2009), Hmeidi et al. (2008), Ababneh et al. (2014), Kanaan et al. (2009), Duwairi (2007), Zrigui et al. (2012), Moh'd Mesleh (2011),	TF.IDF	*k*-NN
Elberrichi and Abidi (2012)	ANSI	
Al-Shalabi and Obeidat (2008)	N-gram	
Syiam et al. (2006), Gharib et al. (2009), Omar et al. (2013), Kanaan et al. (2009), Moh'd Mesleh (2011)	TF.IDF	Rocchio
Jbara (2010), Larkey et al. (2004), Al-Eid et al. (2010), Alghamdi and Selamat (2012), Ezzat et al. (2012), Al-Kabi and Al-Sinjilawi (2007), Erkan and Radev (2004)	TF.IDF	*Cosine*
Froud et al. (2013)	*LSI*	
Gharib et al. (2009), Omar et al. (2013), Hmeidi et al. (2008), Al-Shargabi et al. (2011), Zrigui et al. (2012), Alsaleem (2011), Khorsheed and Al-Thubaity (2013), Hadni et al. (2013), Moh'd Mesleh (2011), Al-Shammari (2010), Harrag et al. (2009), Raheel et al. (2009),	TF.IDF	SVM
Al-Harbi et al. (2008)	Chi-Squared	
Al-Kabi and Al-Sinjilawi (2007), Duwairi (2007)	TF.IDF	Dice distance
Gharib et al. (2009), Omar et al. (2013), Al-Shargabi et al. (2011), Al-Kabi and Al-Sinjilawi (2007), Kanaan et al. (2009), Duwairi (2007), Zrigui et al. (2012), Alsaleem (2011), Khorsheed and Al-Thubaity (2013), Hadni et al. (2013), Moh'd Mesleh (2011), Al-Shammari (2010), Harrag et al. (2009), Raheel et al. (2009),	TF.IDF	NB
Al-Shargabi et al. (2011), Khorsheed and Al-Thubaity (2013), Harrag et al. (2009), Raheel et al. (2009)	TF.IDF	CT
Al-Harbi et al. (2008)	Chi-Squared	
Harrag et al. (2009)	TF.IDF	ME
Harrag and Al-Qawasmah (2010)	*LSI*	*NN*

Table 2
A performance comparison of Arabic text classification.

Researchers	Classifiers	Best classifier	Corpus size (doc., cat.)
Al-Shargabi et al. (2011)	NB, SVM, and CT	SVM	2356, 6
Zrigui et al. (2012)	SVM, NB, and *k*-NN	SVM	1500, 9
Gharib et al. (2009)	*k*-NN, NB, and SVM	SVM	1132, 6
Alsaleem (2011)	NB and SVM	SVM	5121, 7
Khorsheed and Al-Thubaity (2013)	SVM, NB, and CT	SVM	2 corpora: Islamic Poems
Hadni et al. (2013)	NB and SVM	SVM	415, 12
Moh'd Mesleh (2011)	*k*-NN, SVM, NB, Rocchio	SVM	7842, 10
Al-Shammari (2010)	NB and SVM	SVM	2966, 3
Hmeidi et al. (2008)	*k*-NN and SVM	SVM	2066, 2
Raheel et al. (2009)	NB, SVM, and CT	SVM	6825, 7
Harrag et al. (2009)	NB, CT, SVM and ME	CT	2 corpora: 350, 8 280, 14
Al-Harbi et al. (2008)	SVM and CT	CT	Seven corpora
Al-Kabi and Al-Sinjilawi (2007)	Cosine, NB, and Euclidean	NB	80, 12
Kanaan et al. (2009)	*k*-NN, NB, and Rocchio	NB	1445, 9
Duwairi (2007)	NB, *k*-NN, and Distance	NB	1000, 10
Harrag et al. (2009)	NB, CT, SVM and ME	CT	2 corpora: 350, 8 280, 14

tions that is, no learning algorithm is universally best for all problems and datasets.

3. Singular value decomposition

In general, Salton and Buckley (1988) can model text classification features using VSM that was proposed. In the VSM, the vector of the document is represented as "bag of words" in which each word corresponds to one independent dimension. Usually, the elements of the vectors is the weights of the importance of the words in the document. The weight can be represented using a binary value to indicate the presence or absence of the word. Other representations can be employed such as n-gram, keywords, or longer sentences, Zrigui et al. (2012). It is clear that VSM representation has huge feature vectors that should be carefully considered to avoid hardware limitation, software capabilities, and computational time complexity.

