12-1-2020

A set theory based similarity measure for text clustering and classification

Ali A. Amer
Taiz University

Hassan I. Abdalla
Zayed University, hassan.abdalla@zu.ac.ae

Amer and Abdalla *J Big Data* (2020) 7:74
https://doi.org/10.1186/s40537-020-00344-3

Journal of Big Data

RESEARCH

Open Access

A set theory based similarity measure for text clustering and classification

Ali A. Amer[1*] and Hassan I. Abdalla[2]

*Correspondence: aliaaa2004@yahoo.com
[1] Computer Science Department, Taiz University, Taiz, Yemen
Full list of author information is available at the end of the article

Abstract

Similarity measures have long been utilized in information retrieval and machine learning domains for multi-purposes including text retrieval, text clustering, text summarization, plagiarism detection, and several other text-processing applications. However, the problem with these measures is that, until recently, there has never been one single measure recorded to be highly effective and efficient at the same time. Thus, the quest for an efficient and effective similarity measure is still an open-ended challenge. This study, in consequence, introduces a new highly-effective and time-efficient similarity measure for text clustering and classification. Furthermore, the study aims to provide a comprehensive scrutinization for seven of the most widely used similarity measures, mainly concerning their effectiveness and efficiency. Using the K-nearest neighbor algorithm (KNN) for classification, the K-means algorithm for clustering, and the bag of word (BoW) model for feature selection, all similarity measures are carefully examined in detail. The experimental evaluation has been made on two of the most popular datasets, namely, Reuters-21 and Web-KB. The obtained results confirm that the proposed set theory-based similarity measure (STB-SM), as a pre-eminent measure, outweighs all state-of-art measures significantly with regards to both effectiveness and efficiency.

Keywords: Information retrieval, Text retrieval, Text classification, Similarity measures, Empirical study

Introduction

In information retrieval and machine learning, a good number of techniques utilize the similarity/distance measures to perform many different tasks [1]. Clustering and classification are the most widely-used techniques for the task of knowledge discovery within the scientific fields [2–10]. On the other hand, text classification and clustering have long been vital research areas of information retrieval (IR). While text classification is the process of classifying the text/document into its actual class by utilizing a similarity measure and a proper classifier. The clustering, on the other hand, is the process of grouping similar texts into similar groups called clusters. As a matter of fact, with the ever-piling amount of data and information on the internet, the necessity for a highly effective classification algorithm is urgent. Nevertheless, the enhancement of classification performance has still been the main task for researchers in the text mining field.

Given the fact that the similarity/distance measures are the core component of the classification and clustering algorithm, their efficiency and effectiveness directly impact techniques' performance in one way or another. Therefore, the selection of the best similarity measure for the techniques in question is still an open-ended challenging task.

Even though there have been several proposed works in IR literature to compare the similarity/distance measures for clustering and classification purposes [2, 3, 11–16], these studies are still incapable of providing a comprehensive preview of the actual performance of similarity measures. Besides, some of those works have presented an efficient similarity measure while ignoring effectiveness [17, 21]. While the others have presented only an effective similarity measure while ignoring efficiency [2–4]. Consequently, this work comes to cover this critical limitation by introducing a compromised (effective and time-efficient) similarity measure while the most widely used similarity measures are elegantly investigated in a thorough pattern under numerous circumstances. Using the K-nearest neighbor classifier (KNN), K-means clustering algorithm, and the bag of words (BoW) representation model [17–19] for feature selection, the similarity measures are examined in details. The K values (in KNN) is varied from (1) to (120) and the number of features is set to be in (50, 100, 300, 1000, 3000, 6000, and the whole number of features of the considered dataset). In doing so, the superiority of STB-SM measure is emphasized, and each measure is tested under several circumstances so the desired effectiveness including accuracy is being obtained in certain K values on several features. These measures are evaluated against low dimensional datasets (by studying their performance on 50, 100, 200, and 350) and high dimensional datasets (by studying their performance on 3000, 6000, and the number of all features of the dataset). The measures' behavior has been analyzed to determine which measure gives the best results in certain K values on a specific number of features. Furthermore, for the clustering performance analysis, five evaluation metrics were employed with two of them are internal and three are external. The key objective of this work is to present a new competitive measure, compare and benchmark the similarity measures performance on the targeted datasets on both the low and the high-dimensional datasets. Briefly, the main contributions of this work are listed below:

1. Introducing a novel similarity measure for text retrieval that basically behaves based on the set theory mechanism. This measure has been named a set theory based similarity measure for text retrieval (STB-SM). In accordance with the experimental results of both classification and clustering, STB-SM has been shown to be a promising measure with its being superior over the existing state-of-the-art measures.
2. Along with proposing the STB-SM, seven similarity measures, that are commonly applied for text retrieval and machine learning purposes, are thoroughly investigated and evaluated to benchmark their impact on text retrieval. They are comprehensively tested on two of the most publicly available datasets (namely, web-KB and Reuters-21). Using BoW, a thorough comparative analysis for these measures, in terms of their effectiveness and efficiency, are drawn. While the classification effectiveness includes six evaluation factors, namely; accuracy, precision (PRE), recall (REC), F-Measure (FM), G-Measure (GM) and Average Precision Mean (AMP). The clustering effectiveness includes five evaluation metrics namely, Purity, Completeness and Rand Index as the external metrics, along

with Calinski-Harabasz index and Davies-Bouldin index as the internal metrics. Moreover, for both classification and clustering efficiency, the run time, taken by each measure to find the similarity degree, is rigorously observed.

3. The scope of this work concentrates on promoting the performance of text clustering and classification through a new measure along with a detailed comparative analysis for the proposed measure against the state-of-art BoW-based similarity measures. The drawn analyses would provide an influential guide for the selection of similarity measures in terms of considered datasets as well as helping researchers in fully understanding the present and future challenges linked with text retrieval.

The rest of this paper is structured as follows: the most relevant similarity measures for this study are concisely presented in Sect. "Related work". Section "The set theory" briefly describes the basics and definitions of set theory in the context of text retrieval. Section "The proposed similarity measure (STB-SM)" defines, formulates, and analyzes the proposed similarity measure in the context of the set theory. The experimental setup is drawn in Sect. "Experimental setup". The results of the work are given in Sect. "Experimental results". The discussion is profoundly detailed in Sect. "Discussion". Finally, conclusions and future work recommendations are presented in Sect. "Conclusions and future work".

Related work

Vector Space Model (VSM) has long been used to represent document(s) when dealing with text retrieval. In VSM, each document is drawn as an N-dimensional vector. Each dimension represents a vocabulary term/feature. In information retrieval (IR) literature, there are a good number of similarity measures to compute the pairwise document similarity using VSM. While there have been some works that have been proposed in the IR literature to perform the clustering along with the classification using the similarity/distance measures [2–4, 11–16]. These works lack the comprehensive preview of the actual performance of similarity measures. Moreover, some of them have proposed efficient similarity measures irrespective of their effectiveness [21, 22]. Other works, however, have presented only effective similarity measures without consideration to their efficiency [2–4].

Euclidean and Manhattan distances are among the most famous geometric measures which have been utilized to find the distance between each vector pair [2, 20]. Similarly, Cosine similarity finds similarity between each document pair using the angle between their vectors [10]. The triangle distance is also looked at as the Cosine of a triangle between vector pair [10]. The value of this measure range between 0 and 2. On the other hand, for 0–1 vectors, the Hamming distance [4] is used to give the number of positions at which the feature weights are not equal. Kullback–Leibler divergences [23, 24], KLD, as a non-symmetric measure was used in [24] to compute the similarity between each vector pair using the probability distribution that is associated with the both vectors. In [4], a similarity measure for text processing, named SMTP, was found to calculate the similarity between document pair. An Information-Theoretic measure (IT-Sim), was proposed based on information theory in [18] for document Similarity purposes. In [3], a new similarity measure called Improved Sqrt-Cosine (ISC) was proposed. Meanwhile, Bhattacharya coefficient was invented in [21] to approximately calculate the overlap rate between each statistical sample pair. Jaccard coefficient was developed in [25] to find similarity using the ratio of the number of features existing in both documents to the number of features existing in at least one of them. Subsequently in [2], a new similarity measures named pairwise document similarity measure based on present term set (PDSM), was presented based on the feature weights as well as the number of features that existed in at least one of the considered documents.

Some of these measures have shown to be highly effective such as the PDSM [2], the ISC [3], and the SMTP [4], yet unfortunately time-inefficient. In contrast, some measures are not effective yet highly efficient notably the Euclidean and Manhattan. Cosine, on the other hand, has been seen as a compromised solution as an effective and highly efficient measure. Furthermore, as reported in IR literature, almost all of these measures were tested in the context of text classification and clustering. For example, PDSM was compared in [2] with five similarity measures in terms of classification and near duplicate application. Likewise, ISC [3] and SMTP [4] were evaluated against several similarity measures concerning text classification and clustering. Similarly, our proposed paper of this work has been evaluated against some of the most widely used similarity measures in machine learning and information retrieval literature, particularly with respect to text classification and clustering. Finally, [7] assessing the clustering performance of several measures on three collections of web documents. The experimental results of their experiment revealed that Cosine similarity outweighs both the Jaccard coefficient and the Euclidean distance.

The most relevant similarity measures

In this sub-section, the similarity measures that are considered to conduct this study are presented. Seven similarity measures are introduced as the most widely used measures for text clustering and classification [2, 20–24]. These similarity measures work by considering the terms' presence and absence, or by evaluating the angle between each vector pairs or by finding the distance. Assuming that we have two documents doc1 and doc2 that have two vectors d1 and d2, the aim is to find how much similarities are there when using the intended similarity measure as follows;

Euclidean distance (ED)

Every document is drawn as a point in 2D space depending on the term frequency of N terms that would represent the N dimension. ED finds the similarity between each point pair in N-dimensional space using their coordinate based on the following equation:

$$D_{Euc}(doc1, doc2) = \sum \sqrt{(doc_{11-}doc_{12})^2 + (doc_{21-}doc_{22})^2 + \ldots (doc_{n1-}doc_{n2})^2} \tag{1}$$

Manhattan

Manhattan distance (known as sum-norm) finds the sum of absolute differences between the targeted coordinates of each document pair vectors as follows:

$$Manhattan - distance(doc1, doc2) = \sum_{i=1}^{n} \left| doc1_{w1} - doc2_{w2} \right| \quad (2)$$

Cosine similarity measure

The Cosine similarity calculates the pairwise similarity between the document pairs using the dot product and the magnitude of both vectors of both documents. It is mostly utilized within the scientific fields including the IR field [20], and is defined as follows:

$$Sim_{Cos}(doc1, doc2) = \frac{\sum_{i=1}^{n} (doc_{i1} * doc_{i2})}{\sqrt{\sum_{i=1}^{n} doc_{i1}^2} * \sqrt{\sum_{i=1}^{n} doc_{i2}^2}} \quad (3)$$

The union is used to normalize the inner product.

Jaccard similarity measure

This coefficient was invented in [25] to divide the intersection of the points by their unions, and the value of coefficient ranges between 0 (there is no similarity between the documents) and 1 (both documents are identical). The Jaccard similarity is given by the next equation:

$$Sim_{jaccard}(doc1, doc2) = \frac{doc1 \cap doc2}{doc1 \bigcup doc2} \quad (4)$$

Bhattacharya coefficient

The Bhattacharyya coefficient is used to approximately calculate the overlap rate between each statistical sample pair [21]. In our works, however, these samples are thought of as documents. This coefficient is being utilized to find the approximate closeness of each document pair.

$$Sim_{Bhatta}(doc1, doc2) = 1 - \log\left(\sum_{i=1}^{n} \sqrt{doc_{i1*}doc_{i2}}\right) \quad (5)$$

Kullback–Leibler divergence

It is also known as a "relative entropy" [23, 24]. It is used to measure the difference between probability distributions. Simply, when this measure reaches 0, it signals that the intended distributions pair is identical, following that, its equation is then drawn as follow;

$$Sim_{KL}(doc1, doc2) = \sum_{i=1}^{n} (doc_{i1}) * \log\left(\frac{doc_{i1}}{doc_{i2}}\right) \quad (6)$$

PDSM

This measure has been introduced in [2] to tackle the limitation of the-state-of-art measures which included a number of present terms into account. PDSM was seen effective according to the experimental results of [2] as well as the experimental results of our current work. The PDSM equation is formulated as follows

$$D_{pdsm}(doc1, doc2) = \frac{doc_{i1} \cap doc_{i2}}{doc_{i1} U doc_{i2}} * \frac{PF(doc_{i1} doc_{i2})}{M - AF(doc_{i1}, doc_{i2}) + 1} \tag{7}$$

where

$$doc_{i1} \cap doc_{i2} = \min(doc_{i1}, doc_{i2})$$

$$doc_{i1} U doc_{i2} = \max(doc_{i1}, doc_{i2})$$

where $PF(doc_{i1} doc_{i2})$ represents the number of present terms and $AF(doc_{i1} doc_{i2})$ represents the number of absent terms and M is the total number of documents.

