J Inf Process Syst, Vol.16, No.2, pp.277~300, April 2020
https://doi.org/10.3745/JIPS.04.0164

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Combining Distributed Word Representation and Document Distance for Short Text Document Clustering

Supavit Kongwudhikunakorn[*] and Kitsana Waiyamai[*]

Abstract

This paper presents a method for clustering short text documents, such as news headlines, social media statuses, or instant messages. Due to the characteristics of these documents, which are usually short and sparse, an appropriate technique is required to discover hidden knowledge. The objective of this paper is to identify the combination of document representation, document distance, and document clustering that yields the best clustering quality. Document representations are expanded by external knowledge sources represented by a Distributed Representation. To cluster documents, a K-means partitioning-based clustering technique is applied, where the similarities of documents are measured by word mover's distance. To validate the effectiveness of the proposed method, experiments were conducted to compare the clustering quality against several leading methods. The proposed method produced clusters of documents that resulted in higher precision, recall, F1-score, and adjusted Rand index for both real-world and standard data sets. Furthermore, manual inspection of the clustering results was conducted to observe the efficacy of the proposed method. The topics of each document cluster are undoubtedly reflected by members in the cluster.

Keywords

Document Clustering, Document Distance, Short Text Documents, Short Text Document Clustering

1. Introduction

It is known that at present, short text documents are the main form of communication, especially for user-generated content in social media. As the use of social media extensively expands, the number of such documents rapidly increases as well. Short text documents are text documents that contain very few words, and most are data from social media. Examples include social media statuses, tweets from Twitter, news headlines, and product reviews. For example, Twitter imposes a limit of 140 characters for each tweet [1]. Valuable information is usually embodied in these documents. Extracting latent knowledge by analyzing and clustering these short text documents presents a very challenging text-mining task [2-7]. Clustering is a descriptive unsupervised data mining technique that groups data instances into clusters such that similar instances are placed together while unrelated instances are placed in different groups [8]. To attain the best results in short-text clustering, there are three main factors that must be considered: document representation, document similarity, and document clustering. Because this work addresses short text documents, sparsity is the main issue of concern [9]. Thus, the document representation and

Manuscript received April 18, 2018; first revision December 26, 2019; accepted February 4, 2020.

Corresponding Author: Kitsana Waiyamai (fengknw@ku.ac.th)

[*] Dept. of Computer Engineering, Faculty of Engineering, Kasetsart University, Bangkok, Thailand (supavit.kon@ku.th, fengknw@ku.ac.th)

document similarity are two keys to help mitigate the problems caused by document length. The clustering mechanism is also one of the factors that influence the clustering results. A suitable clustering mechanism for short texts should be scrutinized regarding its direct influence on clustering output, cluster shape, and clustering time. In this paper, the best and most appropriate combination of the three main factors will be proposed and explained in detail.

The document representations can be roughly grouped into two main types: statistical-based text representations and learning-based text representations. For statistical-based representations, the text representation is generated by using statistical weighting schemes from the relationship of words in the documents [10,11]. The frequencies of term appearances are used to determine the ratio of the frequency of specific words in documents to other words in the entire corpus of documents [12]. Although this text representation has been widely used for regular text documents, it does not work well for short texts due to the characteristics of texts, which lead to low probabilities of word co-occurrences [13]. Alternately, learning-based text representations increase the background knowledge of text documents by aggregating the text representations with contextual information. Doing so helps to alleviate the word sparsity issues in short texts. Numerous research works have addressed the issues and discussed solutions, such as using convolutional neural networks to learn and capture features [13-15], using Gaussian models to capture topics [1], performing short text expansion [16,17], and adding more contextual information to documents [18-21]. Furthermore, Mikolov et al. [22] introduced a Distributed Representation of Words in a vector space by constructing a representation via a deep learning text representation approach. A vector representing each vocabulary is generated after being trained on a large corpus of documents to incorporate some background knowledge of word contexts and relationships. In this work, the Distributed Representation of Words is adopted as a short document representation.

The clustering mechanism plays an important role in short text document clustering. There are many available document clustering techniques, such as affinity propagation, hierarchical clustering, density-based clustering, partitioning-based clustering, and topic modeling. To incorporate each clustering technique, advantages and disadvantages must be carefully considered. In this work, three types of clustering are studied and presented: density-based, hierarchical-based, and partitioning-based clustering [7,23-25]. To cluster short text documents, there are three aspects that must be considered: the clustering mechanism, clustering time, and document similarity function. For the clustering mechanism, because document representations are usually high-dimensional, density-based and hierarchical-based clustering do not handle this issue well [9,26,27]. Alternately, partitioning-based clustering usually handles this in a much better manner [9]. Furthermore, the time it takes to cluster data also must be considered. The time complexity of partitioning-based clustering is less than that of the other two, making this technique more desirable [7,24]. The similarity functions can be roughly grouped into two main types: string-based and knowledge-based similarity [28]. String-based similarity finds the document relationships by determining the character composition and string sequence [28]. Alternately, knowledge-based similarity finds the semantic similarity of the documents using external information [28]. Because we are dealing with short texts, the number of words in each document is limited, which makes semantic similarity determination difficult. Thus, knowledge-based similarity seems to be more suitable for determining the similarity of short texts. In this work, partitioning-based clustering with knowledge-based similarity is adopted as a short document clustering algorithm, which will be further explained in this work.