In this work, we used LSI and SVD methods that were developed to improve the accuracy and effectiveness of IR techniques. LSI focuses on semantic meaning of words across a series of usage contexts, as opposed to using simple string-matching operations, Kantardzic (2011). LSI has been utilized for many natural language processing applications such as search engines, Carpineto et al. (2009), and other domains such as digital image processing, Andrews and Patterson (1976). The goal of using LSI and SVD decomposition technique is to find the relationships between the terms and documents. That is, LSI generate a *term_by_document* matrix that is mathematically decomposed to identify the semantic correlation between concepts and documents in an unstructured text, of course with no loss (or minimum loss) of information.

SVD is based on a theorem from linear algebra which says that a rectangular *m*-by-*n* matrix *A* can be broken down into the product of three matrices – an orthogonal matrix *U*, a diagonal matrix *S*, and the transpose of an orthogonal matrix *V*. The theorem is usually presented as something like this: $Amn = UmmSmnV^{T}nn$. Fig. 1 demonstrates the reduced rank SVD. The bold **K** in the shaded region of *U*, *S*, and V^{T} represents the values retained in computing rank *k* approximation. There are many free and commercial software that are available for related LSI. We initially used MATLAB for decomposing *term_by_document*. However, according to hardware limitation, we used Genism that is characterized by efficient use of memory.

4. Cosine similarity measure

The objective of this work is to investigate the performance of cosine measure as one of the most popular machine learning methods for Arabic language text classification. More precisely, we evaluated the performance of cosine similarity against NN, NB, *k*-NN, SVM, RF, CT, LR, and CN2 Rules. Theodoridis and Koutroumbas (2008) defines cosine similarity measure as $S_{\text{cosine}}(\mathrm{x},\mathrm{y}) = \frac{\boldsymbol{x}^{T}\boldsymbol{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}$ where $\|\mathrm{x}\| = \sqrt{\sum_{i=1}^{l}\boldsymbol{x}_{i}^{2}}$ and $\|\mathrm{y}\| = \sqrt{\sum_{i=1}^{l}\boldsymbol{y}_{i}^{2}}$ are the lengths of the vectors x and y, respectively. Both $\boldsymbol{x}$ and $\boldsymbol{y}$ are $\boldsymbol{l}$-dimensional vectors. Since cosine measure is easy to interpret and simple to compute for sparse vectors, it is widely used in text mining and information retrieval, Dhillon and Modha (2001). Cosine similarity can also be defined by the angle or cosine of the angle between two vectors. This allows documents with the same composition, but different totals, to be treated identically which makes this the most popular measure for text documents, Strehl et al. (2000).

5. The experiments setup

This section presents two subsections, the data set and the proposed method.

5.1. The data set

We created a corpus that contains 4000 documents belonging to 10 different categories. The corpus contains 2,127,197 words that include more than 139,168 unique words. We got the documents from Alqabas newspaper in Kuwait ("Alqabas", 2016). Table 3 shows the statistics of the corpus used.

The testing set contain 400 documents, 40 documents for each category. Hence, the total documents in the prepared corpus is 4400 documents.

5.2. The proposed method

The proposed method is summarized using the following algorithm:

Step 1: For the entire corpus (4000 documents for training and 400 document for testing), a preprocessing step is performed to

Table 3
The corpus and the category distribution.

#	Category	# Doc.	# Words	# Unique words
1	Health	400	218,214	29,574
2	Economy	400	181,366	29,443
3	Crimes and courts	400	172,145	29,416
4	Education	400	259,127	37,515
5	Technology	400	209,319	36,103
6	Sports	400	168,934	29,568
7	Tourism	400	270,142	40,488
8	Islam and Sharia	400	242,943	45,843
9	Parliament	400	182,503	31,183
10	Political Affairs	400	222,504	37,649
	Total	**4000**	**2,127,197**	**346,782***

* The total number of the unique words in the entire corpus is 139,168.

prepare the text for the classification process. Therefore, cleaning the text has the following three steps:

- The stoplist is declared to remove insignificant words. These words are common and have no discriminative meaning. The stoplist includes words that are found in almost all documents such as the name of the newspaper, the source of the document, the serial number of the documents, etc. an example of stoplist in the corpus: {القبس، رقم‌التسلسل، النص، صورة، الخ}.
- The ignore characters are specified. They include {~, ‘, !, @, #, £, €, $, %, °, ^, &, *, (,), -, _, +, =, », «, {,}, [,], |, \, /, :, ;, 0,1,2,3,4,5,6,7,8,9}. Therefore, any word containing one or more of the listed ignore characters is edited to remove the character/s.
- For all documents in the corpus, “أ” is replaced by “ا”, and “إ” by “ا”.