The set theory

Before introducing the proposed measure, some basics and definitions (upon which our measure behaves) for the set theory in the context of text retrieval should be conceived. So, in this section, the main objective is to introduce the relative set theory operations upon which our proposed measure behaves.

Generally speaking, the set theory is a vital component of modern mathematics and is widely used in all formal descriptions. The set can be a collection, a group, or even a cluster of points that are named members of that set. For instance, a set of documents is a collection of documents, or a set of people is a group of people, etc. For each point to be a member of that set, its membership shall be defined clearly. However, sometimes, due to the lack of information, membership definition is a difficult task and may even be a vague. So, if the membership definition is vague for some collection, the collection is then cannot be called a set. Simply put, if there has been a set S and its two members X and Y, then it shall not be unknown whether X = Y or they are not. Strictly speaking, the set can be either finite, infinite, or empty. In the following, some basic definitions and key operations are introduced to further understand the basics upon which STB-SM measure behaves.

Definition 1 If we have two sets S1 and S2, both sets are equal if and only if they have the same points, and then every $X \in S1 \Leftrightarrow X \in S2$. For example, in the context of text retrieval, if we have Doc1{Ali, Jun, Sarah} and Doc2{Jun, Sarah, Ali}. Then, we can say that Doc1 = Doc2, and they are both identical as every word belongs to Doc1 also belongs to Doc2.

Definition 2 If we have two sets S1 and S2, S1 is "a proper" subset of S2 (S1 ⊆ S2) if there has been X ∈ S1 and also X ∈ S2 as well. For example, in the context of text retrieval, if we have Doc1{Ali, Hassan, Sarah} and Doc2{Hassan, Sarah, Ali, Mark, Farah}. Then, we can say that Doc1 ⊆ Doc2, and Doc1 is a proper subset of S2 as every word belongs to Doc1 also belongs to Doc2.

Definition 3 he document doc is a collection of terms of vectors that holds these terms, that is, any subset of C, when C is the document collection, (involving C itself).

Let doc be a document, a subset of C. We say that doc exists as a vector if the terms of doc exist in the doc itself. First, let us define the key relationships between each document pair doc1 and doc2 in the collection C, as follows;

$$doc1 \subset doc2 \Leftrightarrow T \in doc1 \Rightarrow T \in doc2(containment)$$

$$doc1 = doc2 \Leftrightarrow doc1 \subset doc2 and doc2 \subset doc1\left(equality\right)$$

So, for the given document pair doc1 and doc2, the following set of operations are held as follows;

Operation 1—union

The union of two sets S1 and S2 ($S1 \cup S2$), is the set that contains all the elements of both sets S1 and S2 with the removal of duplication.

$$S1 \cup S2 = \{X|X \in S1 or X \in S2\}$$

In the context of text retrieval, the Union operation of doc1 and doc2, doc1 ∪ doc2, is the group of terms $\{t_1,\ldots, t_n\}$ where n is the number of addressed terms in both documents, that are involved in either doc1, doc2 or both:

$$doc1 \cup doc2 = \left\{t : t \in doc1 \text{or } t \in doc2\right\}.$$

Operation 2—intersection

The Intersection of two sets S1 and S2 ($S1 \cap S2$), is the set that contains shared elements of sets S1 and S2.

$$S1 \cap S2 = \left\{X|X \in S1 and X \in S2\right\}$$

In the context of text retrieval, the Intersection operation of doc1 and doc2, doc1 ∩ doc2, is the group of terms $\{t_1,\ldots, t_n\}$ where n is the number of addressed terms in both documents, that are involved in both documents doc1 and doc2 at the same time:

$$doc1 \cap doc2 = \left\{t : t \in doc1 \text{and } t \in doc2\right\}.$$

Operation 3—negation

The negation operation of doc1 or doc2, doc1/doc2 or doc2/doc1, is the group of terms that are either belongs to doc2/doc1 or doc1/doc2:

$$doc1\backslash doc2 = \left\{t : t \notin doc2\right\}.$$

$$doc2\backslash doc1 = \left\{t : t \notin doc1\right\}.$$

The proposed similarity measure (STB-SM)

The formulation of STB-SM similarity measure

Suppose we have a document pair doc 1 and doc2. Let doc1 = (w_{11}, w_{12},...) and doc2 = (w_{21}, w_{22},...) be the weighting vectors (using BoW model) of the term sets for document 1 and document 2, respectively. Let T_1 {t_{11}, t_{12},... t_{1n}} and T_2 {t_{21}, t_{22},... t_{2n}} be the sets of items that are contained by doc1 and doc2, respectively. For the sake of simplicity, the following is the proposed STB-SM equations:

$$X = \left(\sum_{t \in doc_1 \cap doc_2} W_{1j}\right) * \left(\sum_{t \in doc_1 \cap doc_2} W_{2j}\right) \tag{8}$$

$$Y = \left(\sum_{t \in doc_1 \backslash doc_2} W_{1j}\right) * \left(\sum_{t \in doc \backslash doc} W_{2j}\right) \tag{9}$$

$$Z = \left(\sum_{t \in doc_1} W_{1j}\right) * \left(\sum_{t \in doc_2} W_{2j}\right) \tag{10}$$

$$\mathrm{STB} - \mathrm{SM}(doc_1, doc_2) = \frac{X}{Z} * \left(1 - \frac{Y}{Z}\right) \tag{11}$$

where the notations "∩" and "\" denote the intersection and complement operators in the set theory, and W_{ij} is the weighting value. To further understand the mechanism of this measure and briefly clarify some deficit of the state-of-the-art measures, we have provided three examples as follows:

Example 1 Assuming we have doc1 (2, 5, 7, 8, 0, 9) and doc2 (9, 0, 0, 6. 5, 1), then STB-SM will work as follows; (for simplicity, X is x1 and x2; Y is y1 and y2, Z is z1 and z1, T_i.w suggests weighting of the term i)

	T1.w	T2.w	T3.w	T4.w	T5.w	T6.w
Doc1	2	5	7	8	0	9
Doc2	9	0	0	6	5	1

X1 = 2+8 + 9=19; X2 = 9+6 + 1=16; Z1 = 2+5 + 7+8 + 9=31; Z2 = 9+6 + 5+1 = 21; Y1 = 5+7 = 12; Y2 = 5

While STB-SM yielded (0.47 * 0.91 = 0.43) Cosine and Jaccard yielded (0.42) and (0.22) respectively.

Example 2 Assuming we have doc1 (02, 1, 1, 0, 1) and doc2 (3, 1, 1, 1, 1, 0), then STB-SM will work as follows;

	T1.w	T2.w	T3.w	T4.w	T5.w	T6.w
Doc1	0	2	1	1	0	1
Doc2	3	1	1	1	1	0

$X1=4$ $X2=3$ $Z1=5$ $Z2=7$ $Y1=1$ $Y2=4$

While STB-SM yielded ($0.34 * 0.89=0.30$), Cosine and Jaccard yielded (0.42) and (0.50) respectively.

Example 3 Assuming we have doc1 (1, 1, 3) and doc2 (1, 0, 2), then STB-SM will work as follows;

	T1.w	T2.w	T3.w
Doc1	1	1	3
Doc2	1	0	2

$X1=4$ $X2=3$ $Z1=5$ $Z2=3$ $Y1=1$ $Y2=0$

While STB-SM yielded (0.80), Cosine and Jaccard yielded (0.94) and (0.25) respectively.

As seen from the drawn examples above, Cosine occasionally finds a good similarity as indicated in example (1). However, the Cosine similarity gives the same value for both examples (1 & 2) albeit the clear difference between both vectors, and to further exacerbate the issue the similarity value is highly exaggerated in example 3. It is worth indicating that one novelty of STB-SM measure, is that the similarity value has never been exaggerated as shown in example (3) for Cosine, or the more state-of-the-art measure. STB-SM measure enables non-zero/non-shared features to have an explicit contribution to the similarity computation. Therefore, STB-SM takes the presence and absence of all features into consideration effectively.

On the other hand, Jaccard occasionally produces a good similarity as shown in example (2), but more frequently the Jaccard similarity is poor, as indicated in examples (1 & 3). Our proposed measure, therefore, comes to find a compromised solution where the desired effect is being detected. Examples (1 & 3) show a better and more accurate similarity found by STB-SM in comparison with the Cosine and Jaccard.

STB-SM analysis

In this subsection, we concisely as well as informatively analyze the cases of the proposed measure as follows;

The worst-case:

This case occurs when there is not even one shared feature between the document vectors.

Example (worst case): Assuming we have doc1 (3, 0, 1) and doc2 (0, 2, 0). By applying the worst-case scenario, we find that $X1=0$, $X2=0$; $Z1=4$, $z2=2$, $y1=4$, $y2=2$;

because X = zero. Accordingly, STB-SM = zero, for both documents (1, 0, 1) and (0, 1, 0), which is logically true since there is no shared feature exist.

The average case:

This occurs when there has been at least one shared feature(s) as given in the drawn above examples (1–3). In this case, STB-SM would have a value in the range [0–1].

The best case:

This occurs when both vectors are completely equivalent.

Example (best case): Assuming we have doc1 (4, 4, 4) and doc2 (4, 4, 4), or doc1 (1, 1, 1) and doc2 (1, 1, 1). By applying the best-case scenario, we find that x1 = 9, x2 = 9, z1 = 9, z2 = 9, y1 = 0, y2 = 0. Accordingly, STB-SM = 1 which is logically true as both documents are identical.

The properties of similarity measures

According to [2, 4], six vital properties every similarity measure should have for the relative measure to be considered an optimal measure. The following properties are listed below;

Property 1: *The existence or non- existence of the intended feature is more vital than the difference between the values linked with the existing feature. According to the calculated-above examples, STB-SM explicitly takes the presence and absence of features into consideration.*

Property 2: *The value of similarity should be grown as the difference between the values of non-zero features values decline. For instance, if we have f1 and f2 as two features belong to doc1 and doc2 respectively. Then, for doc1 and doc2, the value of similarity between f1 = 12 and f2 = 6 is higher than the similarity between f1 = 20 and f2 = 6. This property is also clearly shown in example 3, along with the worst-case example.*

Property 3: *The value of similarity should be reduced as the number of existent or non- existent features rises. This was showcased in both the worst and best case examples, clearly indicating the applicability of his property.*

Property 4: *Any pair of documents is low similar to each other if there have been many non-zero-valued features corresponding to many zero-valued features in the same pair. For instance, if we have two vectors for two documents doc1(f1,f2) = (1,0) and doc2(f3, f4) = (1,1). Then, doc1.f2 and doc2.f4 are the key cause for lowering the similarity between both documents as f2 X f4 = 0 and, at the same time, f2 + f4 > 0. Example 2 supports the applicability of this property.*

Property 5: *The similarity measure should possess asymmetrical features. For instance, the similarity between both doc*1 (1, 1, 0) *and doc*2 (1,1,1) *must be the same when doc* 2(1,1,1) *and doc* 1(1, 1, 0) *are considered. According to the drawn above examples, STB-SM enjoys this property completely.*

Property 6: *The distribution value should have a contribution to the similarity between every two documents. That means features with higher spread (standard deviation) contribute more in similarity than that of a lower spread.*

Experimental setup

Text pre-processing

Some operations were carried out normally for the text to be transformed into text vectors for processing. The text was converted from the lower case to upper case, numbers, punctuations, and stop words (common words), in addition to that extra white space were all removed, and some particular symbols (such as $, %)were converted into spaces.

Text representation

The bag of words (BoW) model [26, 27] was used to represent documents that were in the vector space model (VSM). The BoW model represents each document as a word collection disregarding the grammar and word order [28].

Given the fact that we have used a python to run the text pre-processing, the pre-processing was performed using the Ntlk (Natural language toolkit) library of python as follows;

- Tokenization: using the ntlk word tokenizer
- Converting all the words to lower case: using the lower() python string function
- Lemmatizing: using the ntlk stem WordNetLemmatizer
- Stopword Removal: using the ntlk stopwords
- Considering words with only 4 or more letters

The comparison mechanism of classification

After pre-processing, all of the documents were represented using the BoW model in VSM in order for the classification process to start smoothly. Following that, the performance of every similarity measured across the different kinds of documents was compared and evaluated against each other. Six evaluation measures were used to evaluate, namely, accuracy, precision, recall, F-measure, G-measure, and average mean precision. For each criterion, the KNN algorithm runs from K = 1 to K = 120 over each number of features of each dataset, and the averaged results were accumulated and drawn as given in the Tables below (5, 6, 7, 8, 9, 10, 11, 12, 13). Number of features (NF) was varied from NF = 10, NF = 50; NF = 100, NF = 200; NF = 350, NF = 3000, NF = 6000 and NF = the whole number of features (see Appendix samples). In consequence, we have eight runs for the KNN algorithm over two datasets to test and examine six criteria using eight similarity measures. The final number of implementations performed to have the results below were ($8 \times 2 \times 6 \times 8 = 768$) runs. If we also consider the sixty (60) values of K that have been tested in each KNN cycle, the total runs would be 46080.