Because short texts are very short and sparse, the sparsity and high-dimensional representations of text documents entail the design of document clustering algorithms to efficiently cluster short text documents

[9]. In this paper, the combinations of text representation, similarity function, and document clustering mechanism are studied to attain the best clustering quality. Clustering results are not totally dependent on any one factor; all three mentioned factors are integrated to yield the combination that provides the best clustering quality. Experiments on combinations of different methods and data sets have been examined and conducted, and they are further reported in section 4. We found that the proposed method outperforms all other techniques. Distributed Representation of Words, a learning-based text representation, generates a vector for each vocabulary after being trained on a large document corpus by incorporating some background knowledge of word contexts and relationships based on the co-occurrences of words in documents. Each word in the text is transformed and represented as a vector of a certain dimension. Word mover's distance (WMD) [29], a knowledge-based document similarity function, measures the dissimilarity of two text documents, calculated by aggregating the minimum distances between pairs of words in the two documents. WMD is applied to measure the similarity between documents to determine the topic of the documents. To handle sparsity issues, this document representation and document similarity function preserve semantic relationships between vocabularies in vector space. After the words are transformed into document representations, the K-means-based clustering algorithm is applied to cluster these document vectors into groups and thus the text documents into different clusters with semantically closely related topics [30].

The remainder of this paper is organized as follows. First, we briefly describe core concepts of document representation, document similarity, and document clustering. Next, our proposed method is presented, followed by a description of conducted experiments, explanations, discussion, and results. Finally, the study's conclusions are presented in the last section.

2. Background

2.1 Document Representation

To perform clustering on text documents, an initial and important step is to transform these documents into document representations. For short text clustering, the main factors that must be considered are the number of words per document and the word context. Because the short text documents are very concise and contain few words per document, finding a suitable document representation is very important. Furthermore, words with the same context are often used interchangeably (such as synonyms and homonyms), which means that different words with the same context often appear in the document. Various document representation techniques have been proposed to represent the texts, such as bag-of-words, term frequency-inverse document frequency (TF-IDF) [31], one-hot vector, and Distributed Representation of Words [22,32]. To summarize, document representations can roughly be grouped into two types: statistical-based and learning-based.

A statistical-based text document representation is generated based on the frequency of the appearance of each word in the document. Among different statistical-based representation techniques, bag-of-words and TF-IDF are the most popular techniques. Bag-of-words represents text documents as vectors by the number of occurrences of each term in the document, where the length of the vector dimensions equals the number of vocabularies present in the data set. TF-IDF also represents text documents as vectors by the number of occurrences of each term in the document [10-12]. To improve the text representation from bag-of-words, the TF-IDF representation divides the frequency of each word (TF) by its inverse document frequency (IDF), the word's importance in the document [31]. As mentioned, short text documents contain very few words per document. TF-IDF relies heavily on word overlap, but in short text documents, as in this case, word overlap is rare. Thus, TF-IDF is unsuitable [13,33]. Using this technique could result in a very sparse document vector, leading to poor clustering results and high running time. Furthermore, topic-modeling methods also address the limitations of TF-IDF by learning latent topics in a document corpus with the word co-occurrences. Examples of such methods are latent semantic indexing (LSI) [9,34], probabilistic LSI (pLSI) [35], and latent Dirichlet allocation (LDA) [36]. However, these methods generally require at least a few hundred words for accurate determination, making them less suitable for short texts [1,21]. For word context, this representation only determines the word co-occurrences but does not capture any contextual information.

A learning-based text representation is generated by analyzing the relationship of different terms in the text document corpus and represented by vectors of a certain dimension. The key point of learning-based text representation is that, compared to statistical-based representation, learning-based representation adds contextual information to the text representation. Distributed Representation of Words (sometimes called word embeddings or continuous space representation of words), introduced by [22,32], is one of the renowned learning-based text representation techniques and is widely used when working on short text documents. The idea of this is to represent each word in the vocabulary as a vector of a certain dimension. This technique was first used by [37] and [38]. This technique later became more effective and popular, and subsequent works have been conducted to improve the methods of producing the representation by enhancing techniques and tools to handle larger vocabulary size [39,40]. Because this technique represents each word in the document as a dense vector of a certain dimension, there would be no problem with the sparsity from the concise number of vocabulary entries [1]. For word context, this representation undergoes the process of learning and analyzing word relationships that places vectors of similar words near each other and unrelated words far apart [39].