Step 2: Using Python, the Gensim library is utilized to generate TF.IDF features and LSI features using the following steps:

- Create the dictionary that contain all words.
- The documents are converted to vectors using the information in the dictionary and the document word counts.
- The vectors are weighted using TF.IDF. The number of features in each weighted vector is the same number of words in the documents, after passing all filtering processes (i.e. stoplist).
- The LSI vectors (features) are created using TF.IDF vectors. The number of features in each LSI vector is the *k* that is used for LSI transformation.

Step 3: The performance is measured using the TF.IDF features and LSI features generated in the previous step. The cosine similarity measure is used to perform classification. The rank *k* approximation (a suitable singular value) is selected. Hence, different *k* values should be investigated to find the optimum performance. Bradford (2008) indicated that for real corpora, target dimensionality (*k*) of 200–500 is recommended as a “golden standard”. The classification process in this step is like *k*-NN classifier that find the similarity between the document to be tested and all training documents. The *k*-NN classifier finds the *k*-nearest documents, but the cosine similarity classifier return the label of the nearest document (i.e. $k = 1$). The Gensim facilitates using cosine similarity with both TF.IDF and LSI features.

Step 4: The Orange tool is used to measure the performance using the LSI features. The classification is performed using the following classifiers: {NN, NB, *k*-NN, SVM, RF, CT, LR, and CN2 Rules}.

Step 5: The performance is evaluated using confusion matrix to find the performance metrics such as accuracy, precision, recall, and F1 measure. F1 = 2((precision * recall)/(precision + recall)). Sokolova and Lapalme (2009) has a comprehensive review of these measures.

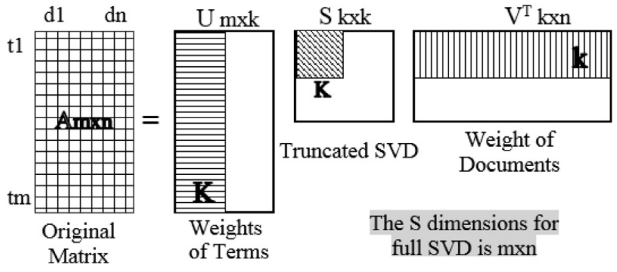

Figure 1. SVD representation of the document-term(s) matrix.

Fig. 2 shows the proposed method in visualization form. The figure shows that TF.IDF features will be compared with LSI features using cosine measure. Then, the LSI features will be compared using all presented classifiers including cosine measure.

6. Experimental results

In this section, the experimental results are presented. Before conducting any experiment, three parameters should be set. The first is the rank *k* approximation (for LSI cases), the small word threshold, and the word frequency threshold. Regarding the small words, it is possible to remove any word that is less than a certain character length. This option gives the choice to remove the single character in the text such as “د محمد”, the character “د” should be removed as it is considered as noise and it will not help in the classification process. In fact, ignoring small words will get rid of many

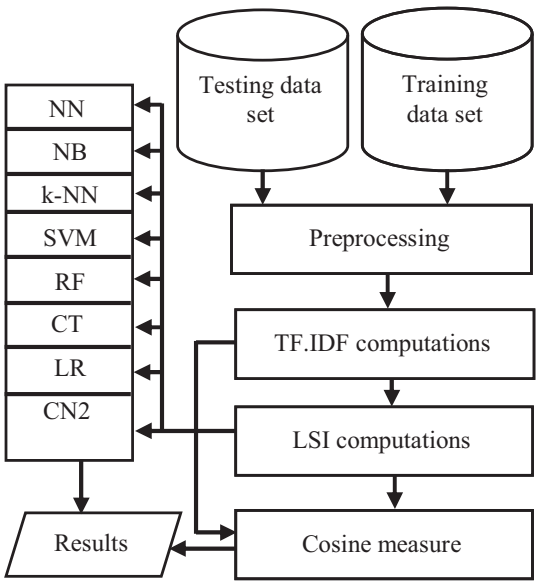

Figure 2. The framework of the proposed method.