Term weighting

We adopted the most widely used Term Frequency (TF) technique of weighting which simply gives the occurrence of each word in the relative document [29, 30].

K-nearest neighbor classifier

The K-nearest neighbor algorithm (k-NN) is most widely used, in the IR literature, to perform document classification. Although it is a lazy algorithm [27], it is nonparametric, simple, and believed to be amongst the top ten algorithms in data mining [31]. It works based on selecting the nearest points to the point at the question. The concept of K-NN is that the points that exist in the same class are highly likely to be close to one another depending on the used similarity measure. KNN assumes the next: (1) Points in the feature space have a specific distance between each other and that distance is used as a metric to gauge closeness, (2) Each point in the training points has its vector and class label. Later, a certain number "k" is determined to draw the neighboring area of the point in question.

K-means clustering algorithm

Generally speaking, the clustering of a huge text dataset can be efficaciously made through utilizing the algorithms of partitional clustering. One of the most-popular partitional clustering algorithms is the K-means algorithm. It is widely known in the literature to be the best-fit approach for handling huge volumes of datasets [8, 32]. Similarly to any clustering algorithm, K-means leverages a similarity measure that finds the similarity between each document and the document representative of the cluster (head of the cluster). The similarity measure represents the core of the clustering process by which clustering algorithm performance is analyzed. However, the most suitable similarity measure to effectively perform clustering is still an open-ended challenge. In our work, for the clustering performance analysis, we ran the K-means for each similarity measure, as well as the values of evaluation of metrics (external metrics including purity, completeness and rand index, and internal metrics including the Calinski-Harabasz index and Davies-Bouldin index) were drawn accordingly. We used the voting technique to determine the best similarity measure that would best fit the K-means algorithm. The voting technique worked by enumerating how many metrics each similarity measure had achieved its best values. The bigger number of metrics is the best fit which is the similarity measure. According to the experimental results of the clustering process, our proposed measure (STB-SM) has been seen as the best fit in most cases. It has achieved (11) out of the (20) points by being the best in four metrics out of five. Unfortunately, in the K-means algorithm, the number of clusters is still an ill-posed problem as stated in [32, 33]. Therefore, in this study, we have picked numbers (4 and 8) to be the number of clusters just to analyze and emphasize the behavior of all the similarity measures. It is worth referring that we are not arguing that ($K=4$ or $K=8$) is optimal or the best value for the number of clusters. It is just chosen as the number of actual classes in each dataset [34] to draw the performance analysis of K-means using the considered similarity measures. In the follow-up work, we plan to further examine the performance analysis with

Table 1 Machine and environment description

Task	Tool	Specification
Classification	Language	Python 3, development Software: Jupyter Notebook
	OS	Windows 7 (64 bit)
	Memory	RAM 4 GB
	CPU	Intel Core i5-3320 M (2.6 GHz)
	Dataset	Reuters and Web-KB
Clustering	Language	Python 3, Development Software: Jupyter Notebook
	OS	Windows 10 (64 bit)
	Memory	RAM 32 GB
	CPU	Intel (R) Core ™ (i9-8950 HK CPU 2.90 GHz/32/2TBSSD/4 GB)
	Dataset	Reuters and Web-KB

several K numbers of clusters, and at the same time, with other clustering algorithms, like hierarchical clustering algorithms.

Machine description

Table 1 displays the machine and environment descriptions used to perform this work.

Dataset description

Reuters Dataset (Table 2): Reuters-R8 Dataset holds the eight most frequent classes of the original ninety classes in Reuter's dataset. After applying pre-processing, a total of 18308 features were extracted.

Web-kb dataset (Table 3): it consists of web pages of the computer science department from the following universities: Cornell, Texas, Washington, and Wisconsin. It was obtained from the World Wide Knowledge Base project of the CMU text learning group. After applying the pre-processing, a total of 33,025 features were extracted. The data in both datasets were divided into training and testing in ratio 2:1 (67%: 33%). To overcome the over-fitting or under-fitting issue, instead of dividing the whole data randomly in the training and testing data, each group was divided individually and then combined as training and testing data. Both datasets are read directly from Python platform as they are already integrated with python.

Table 2 Splitting of documents among eight classes in Reuters-R8 dataset

Class	Samples
Cq	2292
Crude	374
Earn	3923
Grain	51
Interest	271
Money-fix	293
Ship	144
Trade	326
Total	7674

Table 3 Splitting of documents among four classes in Web-KB dataset

Class	Samples
Project	504
Course	930
Student	1641
Faculty	1124
Total	4199

The classification evaluation criterions

This subsection holds the evaluation criterions used for classification as follows;

Accuracy (ACC)

ACC checks the total of samples that are correctly classified out of the whole sample collection. ACC is defined in the next equation.

$$\mathrm{ACC} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}} \tag{12}$$

Precision (PRE)

PRE checks the total number of items that are correctly identified as positive out of the total items identified as positive.

$$\mathrm{PRE} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{13}$$

Recall (REC)

REC checks the total number of items that are correctly identified as positive out of the actual positive.

$$\mathrm{REC} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{14}$$

F-measure or F-Score (FM)

FM is the harmonic mean of precision and recall. It is useful when classes are not distributed evenly.

$$\mathrm{FM} = 2 * \frac{\text{Precision*Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

G-method or G-Score (GM)

GM is the geometric mean of precision and recall. It is also used when classes are not distributed evenly.

$$\mathrm{GM} = \frac{\text{True Positive}}{\sqrt{(\text{True Positive} + \text{False Positive}) * \left(\text{True Positive} + \text{False Negative}\right)}} \tag{16}$$

Average mean precision (AMP)

AMP is the mean of the average precision of all classes. This is used to evaluate how precisely the classifier is performing.

$$AMP = \sum_{n} (R_n - R_{n-1})P_n \tag{17}$$

where P_n and R_n are the precision and recall at the nth threshold. Finally, True Positive, True Negative, False Positive and False Negative are defined as follow;

True positive: the number of class1 testing documents that are correctly identified into class1.

True negative: the number of instances of class2, class3,…., classN correctly identified as class2, class3.. classN respectively.

False positive: the number of class1 testing documents that are incorrectly identified into class2, class3,….., classN.

False negative: the number of class2, class3,….., classN testing documents that are incorrectly identified into class1.

The clustering evaluation criterions

This subsection holds the evaluation criterions used for clustering. While the external metrics require actual labels to assess the cluster quality (see Eqs. 18, 19, 20), the internal metrics do not require actual labels to assess the cluster quality (see Eqs. 21, 22).

Accuracy (also known as Purity)

It is used to check the index to which a cluster is pure. Particularly, every cluster has only one class and different clusters have different classes. In other words, this metric evaluates the coherence of a cluster. It is defined by the following equation.

$$\text{Purity} = \frac{1}{N}\sum_{i=1}^{k} max_j \left| c_i \cap t_j \right| \tag{18}$$

Where N is the number of objects(data points), k is the number of clusters, c_i is a cluster in C, and t_j is the classification which has the max count for cluster c_i.

Completeness

To check whether all members of a given class are assigned to the same cluster.

$$\text{H} = \begin{cases} 1 & \text{if } \text{H}(\text{K}, \text{C}) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & \text{else} \end{cases} \tag{19}$$

where

$$\text{H}(\text{K}|\text{C}) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|} \frac{ack}{N} log \frac{ack}{\sum_{k=1}^{|K|} ack}$$

$$\text{H}(\text{K}) = -\sum_{\text{k}=1}^{|\text{K}|} \frac{\sum_{\text{c}=1}^{|\text{C}|} \text{ack}}{\text{n}} \log \frac{\sum_{\text{c}=1}^{|\text{C}|} \text{ack}}{\text{n}}$$

Rand index

It is used to check how many points are correctly predicted.

$$\mathrm{R} = \frac{(a+b)}{nC2} \tag{20}$$

where n is the total number of samples, and (a + b) is the agreement between real and the assigned cluster label.

Calinski-Harabasz index

It is used to measure the ratio between cluster dispersion and inter cluster dispersion.

$$s = \frac{\mathrm{tr}(B_{\boldsymbol{k}})}{\mathrm{tr}(W_{\boldsymbol{k}})} * \frac{n_{\boldsymbol{E}} - k}{k-1} \tag{21}$$

where

$$W_{\boldsymbol{k}} = \sum_{\boldsymbol{q}=1}^{\boldsymbol{k}} \sum_{\boldsymbol{x} \in \boldsymbol{C_q}} \left(x - c_q\right)\left(x - c_q\right)^T$$

$$\boldsymbol{B_k} = \sum_{\boldsymbol{q}=1}^{\boldsymbol{k}} \boldsymbol{n_q}\left(\boldsymbol{c_q} - \boldsymbol{c_E}\right)\left(\boldsymbol{c_q} - \boldsymbol{c_E}\right)^{\boldsymbol{T}}$$

where Cq is the set of points in cluster q, cq is the center of cluster q, c_E the center of E, and n_q is the number of points in cluster q.

Davies-Bouldin index

This index signifies the average ‘similarity’ between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves.

$$DB = \frac{1}{k} \sum_{i=1}^{\boldsymbol{k}} \max_{i \neq j} R_{ij} \tag{22}$$

where

$$R_{ij} = \frac{s_i + s_j}{d_{ij}}$$

where s_i is the average distance between each point of cluster i and the centroid of that cluster, d_{ij} is the distance between cluster centroids i and j. Finally, the best and worst values and the range of each measure are drawn in Table 4.

Table 4 The best and worst values and the range of each metric

Name	Best value	Worst value	Range
Accuracy/purity	1	0	0–1
Completeness	1	0	0–1
Rand Index	1.0	− 1.0	− 1.0–1.0
Calinski-Harabasz index	(Lower value is better)	–	–
Davies-Bouldin index	0	_	0–1

Experimental results

Classification results

This work investigated all considered measures comprehensively based on six criterions for performance evaluation which is the first study of its type to do such investigation in the information retrieval field with respect to text classification. The K values of KNN were varied from (1) to (120) with an increment of value (2) in each cycle (see Appendix samples). The number of features of each dataset was diversified (10, 50, 100, 200, 350, 3000, 6000, NF) to clearly draw the best performance of each measure under several circumstances. Then, for each measure, the results were averaged for all K values on each NF to yield the results drawn in Tables (5, 6, 7, 8, 9, 10, 11, 12, 13). In other words, the following tables contain the results of each similarity measure which were averaged on each Number of Features (NF) over all K values in the range [1…120] as drawn in the appendix. In each table, the averaged results of all K values for each performance criterion is displayed. Table 5 displays the averaged results of all criterions when NF = 10. For simplicity, we draw the averaged results of all measures while analyzing briefly three criterions, namely, ACC, FM, and AMP.

As shown in Table 5, for the Reuters dataset, Euclidean, followed by STB-SM and Cosine, met the highest accuracy. However, STB-SM, followed by Euclidean and kullback–Leibler, outperformed all measures on both FM and AMP criterions. On the other hand, on the Web-KB dataset, PDSM, followed by STB-SM and Cosine, outperformed all similarity measures in ACC. In regard to FM and AMP, Cosine, followed by STB-SM and PDSM, outweighed all measures with STB-SM being superior to PDSM on FM and PDSM being superior to STB-SM on AMP. So, the best measures, when NF = 10, were Euclidean, STB-SM, and Cosine on Reuters, and PDSM followed by STB-SM and Cosine on Web-KB.

Tables 6, 7, 8 9, 10, show that, for both Reuters and Web-KB, STB-SM, followed by PDSM, and Cosine, achieved the highest ACC, FM and AMP respectively. However, two exceptions are noted as follows; the first exception is when NF = 350, Cosine outweighed PDSM in terms of FM and AMP on both Reuters and Web-KB. The second exception is that when NF = 3000, Cosine outperformed PDSM in terms of FM and AMP on Reuters only. Nevertheless, the top performer measures, when NF in the range [50-3000], were STB-SM, PDSM, and Cosine.