From the perspective of short text clustering, there are two major issues that must be considered: the small number of vocabularies, and the word context-based similar word detection [41]. Thus, a suitable document representation for short texts must be carefully selected. The appropriate document representation should be able to handle the issues related to text conciseness and word context similarities. As two groups of document representations have been thoroughly reviewed, learning-based text representation seems to be the suitable document representation for short texts and will be studied in the subsequent sections of this work.

2.2 Document Similarity

Document similarity plays a crucial role in clustering short text documents. It measures the closeness of one document to another to determine if they discuss the same topic [39]. Various text similarity approaches have been proposed, such as Euclidean distance, cosine similarity, n-gram, and Jaccard Similarity [28]. To select a suitable document similarity metric for short texts, the ability to capture semantic similarity is the key decision factor. Semantic similarity mainly describes the closeness of document context [28]. To summarize, document similarity can be roughly grouped into two main types: string-based and knowledge-based.

For string-based similarity, the measurement operates on the character composition and string sequence

[28]. To compare text documents, it measures the similarity or dissimilarity between them. There are two approaches for this measurement: character-based and term-based. Character-based views the similarity of documents, such as string matching, which compares the text documents character by character. Alternately, term-based views the similarity as the common terms between the text documents. For string-based similarity measures, n-grams, cosine similarity, and Euclidean distance are the most popular and widely used. N-grams compares a sub-sequence of *n* items from a given string by matching the overlapping characters of both strings. The n-grams similarity of documents is measured by dividing the overlapping n-grams by the maximum number of n-grams [42]. Cosine similarity measures the similarity of text document vectors as their cosine angles of an inner product space [28]. The significant property of using this similarity is the independence of document length [43]. Euclidean distance measures the similarity of text document vectors as the square root of the sum of squared differences between corresponding elements in the two vectors [28]. Furthermore, it is the default distance measure in many clustering algorithms [43]. To compare the string matching of the documents, this technique captures similar character/word sequences well. However, because this approach is straightforward string comparison, it does not capture any semantic similarity.

For knowledge-based similarity, the measurement identifies the semantic similarity of the words in documents using external information, such as semantic network and pre-trained word relationships. To compare text documents, it measures the relatedness between words in documents derived from the input external information [28]. Among different measures in knowledge-based similarity, WMD, introduced by Kusner et al. [29], is the interesting document similarity function. WMD, based on the idea of the Earth mover's distance [44,45], measures how far the words in one document must be "moved" to match the words in the other document. This work showed that, even if there are no words in common between the two documents, WMD can still capture the semantic similarity of their contexts well [29].

From the perspective of short text document clustering, most documents are short and contain only a few words. Considering data characteristics, these documents share no words in common, and synonyms and homonyms are the common issues in this type of text document. To efficiently find the semantic similarity of short text documents, external knowledge should be incorporated. For string-based similarity, because no background knowledge is used, it is good to compare documents on a word-by-word basis, but not by their semantic similarity contexts. Alternately, knowledge-based similarity has this knowledge of word relationships. Even if no words are in common between the compared documents, it can capture their semantic similarity. Thus, knowledge-based similarity seems to be an appropriate document similarity candidate and will be studied in a later section of this work.

2.3 Document Clustering

Clustering is an unsupervised data mining technique of dividing documents into groups based on the similarities of their features [8]. The goal is to put similar (or related) documents together in the same group such that they are similar to one another (high intra-cluster similarity) and different from documents in another group (low inter-cluster similarity) [46-48]. To cluster short text documents, various document clustering techniques have been proposed, such as affinity propagation, density-based clustering, hierarchical clustering, partitioning clustering, and topic modeling [1,2,7,9,24,25,41,49-52]. There are advantages and disadvantages of adopting each technique that must be considered. In this work, the main criteria for selecting the clustering techniques are an ability to handle large-dimension data, cluster shape, cluster completeness, and low time complexity [24,53-55]. As different clustering techniques have been addressed, only density-based clustering and partitioning clustering are included in the scope of this work.

Density-based clustering [56,57] is a clustering technique that groups data into clusters by the density of data points in the region. One of the most famous and widely used density-based clustering techniques, especially for text document clustering, is density based spatial clustering of applications with noise (DBSCAN) [50,57-60]. To perform clustering with DBSCAN, two main inputs are required: radius Epsilon (*Eps*) and minimum points (*MinPts*). The algorithm starts with an arbitrary point *p* and retrieves all neighbor points that are density-reachable from point *p* (within distance *Eps*) and have not been visited yet using the two input values. Regions of densely placed objects are considered as clusters and are separated by regions of low density or noise [27]. A cluster of densely connected points is formed when the number of neighbors of point *p* is greater than or equal to *MinPts*. The starting point *p* is marked as visited and, together with its neighbors, added to this cluster. This process recursively repeats for all neighbors of point *p*. Alternately, the point is marked as noise if the number of neighbors of point *p* is less than *MinPts*. When all density-reachable points are visited, the algorithm proceeds to the remaining unvisited points in the data [57,61]. Because text representations have large dimensions, along with the curse of dimensionality, DBSCAN does not work well for high-dimensional data [9,26,27]. For cluster shape, making DBSCAN very attractive, this technique can determine all cluster shapes, whether spherical or arbitrary [24,62]. For cluster completeness, DBSCAN will detect some data points as noise and not assign them to any cluster [24,63]. This technique has non-linear time complexity, which results in high running time [24].