Table 4
The performance of the cosine classifier.

Features	Accuracy (%)	Rank-*k*	Word frequency	Small words
TF.IDF	67.25	No need	1	1
LSA	82.50	46	1	1

common words such as {من, في, على, الى} which are translated using Google ("Google", 2016) to: {to, in, on, of}. However, many other small words are considered as keys phrases for some categories such as: {فن, كرة, دم, بنك, نفط} with the meaning: {art, soccer, blood, bank, oil}. Nevertheless, the experimental results show that this discard will enhance the performance in some cases as demonstrated in this section (Table 4).

Regarding the word frequency, it is the number of occurrences of each word in the entire corpus. For example, it is possible to select any word that appears more than one time to be considered as a feature entity; otherwise, the word will not be selected. This is important for two reasons; SVD finds a better correlation between words, hence, a single word has no correlation. The other reason is to remove typo words that usually appear once. The results are presented in two subsections; the first is considered the cosine classifier using TF.IDF and LSI features, and the other is for comparing the cosine classifier with other classifies.

6.1. Performance of TF.IDF and LSI features

In this experiment, we used cosine similarity to measure the classification performance using TF.IDF and LSI features. The required parameters were set as follows. The word frequency threshold is set to 1 (remove word that appears only once). The small word threshold was set to 1 (remove the words of one character length, less than or equal this threshold). For the rank k approximation, a range is selected to find the best performance using different k values. The range was chosen to start at 10, 12, 14,... up to 100. Fig. 3 shows that the best accuracy was achieved at $k = 46$. The $k = 46$ was considered as the baseline and used when comparing the cosine classifier with the other listed classifiers in the next subsection.

At $k = 46$, the accuracy scored 82.5% based on the LSA features. For TF.IDF performance, the same parameters were used (i.e. the word frequency threshold = 1, and the small words threshold = 1, TF.IDF does need require k value), the accuracy of TF.IDF scored 67.25% as shown in Table 4.

To investigate whether the LSI significantly outperforms the TF.IDF; the performance detection method proposed by Plötz (2005) was used. The confidence interval $[\varepsilon_l, \varepsilon_u]$ has to be computed at

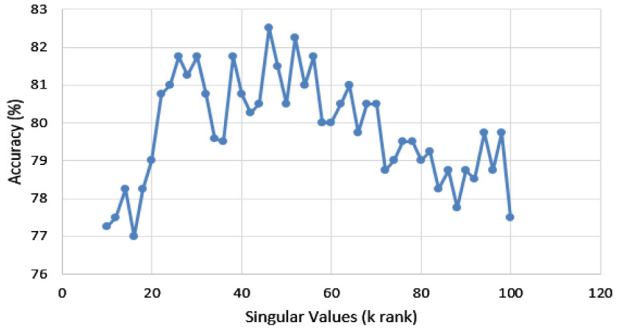

Figure 3. Accuracy of different singular values.

$$\mathcal{E}_l = \frac{N}{N+z^2}\left(\hat{\mathcal{E}} + \frac{z^2}{2N} - z\sqrt{\frac{\hat{\mathcal{E}}(1-\hat{\mathcal{E}})}{N} + \frac{z^2}{4N^2}}\right)$$

$$\mathcal{E}_u = \frac{N}{N+z^2}\left(\hat{\mathcal{E}} + \frac{z^2}{2N} + z\sqrt{\frac{\hat{\mathcal{E}}(1-\hat{\mathcal{E}})}{N} + \frac{z^2}{4N^2}}\right)$$

Figure 4. Confidence interval calculation formula.

the first place. Fig. 4 shows how to find the confidence interval. N is set to the value 400, the number of documents in the Testing set. If the changed classification error rate is outside the confidence interval, these changes can be interpreted as statistically significant. Otherwise, they were most likely caused by chance. We used 95% as a level of confidence. We also used the error probabilities of the TF.IDF method, as 32.75% (100–67.25%) as reported in Table 4. Since we used 95% as a level of confidence, z is equal to 1.96 from the standard normal distribution. It might be interpreted as a 95% probability that a standard normal variable, z, will fall between −1.96 and 1.96.

The confidence interval is found to be [32.75% − 4.42, 32.75% + 4.74] → [28.33%, 37.49%]. Since the error probabilities using the LSI method is 17.5% (100–82.5%), we consider that using LSI features significantly outperform the TF.IDF features as 17.5% is outside the confidence interval.