Reuters in Tables 11, 12, similarly to Table (10), STB-SM, followed by PDSM and Cosine, had been superior with the highest ACC, FM, and AMP respectively. Moreover Cosine outweighed PDSM in terms of FM and AMP. In contrast, on Web-KB, PDSM, followed by STB-SM and Jaccard, had been superior in terms of ACC and AMP. However, Cosine had been superior to Jaccard in terms of FM only. So, the top performer measures, when NF in the range [6000-All features], were STB-SM, PDSM, Cosine, and

Table 5 Performance evaluation of all measures when, NF = 10–the averaged results (K = 1–120; + 2)

No	Dataset	Reuters-8						Web-KB					
	Similarity/criterion	ACC	PRE	REC	FM	GM	AMP	ACC	PRE	REC	FM	GM	AMP
1	Euclidean	*0.713*	0.317	0.293	*0.286*	0.527	0.217	0.605	0.607	0.515	0.524	0.661	0.429
2	Cosine	*0.694*	0.328	0.311	0.281	0.542	*0.218*	*0.621*	0.610	0.548	*0.562*	0.687	*0.451*
3	Jaccard	0.689	0.299	0.258	0.251	0.492	0.202	0.544	0.617	0.438	0.433	0.560	0.371
4	Bhattacharya	0.654	0.173	0.204	0.180	0.435	0.174	0.458	0.545	0.435	0.381	0.595	0.373
5	kullback–Leibler	0.689	0.383	0.329	*0.292*	0.557	*0.228*	0.613	0.625	0.525	0.526	0.670	*0.436*
6	Manhattan	0.648	0.327	0.284	0.273	0.516	0.205	0.605	0.623	0.515	0.524	0.661	0.432
7	PDSM	0.651	0.339	0.301	0.267	0.533	0.216	*0.626*	0.655	0.533	*0.539*	0.676	*0.448*
8	STB-SM	*0.699*	0.334	0.333	*0.303*	0.562	*0.234*	*0.609*	0.590	0.539	*0.544*	0.679	*0.436*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 6 Performance evaluation of all measures when, NF = 50–the averaged results (K = 1–120; + 2)

No	Dataset	Reuters-8						Web-KB					
	Similarity/criterion	ACC	PRE	REC	FM	GM	AMP	ACC	PRE	REC	FM	GM	AMP
1	Euclidean	0.827	0.641	0.520	0.554	0.708	0.430	0.662	0.740	0.567	0.591	0.701	0.490
2	Cosine	*0.847*	0.656	0.565	*0.592*	0.741	*0.467*	*0.719*	0.735	0.650	*0.671*	0.763	*0.554*
3	Jaccard	0.790	0.580	0.417	0.443	0.631	0.343	0.666	0.808	0.546	0.557	0.687	0.487
4	Bhattacharya	0.803	0.606	0.472	0.468	0.674	0.390	0.492	0.665	0.491	0.382	0.639	0.384
5	kullback–Leibler	0.628	0.617	0.288	0.327	0.510	0.245	0.426	0.645	0.296	0.233	0.476	0.279
6	Manhattan	0.833	0.657	0.551	0.582	0.730	0.455	0.642	0.789	0.538	0.566	0.678	0.479
7	PDSM	*0.857*	0.614	0.571	*0.561*	0.746	*0.443*	*0.757*	0.825	0.670	*0.697*	0.779	*0.598*
8	STB-SM	*0.863*	0.640	0.588	*0.596*	0.757	*0.473*	*0.766*	0.792	0.693	*0.711*	0.795	*0.606*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 7 Performance evaluation of all measures when, NF = 100–the averaged results (K = 1–120; +2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**
1	Euclidean	0.834	0.688	0.543	0.588	0.723	0.456	0.643	0.759	0.541	0.564	0.681	0.474
2	Cosine	*0.875*	0.688	0.602	*0.624*	0.767	*0.502*	*0.737*	0.769	0.666	*0.689*	0.775	*0.578*
3	Jaccard	0.819	0.613	0.450	0.486	0.657	0.373	0.702	0.837	0.584	0.592	0.717	0.526
4	Bhattacharya	0.832	0.639	0.521	0.530	0.710	0.440	0.500	0.643	0.499	0.390	0.644	0.389
5	kullback–Leibler	0.555	0.646	0.211	0.231	0.432	0.197	0.396	0.393	0.260	0.167	0.443	0.256
6	Manhattan	0.830	0.685	0.553	0.594	0.729	0.463	0.594	0.796	0.475	0.490	0.629	0.432
7	PDSM	*0.892*	0.665	0.631	*0.632*	0.787	*0.515*	*0.768*	0.827	0.676	*0.696*	0.784	*0.606*
8	STB-SM	*0.901*	0.700	0.645	*0.658*	0.796	*0.547*	*0.777*	0.819	0.699	*0.715*	0.800	*0.620*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 8 Performance evaluation of all measures when, NF = 200–the averaged results (K = 1–120; +2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**
1	Euclidean	0.814	0.741	0.510	0.573	0.697	0.451	0.636	0.774	0.525	0.543	0.669	0.464
2	Cosine	*0.894*	0.739	0.635	*0.661*	0.788	*0.558*	*0.757*	0.792	0.689	*0.711*	0.791	*0.603*
3	Jaccard	0.846	0.668	0.507	0.548	0.701	0.432	0.736	0.843	0.625	0.635	0.748	0.563
4	Bhattacharya	0.863	0.631	0.590	0.589	0.759	0.450	0.515	0.688	0.510	0.404	0.654	0.397
5	kullback–Leibler	0.530	0.612	0.161	0.149	0.375	0.155	0.391	0.350	0.254	0.152	0.437	0.253
6	Manhattan	0.759	0.712	0.427	0.485	0.630	0.379	0.556	0.815	0.428	0.425	0.591	0.394
7	PDSM	*0.905*	0.724	0.655	*0.663*	0.803	*0.558*	*0.789*	0.845	0.707	*0.732*	0.806	*0.638*
8	STB-SM	*0.914*	0.767	0.669	*0.685*	0.811	*0.590*	*0.791*	0.830	0.721	*0.741*	0.815	*0.644*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 9 Performance evaluation of all measures when, NF = 350–the averaged results (K = 1–120; + 2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**
1	Euclidean	0.778	0.771	0.472	0.544	0.666	0.427	0.632	0.775	0.516	0.529	0.663	0.455
2	Cosine	*0.902*	0.773	0.660	*0.694*	0.804	*0.590*	*0.766*	0.800	0.698	*0.719*	0.798	*0.614*
3	Jaccard	0.861	0.698	0.533	0.573	0.719	0.459	0.764	0.855	0.658	0.670	0.772	0.595
4	Bhattacharya	0.876	0.719	0.618	0.613	0.777	0.525	0.517	0.684	0.510	0.403	0.654	0.395
5	kullback–Leibler	0.520	0.551	0.147	0.125	0.358	0.143	0.390	0.368	0.252	0.147	0.435	0.252
6	Manhattan	0.669	0.731	0.320	0.365	0.535	0.291	0.543	0.826	0.411	0.400	0.577	0.380
7	PDSM	*0.912*	0.770	0.673	*0.689*	0.813	*0.582*	*0.801*	0.853093	0.721095	*0.745*	0.816101	*0.652*
8	STB-SM	*0.922*	0.793	0.694	*0.714*	0.827	*0.619*	*0.801*	0.840	0.731	*0.750*	0.823	*0.655*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 10 Performance evaluation of all measures when, NF = 3000– the averaged results (K = 1-120; +2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**	**ACC**	**PRE**	**REC**	**FM**	**GM**	**AMP**
1	Euclidean	0.648	0.743	0.317	0.376	0.533	0.293	0.570	0.777	0.447	0.446	0.607	0.340
2	Cosine	*0.902*	0.904	0.717	*0.771*	0.837	*0.654*	*0.769*	0.814	0.702	*0.723*	0.801	*0.621*
3	Jaccard	0.868	0.807	0.557	0.610	0.735	0.495	0.782	0.858	0.683	0.701	0.790	0.621
4	Bhattacharya	0.888	0.861	0.682	0.693	0.817	0.590	0.533	0.688	0.525	0.433	0.666	0.406
5	kullback–Leibler	0.508	0.151	0.129	0.091	0.336	0.336	0.389	0.167	0.250	0.141	0.433	0.250
6	Manhattan	0.534	0.408	0.166	0.152	0.377	0.161	0.445	0.748	0.309	0.246	0.488	0.297
7	PDSM	*0.912*	0.891	0.709	*0.748*	0.834	*0.632*	*0.799*	0.851	0.712	*0.730*	0.811	*0.644*
8	STB-SM	*0.916*	0.916	0.749	*0.795*	0.858	*0.689*	*0.788*	0.845	0.702	*0.717*	0.803	*0.629*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 11 Performance evaluation of all measures when, NF = 6000–the averaged results (K = 1–120; + 2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	ACC	PRE	REC	FM	GM	AMP	ACC	PRE	REC	FM	GM	AMP
1	Euclidean	0.627	0.724	0.302	0.356	0.519	0.279	0.550	0.771	0.428	0.422	0.590	0.385
2	Cosine	*0.899*	0.902	0.717	*0.769*	0.837	*0.651*	0.766	0.810	0.698	*0.717*	0.798	0.616
3	Jaccard	0.867	0.813	0.555	0.609	0.734	0.493	*0.784*	0.858	0.684	0.700	0.791	*0.621*
4	Bhattacharya	0.888	0.867	0.683	0.693	0.818	0.590	0.533	0.689	0.525	0.433	0.666	0.406
5	kullback–Leibler	0.503	0.163	0.128	0.089	0.335	0.128	0.218	0.078	0.248	0.090	0.431	0.225
6	Manhattan	0.530	0.401	0.164	0.148	0.375	0.160	0.435	0.688	0.300	0.230	0.479	0.290
7	PDSM	*0.912*	0.899	0.709	*0.754*	0.834	*0.639*	*0.802*	0.854	0.717	*0.734*	0.814	*0.647*
8	STB-SM	*0.916*	0.916	0.750	*0.787*	0.854	*0.680*	*0.792*	0.844	0.707	*0.721*	0.807	*0.634*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 12 Performance evaluation of all measures when, NF = the whole size (Reuters = 18308, web-kb = 33025 features)—the averaged results (K = 1–120; +2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	ACC	PRE	REC	FM	GM	AMP	ACC	PRE	REC	FM	GM	AMP
1	Euclidean	0.615	0.716	0.294	0.344	0.510	0.271	0.550	0.768	0.428	0.422	0.591	0.384
2	Cosine	*0.897*	0.903	0.716	*0.767*	0.836	*0.650*	0.766	0.803	0.670	*0.715*	0.799	0.614
3	Jaccard	0.865	0.813	0.550	0.601	0.730	0.488	*0.786*	0.859	0.684	0.694	0.792	*0.619*
4	Bhattacharya	0.888	0.867	0.683	0.693	0.818	0.590	0.534	0.689	0.526	0.434	0.667	0.406
5	kullback–Leibler	0.503	0.164	0.128	0.089	0.335	0.128	0.134	0.079	0.248	0.090	0.431	0.250
6	Manhattan	0.527	0.376	0.162	0.144	0.373	0.158	0.429	0.669	0.294	0.220	0.474	0.284
7	PDSM	*0.909*	0.899	0.700	*0.745*	0.827	*0.631*	*0.801*	0.854	0.714	*0.727*	0.812	*0.642*
8	STB-SM	*0.912*	0.913	0.739	*0.783*	0.851	*0.676*	*0.791*	0.841	0.706	*0.715*	0.806	*0.630*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 13 Performance evaluation of all measures when taken the average of averaged results,—average results (K = 1–120; +2)

No	Dataset	Reuters						Web-KB					
	Similarity/criterion	ACC	PRE	REC	FM	GM	AMP	ACC	PRE	REC	FM	GM	AMP
1	Euclidean	0.732	0.668	0.406	0.453	0.610	0.353	0.606	0.746	0.496	0.505	0.645	0.435
2	Cosine	*0.864*	0.737	0.615	*0.645*	0.769	*0.536*	*0.738*	0.767	0.669	*0.688*	0.776	*0.581*
3	Jaccard	0.790	0.661	0.478	0.515	0.675	0.411	0.721	0.817	0.613	0.623	0.737	0.550
4	Bhattacharya	0.837	0.670	0.557	0.558	0.726	0.475	0.510	0.644	0.503	0.408	0.648	0.395
5	kullback–Leibler	0.555	0.411	0.190	0.174	0.405	0.195	0.370	0.338	0.292	0.193	0.470	0.278
6	Manhattan	0.666	0.537	0.328	0.343	0.533	0.284	0.531	0.744	0.409	0.388	0.572	0.374
7	PDSM	*0.869*	0.725	0.619	*0.632*	0.772	*0.527*	*0.768*	0.821	0.681	*0.700*	0.787	*0.609*
8	STB-SM	*0.880*	0.747	0.646	*0.666*	0.790	*0.564*	*0.764*	0.780	0.687	*0.702*	0.791	*0.607*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Jaccard. It is worth mentioning that from Table 6 , 7, 8, 9, 10, 11, 12, results have been almost the same. In other words, results have been noted in stable condition.