Hierarchical-based clustering is a clustering technique that generates a nested sequence of partitions in a tree-like structure. One of the classical technique that deals with sparsity was presented by Karypis et al. [25] in 1999. They have presented a novel agglomerative hierarchical clustering algorithm named “CHAMELEON”. CHAMELEON is a novel clustering algorithm that overcame the limitations of existing agglomerative hierarchical clustering algorithm. It operates on a K-nearest neighbor sparse graph in which nodes represent data items, and weighted edges represent the similarities among data items. To form the clusters from the data set, CHAMELEON uses a special algorithm called “two-phase algorithm”. In the first phase, a graph partitioning algorithm is applied to the K-nearest neighbor graph to cluster data items into several small sub-clusters. In the second phase, an algorithm is applied to find the genuine clusters by repeatedly merge these sub-clusters. In this two-phase, CHAMELEON uses inter-connectivity and closeness to determine the similarity of the clusters.

Partitioning clustering [7] is a popular clustering technique for constructing a partition of a set of *N* data points into a set of *k* non-overlapping subsets (clusters) such that each data point lies in exactly one cluster [47,64]. One of the most popular and widely used distance-based partitioning clustering techniques, especially for text document clustering, is K-means clustering [2,7,9,13,15,30,51,54,65-67]. To perform clustering with the K-means clustering algorithm, a number of clusters *k* is required. The algorithm starts by randomly determining *k* arbitrary points as cluster centers (centroids) for *k* clusters. Then, each of the remaining points is assigned to the nearest cluster by calculating the distance between the points and the centroids of different clusters. Once all the points are completely assigned to the clusters, the cluster centroids are re-computed based on the intra-cluster similarity of cluster members so that the centroids better represent the center point of the cluster. Then, the point re-assignment process is repeated until all points are converged such that the difference in previous centroids and current centroids is less than a certain threshold value. For text representation, K-means has no problem in dealing with high-dimensional data [9]. For cluster shape, clustering with the K-means technique produces spherical cluster shapes [26]. For cluster completeness, clustering using the K-means technique ensures that all text documents get assigned to a cluster [49,55]. For time complexity, partitioning clustering takes linear time $O(nkl)$, where k is the number of clusters, n is the number of documents, and l is the number of iterations [7].

From the perspective of short text clustering, the ability to handle large-dimensional data, cluster shape, cluster completeness, and time complexity are the most important factors [9]. Thus, the suitable clustering technique must be carefully selected to cluster these short text documents into clusters of similar topics [9]. Because the text representations of short text documents usually contain large dimensions, as described earlier, density-based and hierarchical-based clustering techniques do not work well for this issue. Alternately, partitioning clustering has no problem in dealing with the high dimensionality of short text documents. For cluster shape, density-based and hierarchical-based clustering provides all shapes of clusters, while partitioning clustering provides spherical shapes of clusters. For cluster completeness, density-based clustering has the ability to determine the noise in a data set, which leaves out some documents from any cluster. A partitioning clustering technique does not determine any noise, which results in all documents belonging to a cluster. For time complexity, density-based and hierarchical-based clustering takes quadratic time, while partitioning clustering takes linear time. Aside from the mentioned criteria, the simplicity and less-complicated parameter settings have led to K-means being widely used as a clustering technique. As different clustering techniques have been discussed, K-means seems to be an appropriate partitioning clustering candidate and will be studied in a later section of this work.

3. Proposed Method

In this section, we describe the proposed method in terms of document representation and document similarity function, and the architecture of the proposed method is summarized and presented in Fig. 1. The method begins by creating a vector representation for vocabularies in the short text document data sets from the learned model on an external text corpus. Clustering is performed by the K-means based clustering algorithm on the dense vector representation, and, the similarity of documents is calculated by a document distance function. The document is assigned to the cluster with the nearest centroid. After each iteration, the centroids of each cluster are updated based on similarities of the members. The clustering algorithm runs iteratively until a standard stopping criterion is met.

3.1 Distributed Representation of Words

Distributed Representation of Words in vector space [22,32] is one of the most famous learning-based document representations and has become a popular way to capture the lexical, semantic, and syntactic similarity between words. This representation is generated by training the word relationships with a neural network model on the huge text corpus of a related domain. Each word is represented in vector space by a vector of a certain dimension. Assuming that there are V vocabularies and a real-valued vector of some fixed dimension D, each word w in the vocabulary is represented by $\boldsymbol{w}_i \in \mathbb{R}^d$.