To invest the effect of the small word removal, we performed experiments at $k = 46$, small word threshold was set at different values {2, 3, 4, 5, 6, 7, 8} as indicated in Table 5. The first row entries in the table is the baseline settings for the small word threshold. The word frequency threshold was set at 1. The results are presented in Table 5.

Table 5 shows that the performance was enhanced by removing small words in the case of TF.IDF when removing small words up to less than or equal to 6 characters. Then, it decreases as many of the discriminator words were removed. In the case of LSI, there are some cases that the performance was increased such as removing the words of less than or equal to 2, 4, and 5. The information provided in Table 5 shows that a large number of small words could be discarded while obtaining better performance. In the case of less than or equal to five characters (⩽5), 1,339,592 small words were removed with a better performance. In fact, this is extremely important in the proposed method as the time complexity is linear in the size of the training set, which scales poorly to a large dataset. More research can be conducted to find the optimal performance when discarding such a large amount of noise.

6.2. Performance of LSI using different classifiers

In this section, the performance was compared between the cosine classifier and the other eight classifiers {NN, NB, k-NN, SVM, RF, CT, LR, and CN2 Rules}. The LSI features were used in this evaluation. The performance of the cosine classifier was already obtained using the Gensim as indicated in the previous subsection. The Orange tool was used for the other classifiers. Fig. 5 shows a snapshot of the Orange tool with the implemented classifiers.

In the experiments, we used the accuracy as performance metric. Since the Testing data set has an equal number of documents (i.e. 40 for each class), the accuracy and F-1 measure will have equal values. Therefore, the accuracy is used which is simply the ratio of correctly predicted observations to the number of actuals.

Table 5
The performance of removing small words.

Small words threshold	# of removed words	# of remaining words	TF.IDF accuracy (%)	LSI accuracy at $k = 46$ (%)
Baseline = 1	8409	2,118,788	67.25	82.50
⩽2	271,836	1,855,361	**69.25**	**83.00**
⩽3	590,960	1,536,237	**69.00**	80.75
⩽4	948,454	1,178,743	**68.25**	**83.50**
⩽5	1,339,592	787,605	**71.25**	**83.50**
⩽6	1,664,139	463,058	**70.75**	80.75
⩽7	1,923,017	204,180	66.75	73.75

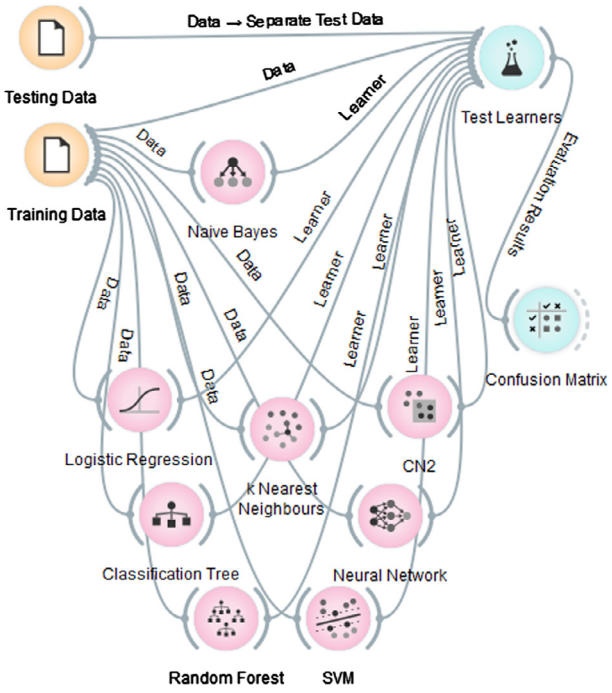

Figure 5. A snapshot of Orange tool.

Table 6
The performance of the classifiers.

Classifier	Accuracy (%)
SVM	**84.75**
Cosine	82.50
LR	81.25
k-NN	77.25
NN	76.75
RF	74.75
NB	65.26
CT	54.00
CN2	47.25

Table 7
The significance test of the classifiers.

Classifier	Accuracy (%)	Error (%)	Inside the confidence interval
SVM	84.75	15.25	Yes
Cosine	82.50	17.50	Yes – the reference
LR	81.25	18.75	Yes
k-NN	77.25	22.75	No
NN	76.75	23.25	No
RF	74.75	25.25	No
NB	65.26	34.74	No
CT	54.00	46.00	No
CN2	47.25	52.75	No

We performed the comparison using the the following parameters: {word frequency = 1, small word length = 1, k = 46}. Table 6 shows that the SVM classifier outperforms all other classifiers followed by cosine measure.