Finally in Table 13, when the average of averaged results has been taken, it is clear that, for both Reuters and Web-KB, STB-SM, PDSM, and Cosine have been the best measures for all criterions. Thus, in conclusions, the top performer measures, when the average results have been taken, are STB-SM, PDSM, and Cosine.

Clustering results

In this subsection, we have evaluated and compared the impact of all considered similarity measures on the behavior of the K-means clustering algorithm. Fixing K on (4 and 8) and using the presented-above evaluation metrics for clustering (see Table 4), the experiments have been conducted on both datasets (Reuters and Web-KB) to experimentally identify and distinguish which measure would be the best fit for K-means. Either positively or negatively, the experiments could clearly signify the selection of similarity measure impact on clustering quality. All features of both datasets were considered when the clustering process has been running (Reuters = 18,308, Web-KB = 33,025 features). As stated earlier, we have used two internal metrics, and three external metrics for clustering evaluation of K-means under the umbrella of all considered similarity measures. As for the stopping condition, the K-means was allowed to stop after running (50) iterations to obtain the best performance, or alternatively, the algorithm reached the stability situation for two consecutive cycles. The stability situation is the case in which K-means clusters were recorded stable (unchanged) for two consecutive cycles. Centroids of clusters were chosen randomly in each iteration. We have used the voting technique (see Table 20) to decide the best fit similarity measure using which the performance of K-means has been noted to be the highest. According to the results drawn in Tables (14, 15, 16, 17, 18), STB-SM, followed by PDSM and Euclidean, has been the best fit in this study. The bolded values in Tables (14, 15, 16, 17, 18) signify the best values each measure had achieved on the corresponding metric.

Table 14 External Metric-Purity (mostly known as "Accuracy")-K-means performance

Similarity measure/metric	K = 4		K = 8	
	Reuters–18308 features	Web-KB-33025 features	Reuters–18308 features	Web-KB-33025 features
Euclidean	*0.6745546742946301*	*0.4420100023815194*	*0.6651930828240801*	*0.5363181709930936*
Cosine	*0.6300871148095176*	*0.5651345558466302*	0.5161877519178261	*0.5710883543700881*
Jaccard	0.5418021063580809	0.4060490592998333	0.5631257313743336	0.42462491069302216
Bhattacharya	0.6602522428812898	0.550845439390331	*0.6573917565986218*	0.3908073350797809
kullback–Leibler	0.5103367572487323	0.39104548702071923	0.5103367572487323	0.39175994284353416
Manhattan	0.528799895982317	0.3912836389616575	0.5342608243401378	0.40128602048106693
PDSM	*0.6628526849564426*	0.4165277447011193	*0.6329476010921856*	0.40533460347701833
STB-SM	0.626706540111819	*0.6110978804477256*	0.6059030035105968	*0.571802810192903*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 15 External metric—completeness-K-means performance

Similarity measure/Metric	K = 4		K = 8	
	Reuters–18308 features	Web-KB-33025 features	Reuters–18308 features	Web-KB -33025 features
Euclidean	*0.199511647995826*	0.0673834489462676	0.179814979894203	*0.15452274170402275*
Cosine	0.176068659196536	*0.2225985559938000*	0.049343505745102	*0.1665642553582776*
Jaccard	0.055007933247357	0.0318290213093328	0.050929724394223	0.03093748911398607
Bhattacharya	0.161670093558948	*0.3071187147856372*	*0.212872136220606*	*0.04952824171446496*
kullback–Leibler	0.138798782038987	0.0661952142325757	0.08238615904440288	0.09274073186745294
Manhattan	0.140585609897670	0.0776936270894379	0.162146390404297	0.2204324456530784
PDSM	*0.224510552205581*	0.0377046283518796	*0.199091523319667*	0.07172116609953971
STB-SM	*0.267581280659119*	*0.2935685999112778*	*0.220934861375524*	*0.17402803377334777*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 16 External metric—Rand index-K-means performance

Similarity measure/Metric	K = 4		K = 8	
	Reuters–18308 features	Web-KB-33025 features	Reuters–18308 features	Web-KB-33025 features
Euclidean	0.18579386266996	0.07259267580718086	*0.134417241084232*	*0.13997725114650594*
Cosine	*0.13679798891326*	*0.19584148736739818*	0.038271439259564	*0.1388214135942175*
Jaccard	0.03156196525375	0.03337323105349306	0.03325438109622	0.02868306344839796
Bhattacharya	0.10437193466823	*0.21881970871431122*	0.037753839600227	− 0.00837183862883132
kullback–Leibler	− 6.969300641e-05	− 0.0003339640498897	− 0.00089845850765	− 0.00025681169419640
Manhattan	− 0.0456795771624	0.05064565361647291	− 0.02973126512374	0.003861173313610305
PDSM	*0.10879494053512*	0.00359089370980009	*0.116230029219350*	− 0.00999313066207844
STB-SM	*0.17377955245469*	*0.23459410623220853*	*0.108893811930251*	*0.17012239790902425*

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 17 Internal Metric-Calinski-Harabasz Index–K-means performance

Similarity measure/metric	K = 4		K = 8	
	Reuters–18308 features	Web-KB-33025 features	Reuters–18308 features	Web-KB-33025 features
Euclidean	282.616242361247	733.8545918925699	175.69520099100788	376.55783811163883
Cosine	158.059649581219	41.1205304663345	*36.67050378019272*	22.764647326363253
Jaccard	*10.76166549332125*	*5.452237604387431*	*12.590273044103254*	*4.818955513623545*
Bhattacharya	142.1019959257336	18.169867898545604	125.70926440159229	7.864068171394914
kullback–Leibler	*0.6047659734858815*	*0.16973494233360648*	*0.3996948922364028*	*0.21630702269179736*
Manhattan	222.98925272777933	687.1305760147841	96.46528179432781	338.1631503564861
PDSM	132.7979522566384	*6.947310633285274*	87.3385616551679	*7.265095316386779*
STB-SM	*127.69695325126817*	37.83093841884232	92.69694747153109	36.67724243309815

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Table 18 Internal metric—Davies-Bouldin Index–K-means performance

Similarity measure/metric	K = 4		K = 8	
	Reuters–18308 features	Web-KB-33025 features	Reuters–18308 features	Web-KB-33025 features
Euclidean	4.186722232186011	*4.485270517901569*	4.138785792328177	*4.414646698351782*
Cosine	4.974207600701769	5.184748331594617	5.120474137629764	6.511148800037687
Jaccard	13.633300152033158	15.865256973163703	14.42610495247111	15.858516305005779
Bhattacharya	4.597427791367133	4.893369736545789	*3.280451019807807*	5.177469928065736
kullback–Leibler	*1.872529736499688*	*2.8045629002126167*	*2.266827595447801*	*2.1725394033652123*
Manhattan	*2.627533355357464*	*2.9998709693117767*	*2.1237224749990613*	*1.6587667141935847*
PDSM	*3.7833377396816443*	7.519681593999572	5.179788734900043	5.386991923494522
STB-SM	5.764560759902967	5.85720075727297	5.391258478217752	5.654144650340377

Italic values indicate the highest values that top measures achieved for corresponding evaluation metrics

Discussion

The discussion revolves around two key points. First, the measure performance stability over both datasets. Second, in which the number of features each measure has performed the best in terms of accuracy (ACC), f-measure (FM), and average mean precisions (AMP).

Classification- performance stability

Based on the results given in Tables (5, 6, 7, 8, 9, 10, 11, 12, 13), 19 observed that the most stable measures on both datasets based on the points every measure has achieved on each number of features. It is concluded that the more points each measure achieves, the more stable it is. In the next Table, R and W indicate Reuters and Web-KB datasets respectively.

Table 19 shows that the most stable measures were STB-SM PDSM and Cosine with 48, 45, and 45 points respectively. While PDSM was more stable on web-kb than Cosine, Cosine was more stable on Reuters than PDSM. However, while Table 19 gives the stable measures according to its giving one plus for each measure when a measure has been superior in terms of specific criterions (out of three criterions, namely, ACC, FM, and AMP) on each dataset, the real numbers in results drawn in Tables (5, 6, 7, 8, 9, 10, 11, 12, 13) also showed indisputably that the top performer measures are also STB-SM, PDSM, Cosine, and Jaccard. It can also be deduced from the results that, unlike Reuters upon which measures have unstable performance, all measures on web-kb have almost stable performance, chiefly the top performers. On Reuters, the competition was held between STB-SM, PDSM, and Cosine. Moreover, from Table 13, based on points recorded on each NF, it can be concluded that STB-SM, PDSM, Cosine, and Jaccard can be used effectively for low, middle and high dimensional datasets as these measures performed well on each NF value. Euclidean and Manhattan also performed well on low dimensional datasets (NF in [10–200] features). Bhattacharya was observed to behave well on middle and high dimensional datasets (NF in [200-N] features).

Strictly speaking, the highest performance over both datasets was seen for both STB-SM and PDSM as both measures showed almost stable and close performance on all values of NF when the poor performance was seen for kullback–Leibler, Manhattan and Euclidean chiefly on high dimensional datasets.

Table 19 Measure stability points

NF/Measure	Euclidean		Cosine		Jaccard		Bhattacharya		kullback–Leibler		Manhattan		PSDM		STB-SM	
	R	W	R	W	R	W	R	W	R	W	R	W	R	W	R	W
10	2		2	3					2	1				3	3	3
50			3	3									3	3	3	3
100			3	3									3	3	3	3
200			3	3									3	3	3	3
350			3	3									3	3	3	3
3000			3	3									3	3	3	3
6000			3	1		2							3	3	3	3
The whole features			3	1		2							3	3	3	3
Sum	2		23	22		4			2	1			21	24	24	24
Stable points	2		45		4				3				45		48	

Italic values indicate the most stable similarity measures

Classification-performance climax

Second, in which the number of features, the similarity measure performance was at its climax in terms of accuracy, f-measure, and average mean precisions.

Performance analysis-reuters

The next Figs. (1, 2, 3) hold the map of the criterions movements (results were averaged) for all measures over several NF values.

Figure 1-a depicts that Manhattan followed by Euclidean and kullback–Leibler did not have a stable accuracy and performed the worst as NF grew. In contrast, STB-SM, PDSM, and Bhattacharya had the most stable performance. The Cosine and Jaccard showed a punctuated accuracy while NF grew from 10 to 3000, and then started to be slightly declined as NF grew. Figure 1b draws the top competitors which were STB-SM, PDSM, and Cosine with STB-SM being superior.

Figure 2a shows that Euclidean and Manhattan had a higher FM when NF was in the range of [10–100]. However, their performance started deteriorating as NF grew. Like accuracy movement, STB-SM, PDSM, Cosine, Bhattacharya, and Jaccard yielded an almost stable FM while NF grew from 10 to N, with STB-SM and PDSM being the best, and Bhattacharya bettered Jaccard. Bhattacharya's performance declined slightly for the favor of STB-SM, PDSM, and Cosine, though. Finally, kullback–Leibler was shown to perform the worst. Figure 2b draws the top competitors which were STB-SM, Cosine, and PDSM with STB-SM being superior over all of them, and Cosine superior over PDSM.

Finally, from Fig. 3a, it is noted that Euclidean followed by Manhattan and kullback–Leibler had the worst performance in terms of AMP. On the other hand, STB-SM, PDSM, Cosine, and Bhattacharya drew the best performance. The Jaccard measure was seen to represent a middle ground between those of the highest AMP and those of lowest AMP. It is worth noting that all measures of the highest performance were seen effective from NF = 10 till NF = 3000, and their effectiveness keeps improving on all features.