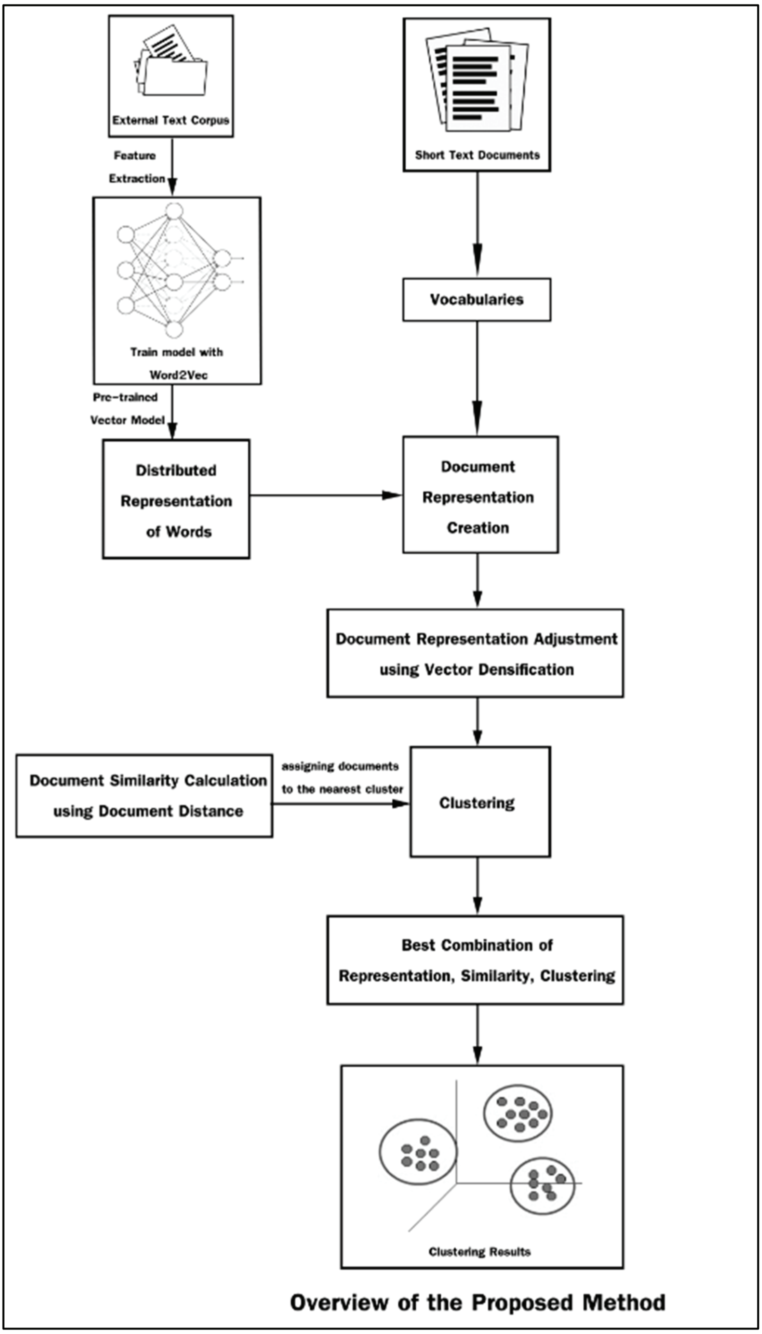

External Text Corpus
Feature Extraction
Train model with Word2Vec
Pre-trained Vector Model
Distributed Representation of Words
Short Text Documents
Vocabularies
Document Representation Creation
Document Representation Adjustment using Vector Densification
Document Similarity Calculation using Document Distance
assigning documents to the nearest cluster
Clustering
Best Combination of Representation, Similarity, Clustering
Clustering Results
Overview of the Proposed Method

Fig. 1. Architecture of the proposed short text clustering method.

To create a document representation with this technique, the Skip-gram model [22,32] is used. The concept of the model is to predict surrounding words in a document based on the current word. The objective of this Skip-gram model is to maximize the average of log probability, which is done by optimizing the neural network with input, projection, and output layers. With a given sequence of training words $w_1, w_2, \dots, w_T,$ the model is

$$\frac{1}{T}\sum_{j=1}^{T}\sum_{j\in[-c,c],j\neq 0} log\, p(w_{t+j}|\, w_t) \qquad (1)$$

where c denotes the interval of training context from the current center word w_T. As the value of c increases, it enables the model to increase the number of training words and the complexity of the word syntactic and semantic relationships to be learned. Thus, it results in increasing model accuracy, at the expense of training time. $p(w_{t+j}|w_t)$ is the hierarchical softmax function of the word vectors w_{t+j} and w_t. The process of creating Distributed Representation of Words is completely unsupervised learning, and it can also be trained on any corpus of text or even be a pre-trained model in advance. The vectors of contextually related or similar words are closely placed, while less related are farther apart. Word2Vec, a renowned word embedding procedure, is an implementation of the Skip-gram model architecture [22]. This implementation is used in the work and is also readily available through the Python Gensim [68] framework.