The confidence interval was calculated using the cosine accuracy and found to be [14.09, 21.53]. Table 7 shows that the SVM is inside the confidence interval which means that cosine and SVM and LR have the same performance from the statistical point of view. Hence, the shaded classifiers in Table 7 were found to score the best performance.

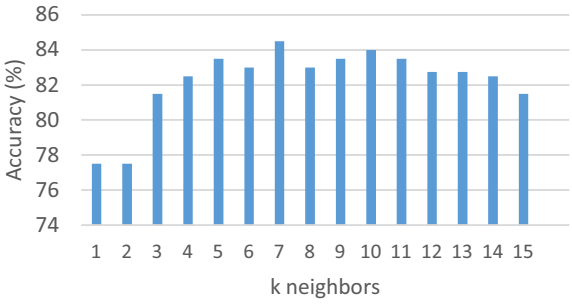

Figure 6. The performance with different k values in k-NN.

The k-NN performance shown in Table 7 was measured using Manhattan distance (with k = 10 neighbors). The Manhattan distance was found to achieve a better performance than other distance measures such as Euclidian, Hamming, and Maximal. However, as this research mainly focuses on the cosine similarity measure, we conducted more research on the performance of k-NN based on cosine measure. Neither Orange nor Weka ("Weka", 2016) tools provide the option to implement k-NN with cosine measure. Therefore, we used the RapidMiner (RapidMiner, 2016) machine-learning tool that was found to provide this option (i.e. k-NN with cosine measure). Hence, the performance was measured using a different value of k in the k-NN classifier. Fig. 6 shows the accuracies achieved using k-NN based on cosine measure. The results found to be better than other measures such as Manhattan distance measure that was the best using the Orange tool.

In Fig. 6, the highest accuracy was 84.5% at k = 7. We also reinvestigated the performance of SVM using the RapidMiner tool and found to be 84.5% (almost same with the accuracy achieved using Orange tool, 84.75%). Hence, our findings indicate that k-NN based on cosine measure scored the same accuracy as SVM, the powerful classification method.

7. Conclusion and future work

This paper shows that cosine similarity measure is a good option to be considered for the Arabic language text classification. It also provides an experimental comparison between eight text classification methods. The results show that SVM and k-NN (cosine measure based) classifiers have almost the same performance.

As a future direction, we propose to investigate the performance of multi-level and multi-label Arabic text classification. We would also propose investigating the feature reduction methods such as what we proposed in this research (small words threshold) and weighting schemes.

Acknowledgments

This work is supported by Kuwait Foundation of Advancement of Science (KFAS), Research Grant Number P11418EO01 and Kuwait University Research Administration Research Project Number EO06/12.

References

Ababneh, J., Almomani, O., Hadi, W., El-Omari, N.K.T., Al-Ibrahim, A., 2014. Vector space models to classify Arabic text. Int. J. Comput. Trends Technol. (IJCTT) 7 (4), 219–223.

Al-Eid, R.M.B., Al-Khalif, R.S., Al-Salman, A.S., 2010. Measuring the credibility of Arabic text content in Twitter. In: 2010 Fifth International Conference on Digital Information Management (ICDIM). IEEE, pp. 285–291.