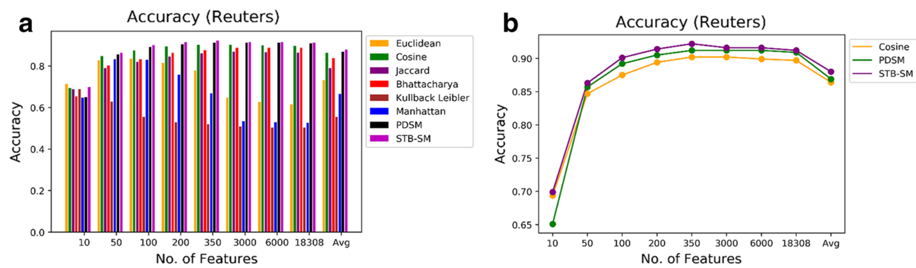

Fig. 1 **a** Accuracy over all Measures on all NF values– Average results (K = 1–120; + 2)—Reuters. **b** Accuracy over competitive measures on all NF values– Average results (K = 1–120; +2)–Reuters

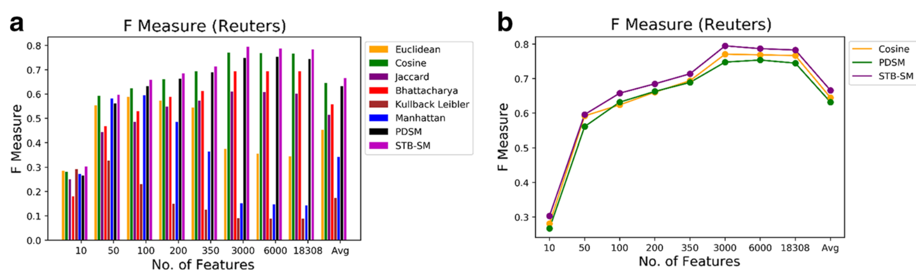

Fig. 2 **a** F-measure over all Measures on all NF values– Average results (K = 1–120; + 2)—Reuters. **b** F-measure over competitive measures on all NF values–Average results (K = 1–120; +2)–Reuters

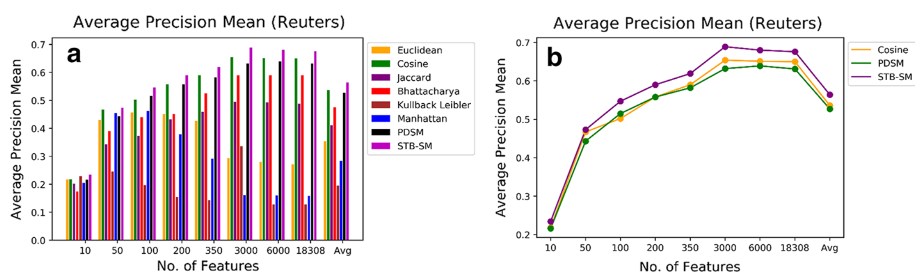

Fig. 3 **a** AMP over all Measures on all NF values– Average results (K = 1–120; +2)–Reuters. **b** AMP over competitive measures on all NF values– Average results (K = 1–120; +2)–Reuters

However, as NF surpassed 3000 to reach 6000 or bigger, their performance started to lower slightly as shown in Fig. 3b.

Performance analysis-web-Kb

The next Figs. (4, 5, 6) hold the map of the criterions movements (results were averaged) for all measures over several NF values.

From Fig. 4a, it can be seen that Manhattan followed by kullback–Leibler had got an almost stable accuracy albeit the fact that they performed poorly as NF grew. STB-SM, PDSM, Cosine, and Jaccard showed a clear stable higher accuracy while NF grew from 10 to all features, with STB-SM and PDSM being highly superior. While STB-SM outweighed PDSM when NF is in the range [50–3000], PDSM outweighed STB_SM from 6000 to all features as shown in Fig. 4b both measures intersected at 3000 features, though. In addition, on average, STB-SM was still taking the lead. Manhattan and Euclidean had a close performance from each other when NF was in the range [10–200].

However, as NF grew, Euclidean outperformed Manhattan, while seen more closely to Bhattacharya when NF in [350–33,025].

From Fig. 5a, it is shown that Manhattan and kullback–Leibler had the worst performance albeit the fact that kullback–Leibler was seen closer to Cosine when NF = 10. It was a rare case, though. Similarly to Fig. 4b, PDSM, Fig. 5b exhibits that PDSM, STB-SM and Cosine were the best performance with PDSM and STB-SM being fiercely rivals. On the other hand, Jaccard, and Euclidean outperformed Manhattan and kullback–Leibler and Bhattacharya as NF was in the range [50–6000]. However, as NF grew bigger, Bhattacharya started to show a gradually-increasing performance over Euclidean.

Finally, from Fig. 6a, it is clear that Manhattan, Bhattacharya followed by kullback–Leibler had the worst performance in terms of AMP albeit the fact that Manhattan had higher AMP when NF was in [10-100]. When NF was in the range [50-200], Bhattacharya followed by kullback–Leibler were seen to have the worst AMP values. As NF grew, Bhattacharya started to have better performance over Manhattan, though. Similarly, Euclidean outweighed Jaccard as NF was in the range [10–350]. However, as NF grew, Jaccard behaved better than Euclidean. Similar to Fig. 5b, Fig. 6b exhibited that PDSM, STB-SM and Cosine had the best performance with PDSM and STB-SM being highly rivals.

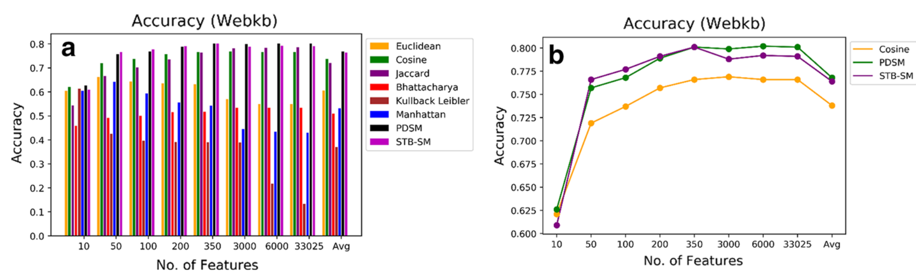

Fig. 4 **a** Accuracy over all Measures on all NF values– Average results (K = 1–120; + 2) – Web-KB. **b** Accuracy over competitive measures on all NF values– Average results (K = 1–120; + 2)–Web-KB

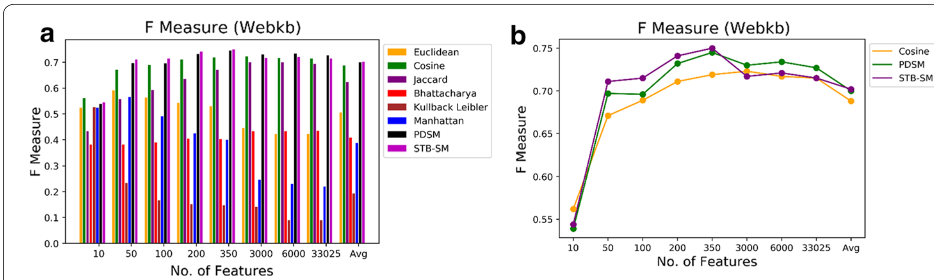

Fig. 5 **a** F-measure over all Measures on all NF values–average results (K = 1–120; +2)–Web-KB. B F-measure over competitive measures on all NF values– Average results (K = 1–120; +2)–Web-KBB F-measure over competitive measures on all NF values– Average results (K = 1–120; +2)–Web-KB

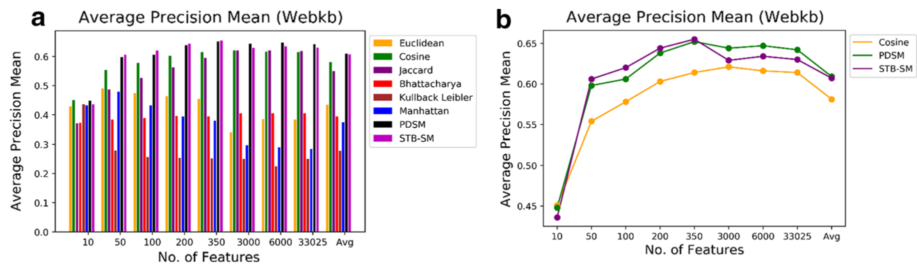

Fig. 6 **a** AMP over all Measures on all NF values–average results (K = 1–120; +2)–Web-KB. B AMP over competitive measures on all NF values–Average results (K = 1–120; +2)–Web-KB

Classification-execution time analysis

Finally, the time consumed by each measure on each dataset over each NF was accumulated and averaged to show which one runs the fastest and which one runs the slowest. A certain measure could give higher accuracy and desired performance while it ran slower compared with others and vice versa. The next Figures map the time taken by each measure to produce the results. According to execution time drawn in Figs. 7, 8, it is abundantly clear that all measures share one fact: the execution time is growing steadily as NF increases, PDSM in particular. It is worth mentioning that time was calculated as the similarity measure run on all evaluation metrics (six metrics) of classification.

Figure 7 clearly shows that Bhattacharyya and Manhattan were the fastest similarity measures with Manhattan being much faster on all features. However, this came at the expense of the drawn-above results by both measures as it occupied the second and third-worst measures after kullback–Leibler. Euclidean had been observed to be the middle ground in terms of speed between the first group (Bhattacharyya, Manhattan) and the second group (PDSM, kullback–Leibler, Jaccard, Jaccard, Cosine, and STB-SM). Surprisingly, when all features addressed, Manhattan was the fastest measure and PDSM was the slowest measure as it took roughly **1493.85** min on Reuters when NF = all features. On the other hand, closer to Bhattacharyya, Euclidean recorded worse results when compared with Cosine, Jaccard, STB-SM, and PDSM. Jaccard, on the other hand, was faster than Cosine and STB-SM, and Cosine was slower than STB-SM. In order, PDSM, Jaccard followed by Cosine, and STB-SM were all observed to be the slower measures comparing with the first group, though. In fact, PDSM has been seen to be the slowest measure.

Similarly to Fig. 7, 8 clearly shows that Bhattacharyya and Manhattan were also the fastest similarity measure with Manhattan being subtly faster on all features. However, similar to Reuters, this came at the expense of the drawn-above results by both measures as it occupied the second and third-worst measure after kullback–Leibler. Euclidean, on the other hand, had been observed to be the fastest metric when all features considered, and PDSM was seen to be the slowest ever as it took almost *1001.067* min on all features–Web-KB. However, like Bhattacharyya, Euclidean recorded to have worse results compared with Cosine, Jaccard, PDSM, and STB-SM. Meanwhile, Cosine was faster than Jaccard and STB-SM in all NF cases except for the case in which all features were addressed. In this case, Jaccard was faster than both Cosine and STB-SM. In general, in order, PDSM, kullback–Leibler, Jaccard, STB-SM, and Cosine were the slowest measures with PDSM being the slowest ever.

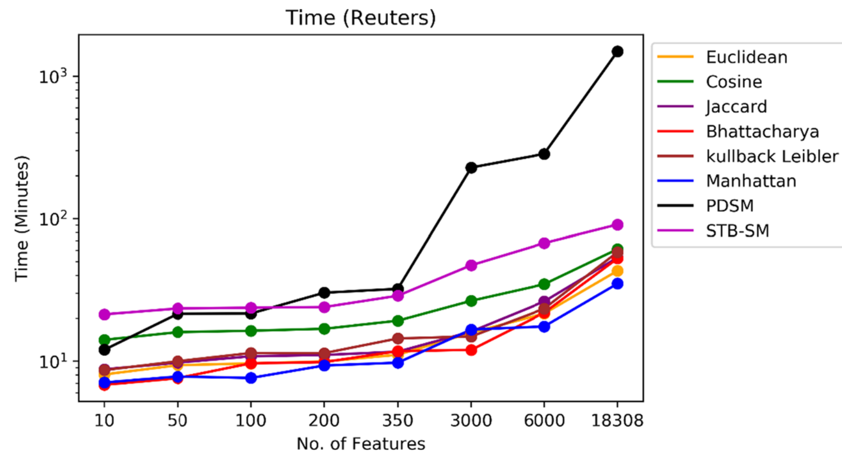

Fig. 7 Execution Time–Reuters

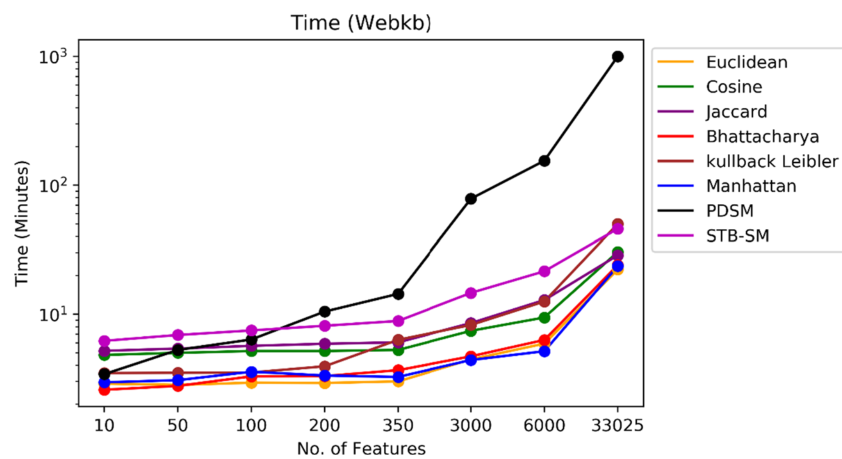

Fig. 8 Execution Time–Web-KB

Clustering analysis

Based on the results drawn in Tables 14, 15, 16, 17, 18, the analysis is done briefly in Table (20) across counting the points each similarity measure had achieved on each metric. The point is counted for measure if it is being bolded as higher value, in Tables 14, 15, 16, 17, 18. The total number of points are 20 points as we have two datasets and five metrics on two values of clustering variable (K = 4, K = 8). In each Table, there has been four points each measure could achieve based on the drawn results. For example Euclidean in Table 4 got 4 points as its results are spotted as top values for purity metric on both datasets on both K (4 and 8). The next Table draws the points recorded for each measure on each metric (Tables 14, 15, 16, 17, 18), and the points in total and rank as well. The bolded values in Table 22 suggest the highest values in Table 20, which reflect the optimality of each measures on the corresponding metric.