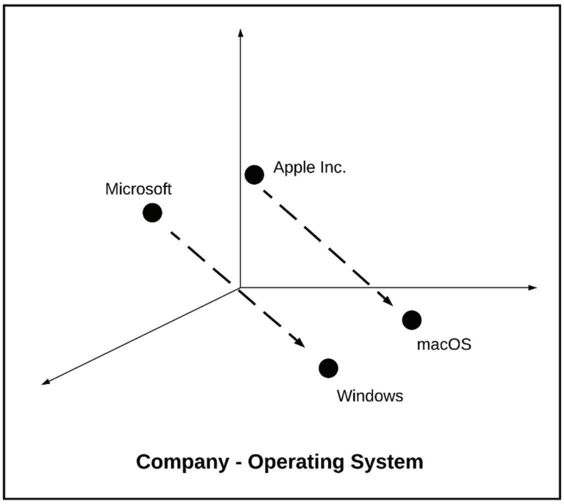


Fig. 2. An illustration of vector correlation between the company and the renowned operating system product in three-dimensional space. The more correlated vectors are placed closer together compared to less correlated vectors.

The input short text documents are preprocessed by standard text preprocessing techniques, such as the removal of stop-words and non-informative characters [69,70], leaving the relevant unique vocabularies in the short texts. To generate expressive word vectors, the pre-trained model is applied to these vocabularies, which helps expand on background knowledge from external sources. Thus, a particular document is represented by a group of aggregated word vectors. As a result, the document representations are generated, which will be further used for document clustering in the next step.

One of the special capabilities of the Distributed Representation of Words is the ability to automatically learn and capture the concepts and semantics of words to find their correlation. To illustrate the concept of this ability, an example is presented in Fig. 2. The figure shows the association of the technology company and the renowned operating system product. Microsoft Corporation is closely associated with Microsoft Windows, the operating systems that are developed, marketed, and sold by Microsoft

Corporation. On the other hand, Apple Inc. is closely associated with Mac OS, the operating systems that are developed and marketed by Apple Inc.

3.2 Document Similarity Calculation using Document Distance

To group related documents together as clusters, a measure of document similarity is needed. As mentioned, various document distance measurements have been proposed. In this work, WMD [29 works as a document distance measurement. WMD, based on the Earth Mover's Distance idea [44,45], measures how far the words in document D_1 must be "moved" to match the words in document D_2 to measure the similarity of the two documents. To apply WMD, documents must be represented by vectors of a certain fixed dimension *d* containing vocabularies, the unique words from the documents. The vocabularies are represented by the matrix **X** of size $d \times n$, where d is the vector dimension and n is the number of vocabularies. For any i^{th} column, the word embeddings of word i in vector space are represented by $\mathbf{x}_i \in \mathbb{R}^d$. The minimum cost of moving words in document D to document D is computed by the following document transportation metric,

$$\min_{T \in \mathbb{R}^+} \sum_{i,j=1}^{n} T_{i,j}\, c(i,j) \tag{2}$$

$c(i,j)$ is computed by the distance function $c(i,j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$ and refers to the semantic similarity between word i and word j. In simple words, it is the cost of traveling from one word to another. $T_{i,j}$ is a flow matrix that indicates how much of word i in D_i travels to word j in D_j and must be subject to $T_{i,j} \in \mathbb{R}^+$. Thus, the distance between two documents is calculated as the minimum cumulative cost in moving all words from document D_i to document D_j, $\sum_{i,j} T_{ij}\, c(i,j))$ [29].

WMD measures the dissimilarity of a pair of text documents and how much it costs to transform the vocabularies in one document into the other. Originally, Kusner et al. [28] proposed the document distance for text classification with the K-nearest neighbor algorithm. In this work, the same distance is applied for short text clustering. To illustrate the concept of short text similarity calculation, suppose there are two sentences, say D_1 and D_2, and a referenced query sentence D_0. To compare two sentences using WMD, stop-words in the sentences must be removed, leaving vocabularies in each sentence. The comparison is made on each pair of sentences, say D_0 and D_1. The vocabularies in D_0 are businessman, arrives, and Rolls-Royce, while D_1 contains CEO, appears, and Mercedes. The arrows pointing from each word i in D_1 to j in D_0 represent the travel cost of each word from word i to word j. The travel cost of Mercedes to Rolls-Royce is cheaper than wall to Rolls-Royce because the word2vec embedding places the vector of Mercedes closer to the vector Rolls-Royce, the luxury car, than the vector wall to the vector Rolls-Royce. The travel cost of document i to document j is the cumulative travel cost of all words in document i and document j, respectively. As a result, the travel cost of D_1 to D_0 (0.42) is outstandingly smaller than the cost of D_2 to D_0 (0.60). The illustration of this concept is shown in Fig. 3 (top). It is amazingly surprising to say that all of these documents share no words in common but are able to capture the semantic similarity of different documents. Alternately, the distance turns out to be equal in the case of capturing similarity by the bag-of-words/TF-IDF method.