Alghamdi, H.M., Selamat, A., 2012. Topic detections in Arabic dark websites using improved vector space model. In: 2012 4th Conference on Data Mining and Optimization (DMO). IEEE, pp. 6–12.
Al-Harbi, S., Almuhareb, A., Al-Thubaity, A., Khorsheed, M., Al-Rajeh, A., 2008. Automatic Arabic text classification. In: JADT;08, France, pp. 77–83.
Al-Kabi, M., Al-Sinjilawi, S., 2007. A comparative study of the efficiency of different measures to classify Arabic text. Univ. Sharjah J. Pure Appl. Sci. 4 (2), 13–26.
Al-Kharashi, I.A., Evens, M.W., 1994. Comparing words, stems, and roots as index terms in an Arabic information retrieval system. J. Am. Soc. Inf. Sci. 45 (8), 548–560.
Alqabas, 2016, January. Retrieved from <http://www.alqabas.com.kw/Default.aspx>.
Alsaleem, S., 2011. Automated Arabic text categorization using SVM and NB. Int. Arab J. e-Technol. 2 (2), 124–128.
Al-Shalabi, R., Obeidat, R., 2008. Improving KNN Arabic text classification with n-grams based document indexing. In: Proceedings of the Sixth International Conference on Informatics and Systems, Cairo, Egypt, pp. 108–112.
Al-Shammari, E.T., 2010. Improving Arabic document categorization: introducing local stem. In: 2010 10th International Conference on Intelligent Systems Design and Applications (ISDA). IEEE, pp. 385–390.
Al-Shargabi, B., Al-Romimah, W., Olayah, F., 2011. A comparative study for Arabic text classification algorithms based on stop words elimination. In: Proceedings of the 2011 International Conference on Intelligent Semantic Web-Services and Applications. ACM, p. 11.
Andrews, H.C., Patterson, C.L., 1976. Singular value decompositions and digital image processing. IEEE Trans. Acoustics Speech Signal Process. 24 (1), 26–53.
Bradford, R.B., 2008. An empirical study of required dimensionality for large-scale latent semantic indexing applications. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management. ACM, pp. 153–162.
Carpineto, C., Osiński, S., Romano, G., Weiss, D., 2009. A survey of web clustering engines. ACM Comput. Surv. (CSUR) 41 (3), 17.
Dhillon, I.S., Modha, D.S., 2001. Concept decompositions for large sparse text data using clustering. Mach. Learn. 42 (1–2), 143–175.
Duwairi, R.M., 2007. Arabic text categorization. Int. Arab J. Inf. Technol. 4 (2), 125–132.
El Gohary, A.F., Sultan, T.I., Hana, M.A., El, M.M., 2013. A computational approach for analyzing and detecting emotions in Arabic text. Int. J. Eng. Res. Appl. (IJERA) 3, 100–107.
Elberrichi, Z., Abidi, K., 2012. Arabic text categorization: a comparative study of different representation modes. Int. Arab J. Inf. Technol. 9 (5), 465–470.
Erkan, G., Radev, D.R., 2004. LexRank: graph-based lexical centrality as salience in text summarization. J. Artif. Intell. Res., 457–479
Ezzat, H., Ezzat, S., El-Beltagy, S., Ghanem, M., 2012. Topicanalyzer: a system for unsupervised multi-label Arabic topic categorization. In: 2012 International Conference on Innovations in Information Technology (IIT). IEEE, pp. 220–225.
Froud, H., Lachkar, A., Ouatik, S.A., 2013. Arabic text summarization based on latent semantic analysis to enhance Arabic documents clustering. arXiv preprint arXiv:1302.1612.
Gensim, 2016, January. Retrieved from <https://radimrehurek.com/gensim/>.
Gharib, T.F., Habib, M.B., Fayed, Z.T., 2009. Arabic text classification using support vector machines. Int. J. Comput. Appl. 16 (4), 192–199.
Google Translate, 2016, January. Retrieved from <https://translate.google.com/>.
Hadni, M., Ouatik, S.A., Lachkar, A., 2013. Effective Arabic stemmer based hybrid approach for Arabic text categorization. Int. J. Data Min. Knowl. Manag. Process (IJDKP) 3.
Harrag, F., Al-Qawasmah, E., 2010. Improving Arabic text categorization using neural network with SVD. JDIM 8 (4), 233–239.
Harrag, F., El-Qawasmeh, E., Pichappan, P., 2009. Improving Arabic text categorization using decision trees. In: First International Conference on Networked Digital Technologies, 2009, NDT'09. IEEE, pp. 110–115.
Hmeidi, I., Hawashin, B., El-Qawasmeh, E., 2008. Performance of KNN and SVM classifiers on full word Arabic articles. Adv. Eng. Inform. 22 (1), 106–111.
Jbara, K., 2010. Knowledge discovery in Al-Hadith using text classification algorithm. J. Am. Sci. 6 (11), 409–419.