In general STB-SM behaves better than PDSM on web-kb clustering chiefly when k grows. That means STB-SM could work optimally on big data, and STB-SM enjoys the scalability properties. The scalability case is the case in which dataset grow larger and larger in terms of data, for example. Ironically, unlike classification, PDSM works better than STB-SM on Reuters, though. Briefly, according to Tables (14, 15, 16, 17, 18), the order of top performer measures on Reuters was: Euclidean PDSM, Cosine and STB-SM, On the other extreme, the order of top performer measures on Web-KB was: STB-SM, Cosine, Euclidean and PDSM. Strictly speaking, the competition process is fiercely held between STB-SM, Euclidean, Cosine and PDSM with STB-SM being maximally superior. In other words, according to the real numbers drawn in results (Tables 14, 15, 16, 17, 18), STB_SM has better values than all measures in most cases. For example, in purity web-kb, completeness and Rand Index of both datsets, the result values of STB-SM are much bigger than those values of other measures. Thus, it can be confidently said that STB-SM outperformed all similarity measures significantly in most cases of clustering evaluation metrics.

Table 20 Rank of similarity measures based on clustering results

Measure/table	Purity	Completeness	Rand index	-Calinski-Harabasz Index	Davies-Bouldin Index	Point Total out of 20	Rank
Euclidean	*4*	2	2	0	2	10	2
Cosine	3	2	3	1	0	9	4
Jaccard	0	0	0	*4*	0	4	7
Bhattacharya	1	3	1	0	1	6	6
kullback–Leibler	0	0	0	*4*	*4*	8	5
Manhattan	0	0	0	0	*4*	4	7
PDSM	2	2	2	2	1	9	3
STB-SM	2	*4*	*4*	1	0	*11*	*1*

Italic values indicate the high-ranking similarity measure

Clustering–execution time analysis

Based on the time drawn in Tables (21, 22), PDSM was the slowest measure and Manhattan was the fastest measure. As given in Tables (14, 15, 16, 17, 18; 21, 22), our proposed STB-SM measure came as a compromised solution for both efficiency and effectiveness. It is worth referring that the clustering time has been calculated while K-mean run on nine evaluation metrics. However, in this work, we just used five metrics. So, this time (drawn time in Tables 21, 22) would be shorter (either slightly or significantly) than the expected time when the K-means is running on only these five metrics. That is because adding each metric in clustering often takes extra time, and consequently increase clustering time either slight or significantly. Nevertheless, this claim has not refuted or contradicted the final conclusion drawn in this paper on the speed or slowness of each similarity measure.

Table 21 Reuters–run time in (hour: minute: second)

K/measure	Euclidean	Cosine	Jaccard	Bhattacharya	kullback–Leibler	Manhattan	PDSM	STB-SM
K = 4	0:8:18	0:9:06	0:8:04	0:07:56	0:13:13	0:7:04	2:26:14	0:10:18
K = 8	0:15:03	0:16:37	0:15:00	0:15:05	0:25:04	0:12:59	4:44:16	0:41:58

Table 22 Web-KB – Run Time in (Hour: Minute: Second)

K/Measure	Euclidean	Cosine	Jaccard	Bhattacharya	kullback–Leibler	Manhattan	PDSM	STB-SM
K = 4	0:6:36	0:7:26	0:7:01	6:42.98	0:11:14	0:5:55	2:33:09	0:8:54
K = 8	0:12:09	0:13:50	0:12:40	0:12:22	0:21:08	0:10:41	6:21:26	0:17:10

The applicability of proposed measure (STB-SM) on big data environment

Since the advent of Internet, the size of textual information keeps growing because of the continuous evolution of information technologies. These technologies have allowed massive volumes of data to be exponentially increasing across the online contents like all kinds of webpages (academic, scientific, news, medical, etc.), blogs, social networking like Facebook and twitter, and Youtube. In daily basis, trillions of bytes of data are generated that 90% of data in the world was thought to be existed in last couple of years [34, 35]. Consequently, this fast growing of data volumes has led to a critical information retrieval problems. Among these problems is how to get the relevant document(s) of interest amid such gigantic volumes of textual data and information. To solve such problem, the clustering as a data mining technique come for analyzing these massive volumes of data which is called "Big Data". Without the clustering and classification, it is challenging to manage and discover the knowledge in the environment of big data. However, there have been difficulties for implementing clustering algorithms to big data as clustering algorithms accompanied with high computational costs and complexity. To make it worse, recently, emanation of big data (with all its characteristics including volumes, variety, velocity, variability and complexity) draws more difficulties to this issue which pushes more studies and research to find every possible way to improve clustering algorithms.

That lead us to the question of how to overcome this dilemma, and how to apply clustering algorithms to big data while obtaining the results in a reasonable time. One possible solution to improve the performance of clustering to get results of higher accuracy in reasonable time is to use the well-designed time-efficient similarity measure. In fact, the performance of clustering and classification is maximally dependent on the similarity measure in use as we have seen in this work in PDSM and STB-SM cases. Despite the fact that both measures are effective, PDSM is seen time-inefficient chiefly when used to clustering purpose. Unlike PDSM, STB-SM is time-efficient making it a promising measure for scalability of clustering.

Presently, similarity measure have been sought to mainly promote the accuracy of classification and clustering as well as the efficiency with the intended techniques like KNN classifier and k-means clustering algorithm. Therefore, in this work, we proposed a similarity measure which is thought to be capable of handling the big data analysis effectively and efficiently. Based on the results drawn for both classification and clustering in particular, we believe that our proposed measure (STB-SM) is promising to be an effective technique to process voluminous data in reasonable time with higher accuracy. When we applied STB-SM on all features of each dataset to perform clustering, STB-SM drew highly competitive results in a reasonable time comparing with all state of art. That means STB-SM enjoys it is being significantly effective and maximally efficient and would add a valued-contribution to the field of information retrieval (which is vital part of big data) in particular and machine learning in general. In fact, while designing STB-SM, our focus has been on drawing the measure that would help scale up and expedite clustering algorithm without sacrificing results quality. In doing so, the clustering process will enjoy flexibility and provide faster response time at the same time. In other words, with the proposed measure (STB-SM) being effective and efficient, the clustering for big data (including document clustering) can be efficaciously implemented to enhance the speed of search, precision, recall, search engines, and so on.

Conclusions and future work

Using the BoW model, KNN classifier, and K-means algorithm, in the context of text classification and clustering, this paper introduces a new similarity measure that is based on the set theory mechanism and is named the STB-SM. Besides the STB-SM, a comparative study has thoroughly been carried out on seven similarity measures using six classification criterions and five clustering metrics. The obtained results demonstrated that STB-SM similarity measure achieved almost the best performance on all classification and clustering criterions on both datasets (Reuters-21 or Web-KB). Moreover, to stress proposed measure superiority, it was imperative to utilize more than one performance criterion to effectively assess all similarity measures. In fact, it was difficult to determine which measure was the optimal one for any dataset and/evaluation criterion unless they are all evaluated against each other comprehensively. Because of that, each dataset displayed different characteristics when classification or clustering were performed on them. Nonetheless, from the obtained results, it can be concluded that STB-SM, PDSM, Cosine, and Jaccard showed superiority over other measures, and obtained the most stable performance trends on both datasets for all K values, compared to Euclidean, Manhattan, and kullback–Leibler measures with Manhattan and kullback–Leibler being noted to have the worst results. On the other extreme, Euclidean and Bhattacharya had a fluctuating performance which can be classified as a middle-ground between high performance and poor performance measures.

Additionally, using the K-means clustering algorithm, all similarity measures were involved in a fierce clustering competition. All similarity measures were individually used to evaluate K-means performance with respect to five evaluation metrics from which three metrics are external and the last two are internal metrics. The STB-SM, PDSM, and Euclidean were observed to be the top performers in terms of clustering. The STB-SM has outperformed Euclidean and PDSM in most stages of evaluation metrics. It worth mentioning that all these results of clustering were collected and analyzed for the case in which the number of clusters K is taken as number of actual classes in both datasets (4 and 8). Thus, in the follow-up work, to avoid and biasedness and get a deeper insight into clustering performance, an exhaustive analysis with several K values on different clustering algorithms will be carried out.

All these measures were rigorously examined with regard to their execution time when classification and clustering are run on either dataset. For classification, results has shown that some measures met the highest speed but at the expense of their overall performance, such as the Bhattacharyya, Manhattan, and Euclidean. On the other hand, and to confirm the fact that the trade-off is un-escapable, PDSM had been able to achieve better effectiveness results but again at the expense of its efficiency as this measure in particular was the slowest measure. Nevertheless, the STB-SM, Jaccard, and Cosine measures were a suitable compromised solution between the fastest measures (the Bhattacharyya, Manhattan, and Euclidean) and the slowest measure (the PDSM). They were not only faster than the PDSM but they were also closer to the speed of the fastest measures. On the other hand, for clustering, the PDSM was also the slowest measure and Manhattan was the fastest measure. As a compromised solution for both effectiveness and efficiency on both the classification and the clustering, our proposed measure the STB-SM has shown superiority with regard to clustering as well as classification. Finally, this work briefly described the applicability of the STB-SM to big data scenarios. In the future work, we plan to broaden the current work to involve more state-of-the-art measures such as that described in [3, 4]. Moreover, the behavior of all these measures will thoroughly be examined on different machine learning tasks such as text summarization [36] and plagiarism detection.

Abbreviations
IR: Information Retrieval; NF: Number of features; ACC: Accuracy; PRE: Precision; REC: Recall; FM: F measure; GM: G Measure; AMP: Average mean precision.

Acknowledgements
The authors would like to sincerely express their thanks to both Prof Dr. Loc Nguyen (Loc Nguyen's Academic Network) and Prof Dr. Abdelmoneim Artoli (King Saud University) for their constructive suggestions while doing this work. The authors would like to heartily extend their eternal thanks to both the Journal of Big Data Team (Editors, in particular) along with those respected anonymous reviewers for their valuable comments and directions that otherwise this work wouldn't have been improved.

Authors' contributions
Both authors have been the key contributors in conception and design, implementing the approach and analyzing results of all experiments, and the preparation, writing and revising the manuscript. Both authors read and approved the final manuscript.

Authors' information
Ali A. Amer is an assistant professor in Computer Science Department at Taiz University (YEMEN). He has been publishing many research papers in highly-ranked and top-tier journals as well as refereed International conferences. He has also acted as a reviewer for many top-venue platforms. Of these platforms, he has published in, and reviewed for: ACM Computing Surveys, IEEE Access, Computer in Human Behavior, Journal of Big Data (Springer), International Journal on Semantic Web and Information Systems, Universal Computer Science Journal, Journal of Supercomputing, Heliyon, and Journal of Evolutionary Intelligence, to name a few. His primary interest of research falls into: Information Systems, Database, Distributed and Parallel Database Systems, Data Integration, Data Mining, Network, and Information Retrieval.

Hassan Abdalla is an Associate Professor of Information Systems at the College of Technological Innovation since 2018. He holds a PhD in Information Systems from London, UK. Prior to joining Zayed University, Dr. Abdalla who is an Oracle Certified Professional has worked as an associate professor at the College of Computer and Information Sciences at King Saud University (KSU), Riyadh, Saudi Arabia. He also served there as a head of Quality Unit. Dr. Abdalla has published his research work in many reputable refereed international journals and conferences, he has also served as a reviewer for many top-ranked Journals. Dr. Abdalla's research interests include Distributed database systems, Information Retrieval, Knowledge Management and Enterprise Computing.

Funding
This research has been supported by Research Incentive Fund (RIF) Grant Activity Code: R19093–Zayed University, UAE.

Availability of data and materials
Datasets used in this work is publically available, and code is being uploaded on GitHub (https://github.com/aliamer/Information-Retrieval---A-Set-Theory-Based-Similarity-Measure-for-Text-Clustering-and-Classification).