Usually, the numbers of words in each sentence are not necessarily equal, and the sequences of words do not share the same pattern. As shown in Fig. 3 (bottom), sentence D_0 has three vocabularies, while sentence D_3 has seven vocabularies. Since these two sentences do not have the same number of vocabularies, the comparison is made using all pairs of vocabularies in the two sentences to determine the pair of words that has the lowest travel cost. To do so, there are outgoing and incoming weights from each of the sentences. This weight, together with the travel cost, is a main criteria in computing word moving distance to find the semantic and syntactic similarity of documents. This algorithm is implemented in this work to handle this issue.

Document distance, a document similarity measure, determines the difference between the document (represented by a vector) and the centroid vector of each cluster. The document is assigned to the cluster with the smallest document-centroid distance.

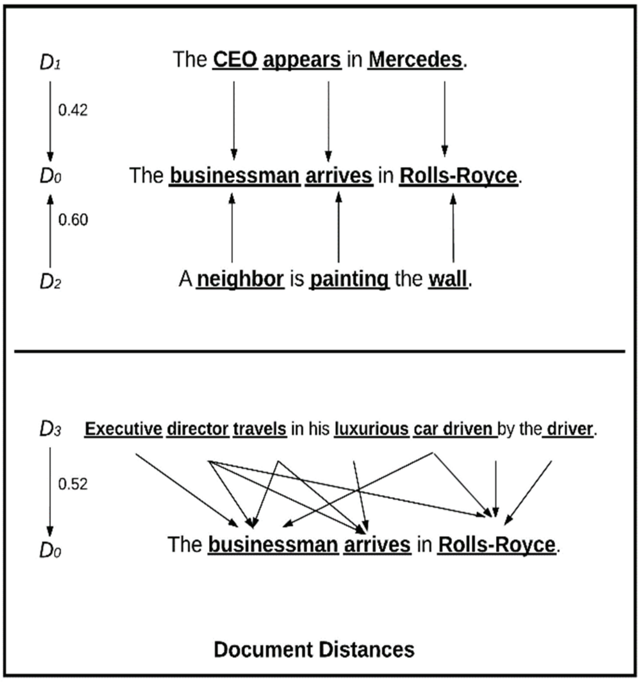

D1
The CEO appears in Mercedes.
0.42
D0
The businessman arrives in Rolls-Royce.
0.60
D2
A neighbor is painting the wall.
D3
Executive director travels in his luxurious car driven by the driver.
0.52
D0
The businessman arrives in Rolls-Royce.
Document Distances

Fig. 3. (Top) The movement in comparison between the query sentence D_0 and the two sentence D_1 and D_2. These two documents have the same bag-of-words distance to D_0. The arrows represent movement between two documents which are labeled with cumulative distance of words in each document. (Bottom) The movement in comparison between the query sentence D_0 and sentence D_3 where the two sentences have different number of words. This causes WMD to compare all pairs of similar words in these sentences. Note that the **underlined bold** word represents the vocabulary of that sentence.

3.3 Document Representation Adjustment using Vector Densification

As addressed by Song and Roth [71], Phan et al. [72], and Shrestha [73], short text documents consist of the range from a very few words to a dozen words per document. The document representations are comprised of many zeroes. As a result, data sparseness is a main issue. Data sparseness has an impact not only on the memory used but also on the processing time.

To address sparse data, the document representation must undergo the vector densification process. To illustrate the concept, the process is summarized and illustrated in Fig. 4. It is shown that the sparse vectors *S* consist of zero and non-zero elements. Because there are many zero elements, it requires more memory storage to store these meaningless values and also more processing time. Thus, these sparse vectors *S* should be condensed to a dense form of vectors to consume fewer computational resources. To condense the sparse vector *S*, only the non-zero elements along with their indices are stored and transformed into the dense vector *D* by the following transformation equation,

$$D = [s \,|s \ \neq 0, \forall s \in S] \tag{3}$$

As a result, a dense vector is generated and used as the document representation in the clustering process.

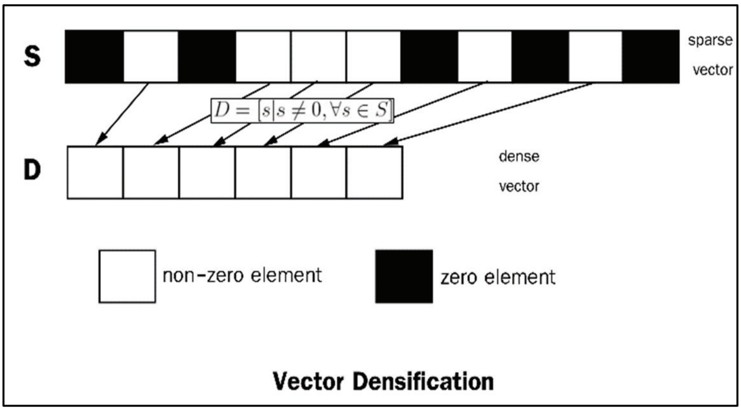


Fig. 4. Text document representation adjustment by the vector densification process.