Kanaan, G., Al-Shalabi, R., Ghwanmeh, S., Al-Ma'adeed, H., 2009. A comparison of text-classification techniques applied to Arabic text. J. Am. Soc. Inform. Sci. Technol. 60 (9), 1836–1844.
Kantardzic, M., 2011. Data Mining: Concepts, Models, Methods, and Algorithms. John Wiley & Sons.
Khorsheed, M.S., Al-Thubaity, A.O., 2013. Comparative evaluation of text classification techniques using a large diverse Arabic dataset. Lang. Resour. Eval. 47 (2), 513–538.
Larkey, L.S., Feng, F., Connell, M., Lavrenko, V., 2004. Language-specific models in multilingual topic tracking. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 402–409.
Lin, C.H., Chen, H., 1996. An automatic indexing and neural network approach to concept retrieval and classification of multilingual (Chinese-English) documents. IEEE Trans. Systems Man Cybern. Part B: Cybern. 26 (1), 75–88.
Manning, C.D., Schütze, H., 1999. Foundations of Statistical Natural Language Processing. MIT Press.
Moh'd Mesleh, A., 2011. Feature sub-set selection metrics for Arabic text classification. Pattern Recogn. Lett. 32 (14), 1922–1929.
Nguyen, H.V., Bai, L., 2011. Cosine similarity metric learning for face verification. In: Computer Vision–ACCV 2010. Springer, Berlin Heidelberg, pp. 709–720.
Omar, N., Albared, M., Al-Shabi, A., Al-Moslmi, T., 2013. Ensemble of classification algorithms for subjectivity and sentiment analysis of Arabic customers' reviews. Int. J. Adv. Comput. Technol. 14 (5), 77–85.
Orange, 2016, January. Retrieved from <http://orange.biolab.si/>.
Plötz, T., 2005. Advanced stochastic protein sequence analysis PhD Thesis. Faculty of Technology, Bielefeld University.
Raheel, S., Dichy, J., Hassoun, M., 2009. The automatic categorization of Arabic documents by boosting decision trees. In: 2009 Fifth International Conference on Signal-Image Technology & Internet-Based Systems (SITIS). IEEE, pp. 294–301.
RapidMiner, 2016, January. Retrieved from <https://rapidminer.com/>.
Roberts, A., Al-Sulaiti, L., Atwell, E., 2005. aConCorde: towards a proper concordance of Arabic. In: Proceedings of the Corpus Linguistics 2005 Conference. University of Birmingham, UK.
Rosario, B., 2000. Latent semantic indexing: an overview. Tech. Rep. INFOSYS, 240.
Salton, G., Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. 24 (5), 513–523.
Sebastiani, F., 2002. Machine learning in automated text categorization. ACM Comput. Surv. (CSUR) 34 (1), 1–47.
Silber, H.G., McCoy, K.F., 2002. Efficiently computed lexical chains as an intermediate representation for automatic text summarization. Comput. Ling. 28 (4), 487–496.
Sobh, I., Darwish, N., Fayek, M., 2006. A trainable Arabic Bayesian extractive generic text summarizer. In: Proceedings of the Sixth Conference on Language Engineering ESLEC, pp. 49–154.
Sokolova, M., Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. 45 (4), 427–437.
Strehl, A., Ghosh, J., Mooney, R., 2000. Impact of similarity measures on web-page clustering. In: Orkshop on Artificial Intelligence for Web Search (AAAI 2000), pp. 58–64.
Syiam, M.M., Fayed, Z.T., Habib, M.B., 2006. An intelligent system for Arabic text categorization. Int. J. Intell. Comput. Inf. Sci. 6 (1), 1–19.
Takçı, H., Güngör, T., 2012. A high performance centroid-based classification approach for language identification. Pattern Recogn. Lett. 33 (16), 2077–2084.
Thabtah, F., Mahazah, M., Hadi, W., 2008. VSMs with K-Nearest Neighbour to Categorise Arabic Text Data. In: Proceedings of the International Conference on Machine Learning and Data Analysis 2008 (ICMLDA), Oct. 2008, San Francisco, CA, USA.
Theodoridis, S., Koutroumbas, K., 2008. Pattern Recognition. Academic Press.
Torunoğlu, D., Çakırman, E., Ganiz, M.C., Akyokuş, S., Gürbüz, M.Z., 2011. Analysis of preprocessing methods on classification of Turkish texts. In: 2011 International Symposium on Innovations in Intelligent Systems and Applications (INISTA). IEEE, pp. 112–117.
Weka, 2016, January. Retrieved from <http://www.cs.waikato.ac.nz/ml/weka/>.
Zrigui, M., Ayadi, R., Mars, M., Maraoui, M., 2012. Arabic text classification framework based on latent Dirichlet allocation, CIT. J. Comput. Inf. Technol. 20 (2), 125–140.