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Computer Science Department, Taiz University, Taiz, Yemen. [2] College of Technological Innovation, Zayed University, P.O. Box 144534, Abu Dhabi, UAE.

Appendix

Appendix 1

The code is now publically available on GitHub.

Appendix 2

In order for readers/researchers to absorb the idea of this work, we just provide a sample for accuracy Tables of STB-SM and PDSM similarity measures that would be similarly produced for all similarity measures when the code of this work is used.

See Tables 23, 24

Table 23 STB-SM Accuracy when NF = 6000 on both datasets

No. of features	Metric	Measure	k	Reuters-8	Web-Kb
6000	STB_SM	Accuracy	1	0.9255733448723497	0.7456418383518225
			3	0.9363911726525314	0.777337559429477
			5	0.9333621808740805	0.786053882725832
			7	0.9316313284292514	0.7868462757527733
			9	0.9294677628732151	0.7828843106180665
			11	0.927736910428386	0.7884310618066561
			13	0.9303331890956296	0.7939778129952456
			15	0.9264387710947641	0.7995245641838352
			17	0.9264387710947641	0.7995245641838352
			19	0.927736910428386	0.8042789223454834
			21	0.927736910428386	0.805863708399366
			23	0.9255733448723497	0.8082408874801902
			25	0.9242752055387278	0.7995245641838352
			27	0.9273041973171787	0.803486529318542
			29	0.9242752055387278	0.8019017432646592
			31	0.9255733448723497	0.8066561014263075
			33	0.9242752055387278	0.8019017432646592
			35	0.9251406317611424	0.7995245641838352
			37	0.9247079186499351	0.8019017432646592
			39	0.9225443530938987	0.8019017432646592
			41	0.9212462137602769	0.803486529318542
			43	0.9225443530938987	0.8019017432646592
			45	0.9212462137602769	0.8026941362916006
			47	0.9208135006490696	0.8011093502377179
			49	0.9208135006490696	0.7971473851030111
			51	0.9186499350930333	0.8011093502377179
			53	0.9173517957594115	0.8003169572107766
			55	0.9177845088706188	0.8003169572107766
			57	0.9182172219818261	0.8011093502377179
			59	0.9160536564257897	0.7971473851030111
			61	0.9151882302033751	0.7971473851030111
			63	0.9130246646473388	0.794770206022187
			65	0.9138900908697534	0.7955625990491284
			67	0.9134573777585461	0.7923930269413629
			69	0.9134573777585461	0.7908082408874801
			71	0.9130246646473388	0.7908082408874801
			73	0.9121592384249243	0.7876386687797148

Table 23 (continued)

No. of features	Metric	Measure	k	Reuters-8	Web-Kb
			75	0.9112938122025097	0.7884310618066561
			77	0.911726525313717	0.7908082408874801
			79	0.911726525313717	0.7923930269413629
			81	0.9086975335352662	0.7900158478605388
			83	0.9095629597576806	0.7892234548335975
			85	0.9086975335352662	0.7892234548335975
			87	0.9086975335352662	0.7876386687797148
			89	0.906966681090437	0.7876386687797148
			91	0.9078321073128516	0.786053882725832
			93	0.9073993942016443	0.7844690966719493
			95	0.9065339679792298	0.786053882725832
			97	0.906966681090437	0.783676703645008
			99	0.9095629597576806	0.783676703645008
			101	0.9065339679792298	0.7844690966719493
			103	0.9061012548680225	0.7852614896988906
			105	0.9048031155344007	0.783676703645008
			107	0.9030722630895716	0.7844690966719493
			109	0.9043704024231934	0.7852614896988906
			111	0.9026395499783644	0.7876386687797148
			113	0.9022068368671571	0.7884310618066561
			115	0.9022068368671571	0.7828843106180665
			117	0.900475984422328	0.7797147385103012
			119	0.8991778450887062	0.7828843106180665
			Average	0.9163781912591951	0.792247754886423

Table 24 PDSM Accuracy when NF = 6000 on both datasets

No. of features	Metric	Measure	k	Reuters-8	Web-Kb
6000	PSDM	Accuracy	1	0.9381220250973604	0.7955625990491284
			3	0.938987451319775	0.8335974643423137
			5	0.9394201644309823	0.8248811410459588
			7	0.9376893119861531	0.8335974643423137
			9	0.9398528775421895	0.8280507131537242
			11	0.9376893119861531	0.8312202852614897
			13	0.9368238857637387	0.8280507131537242
			15	0.9350930333189096	0.8240887480190174
			17	0.9324967546516659	0.8264659270998416
			19	0.9303331890956296	0.8193343898573693
			21	0.9294677628732151	0.8225039619651348
			23	0.9268714842059714	0.8232963549920761
			25	0.926006057983557	0.820919175911252
			27	0.9268714842059714	0.8177496038034865
			29	0.926006057983557	0.8169572107765452
			31	0.9264387710947641	0.8137876386687797
			33	0.922977066205106	0.812202852614897
			35	0.9199480744266552	0.8090332805071315
			37	0.9186499350930333	0.8050713153724247
			39	0.9169190826482042	0.8082408874801902
			41	0.9186499350930333	0.8066561014263075
			43	0.9160536564257897	0.8042789223454834
			45	0.9138900908697534	0.8050713153724247
			47	0.9156209433145824	0.8019017432646592
			49	0.9169190826482042	0.8019017432646592
			51	0.9156209433145824	0.8026941362916006
			53	0.9125919515361316	0.7979397781299524
			55	0.9104283859800952	0.7987321711568938
			57	0.9091302466464734	0.7987321711568938
			59	0.9095629597576806	0.7979397781299524
			61	0.9099956728688879	0.7971473851030111
			63	0.9086975335352662	0.7987321711568938
			65	0.9086975335352662	0.7963549920760697
			67	0.9086975335352662	0.7963549920760697
			69	0.9073993942016443	0.7955625990491284

Table 24 (continued)

No. of features	Metric	Measure	k	Reuters-8	Web-Kb
			71	0.9052358286456079	0.7955625990491284
			73	0.9039376893119861	0.794770206022187
			75	0.9026395499783644	0.7979397781299524
			77	0.9026395499783644	0.794770206022187
			79	0.9022068368671571	0.7931854199683043
			81	0.900475984422328	0.7955625990491284
			83	0.900475984422328	0.7923930269413629
			85	0.900475984422328	0.7923930269413629
			87	0.8996105581999134	0.7916006339144216
			89	0.9000432713111207	0.7916006339144216
			91	0.8974469926438771	0.7876386687797148
			93	0.8978797057550844	0.7884310618066561
			95	0.8970142795326699	0.7876386687797148
			97	0.8961488533102553	0.7876386687797148
			99	0.8965815664214626	0.7868462757527733
			101	0.8948507139766335	0.7884310618066561
			103	0.8970142795326699	0.7900158478605388
			105	0.8952834270878408	0.7868462757527733
			107	0.8952834270878408	0.7876386687797148
			109	0.8935525746430116	0.7852614896988906
			111	0.8926871484205972	0.786053882725832
			113	0.8922544353093899	0.7844690966719493
			115	0.8909562959757681	0.7844690966719493
			117	0.889225443530939	0.783676703645008
			119	0.8887927304197317	0.7797147385103012
			Average	0.9120222126063754	0.8021526677231899

Received: 24 April 2020 Accepted: 14 August 2020
Published online: 14 September 2020

References

1. Alvarez, J.E. and H. Bast, A review of word embedding and document similarity algorithms applied to academic text. Bachelor thesis, 2017.
2. Oghbaie M, Zanjireh MM. Pairwise document similarity measure based on present term set. J Big Data. 2018;5(1):52.
3. Sohangir S, Wang D. Improved sqrt-Cosine similarity measurement. J Big Data. 2017;4(1):25.
4. Lin Y-S, Jiang J-Y, Lee S-J. A similarity measure for text classification and clustering. IEEE Trans Knowl Data Eng. 2013;26(7):1575–90.

5. Xu S. Bayesian Naïve Bayes classifiers to text classification. J Inform Sci. 2018;44(1):48–59.
6. Sheydaei N, Saraee M, Shahgholian A. A novel feature selection method for text classification using association rules and clustering. J Inform Sci. 2015;41(1):3–15.
7. Subhashini R, Kumar VJ. Evaluating the performance of similarity measures used in document clustering and information retrieval. In: 1st Int Conf integrated intelligent computing, Bangalore, 2010, p. 27–31. https://doi.org/10.1109/iciic.20https://doi.org/10.42.
8. Amer AA. On K-means clustering-based approach for DDBSs design. J Big Data. 2020;7(1):1–31.
9. Amer AA, Mohamed MH, Asri K. ASGOP: An aggregated similarity-based greedy-oriented approach for relational DDBSs design. Heliyon. 2020;6(1):e03172.
10. Nguyen L, Amer AA. Advanced cosine measures for collaborative filtering. Adapt Personalization (ADP). 2019;1:21–41.
11. Shahmirzadi O, Lugowski A, Younge K. Text similarity in vector space models: a comparative study. In 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA). 2019. IEEE.
12. Strehl A, Ghosh J, Mooney R. Impact of similarity measures on web-page clustering. In Workshop on artificial intelligence for web search (AAAI 2000). 2000.
13. White RW, Jose JM. A study of topic similarity measures. In Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. 2004.
14. Huang A. Similarity measures for text document clustering. In Proceedings of the sixth New Zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand. 2008.
15. Forsyth RS, Sharoff S. Document dissimilarity within and across languages: a benchmarking study. Literary Linguistic Comput. 2014;29(1):6–22.
16. Thompson VU, Panchev C, Oakes M. Performance evaluation of similarity measures on similar and dissimilar text retrieval. In 2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K). IEEE. 2015.
17. Fahad A, et al. A survey of clustering algorithms for big data: taxonomy and empirical analysis. IEEE Trans Emerg Topics Comput. 2014;2(3):267–79.
18. Aslam JA, Frost M. An information-theoretic measure for document similarity. In: Proc 26th SIGIR, Toronto. 2003. p. 449–50.
19. Zhao Y. R and data mining: examples and case studies. Cambridge: Academic Press; 2012.
20. Tata S, Patel JM. Estimating the selectivity of tf-idf based Cosine similarity predicates. ACM Sigmod Record. 2007;36(2):7–12.
21. Bhattacharyya A. On a measure of divergence between two statistical populations defined by their probability distributions. Bull Calcutta Math Soc. 1943;35:99–109.
22. Schoenharl TW, Madey G. Evaluation of measurement techniques for the validation of agent-based simulations against streaming data. In International Conference on Computational Science. 2008. Springer.
23. Kullback S, Leibler RA. On information and sufficiency. Ann Math Stat. 1951;22(1):79–86.
24. Kullback S. Information theory and statistics Wiley. New York, 1959.
25. Jaccard P. The distribution of the flora in the alpine zone. 1. New phytologist, 1912. 11(2): p. 37–50.
26. Turney PD, Pantel P. From frequency to meaning: vector space models of semantics. J Artificial Intell Res. 2010;37:141–88.
27. Al-Ghuribi SM, Alshomrani S. A simple study of webpage text classification algorithms for Arabic and English Languages. In 2013 International Conference on IT Convergence and Security (ICITCS). 2013. IEEE.
28. Patil DB, Dongre YV. A fuzzy approach for text mining. IJ Math Sci Comput. 2015;4:34–43.
29. Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. Inf Process Manage. 1988;24(5):513–23.
30. Jabalameli, M., A. Arman, and M. Nematbakhsh, Improving the efficiency of term weighting in set of dynamic documents. 2015. International Journal of Modern Education and Computer Science, 7, 42-47.
31. Aggarwal CC, Zhai C. A survey of text classification algorithms, in mining text data. Boston: Springer; 2012. p. 163–222.
32. Lakshmi R, Baskar S. DIC-DOC-K-means: dissimilarity-based Initial Centroid selection for DOCument clustering using K-means for improving the effectiveness of text document clustering. J Inform Sci. 2019;45(6):818–32.
33. Xu R, Wunsch D. Survey of clustering algorithms. IEEE Trans Neural Networks. 2005;16(3):645–78.
34. Khadija A. Almohsen, Huda Al-Jobori, "Recommender Systems in Light of Big Data", International Journal of Electrical and Computer Engineering (IJECE), Vol. 5, No. 6, December 2015, pp. 1553–1563, 2015.
35. Hoad TC, Zobel J. Methods for identifying versioned and plagiarized documents. JASIST. 2003;54:203–15.
36. Nagwani NK. Summarizing large text collection using topic modelling and clustering based on MapReduce framework. J Big Data. 2015;2:6.

Publisher's Note