3.4 Discussion on Time Reduced for Vector Densification

Assumed that the size of document representation vector is *N* and the size of densed document representation vector is *n*, where *n* << *N*. Considering the short text document clustering for document representation vector, the time it takes to cluster is *O*(2*Nkdl*), where *k* is the number of cluster, *d* is number of documents in data set, and *l* is the number of iteration in running the clustering. On the other hand, clustering short text document using densed document representation vector takes *O*(*N*+*n*+2*nkdl*).

Since $k \geq 1$, $d \geq 1$, $l \geq 1$, the term *kdl* should be always greater than 1. From *O*(*N*+*n*+2*nkdl*) and *O*(2*Nkdl*), $kdl \geq 1$, *O*(*N*+*n*+2*nkdl*) becomes *O*(*N*+3*n*), and *O*(2*Nkdl*) becomes *O*(2*N*).

Because $O(N+3n) \leq O(2N)$, *N*+3*n* is always less than or equal to 2*N*. Thus, $n \leq N/3$. This means that *n* is less than *N* on a factor of 3. As a result, the time it takes to run densed document representation vector is at least three times less than the time for document representation vector. This can be concluded that densed document representation helps the algorithm running faster.

3.5 Short Text Document Clustering

To cluster short text documents, the method begins by generating a pre-trained vector model from an external huge text corpus of a related domain. From this model, the Distributed Representation of Words is generated. With the interested short text document data set, the unique words are extracted as vocabularies represented by the vector of fixed dimension. The detailed process of creating the document representation has previously been explained in Section 3.1. As previously mentioned in Section 3.3, text representations for short text documents in high-dimensional space are very sparse. Thus, these representations are processed with a vector densification mechanism to reduce the dimensions. A clustering process is performed using the K-means clustering algorithm [53], where the number of desired document clusters depends on the number of different classes in the documents. Documents are assigned to the cluster with the lowest document distance between document representation and cluster centroid. The document distance function is modified to handle the dense document representation. The details of document distance were previously explained in Section 3.2. After the passing of each iteration, the cluster centroids are updated based on their members' similarities. The K-means algorithm runs iteratively until a standard terminating condition is satisfied. An example of the short text document clustering process is presented in Fig. 5.

From Fig. 5, we can see that the text documents are preprocessed and then represented in the form of Distributed Representation of Words. The representation is condensed and undergoes the clustering process. Finally, after the passing of certain iterations, document clustering results are available for evaluation.

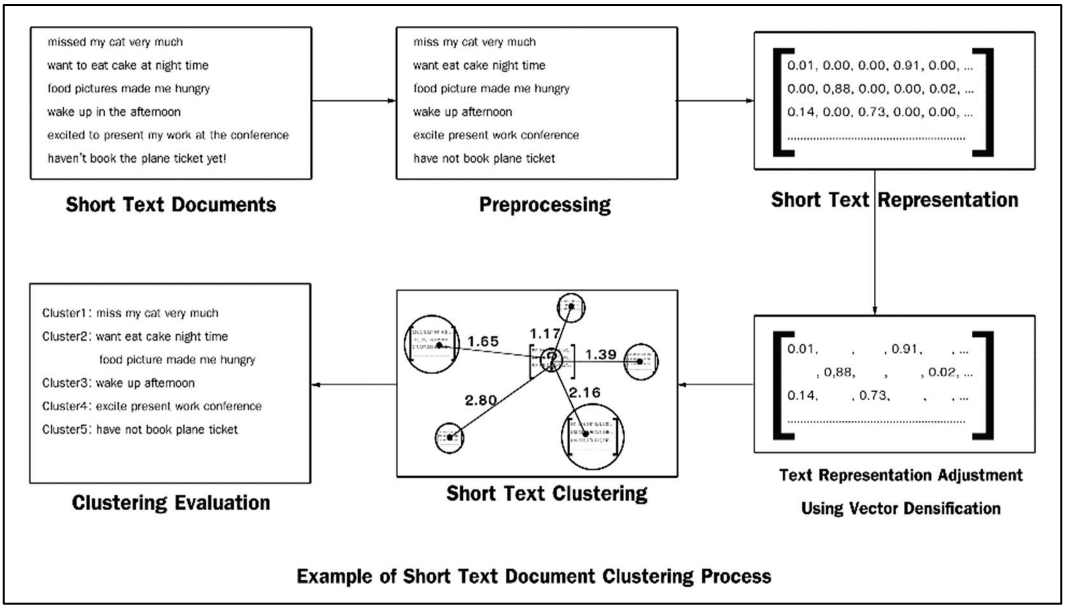

missed my cat very much
want to eat cake at night time
food pictures made me hungry
wake up in the afternoon
excited to present my work at the conference
haven't book the plane ticket yet!
Short Text Documents
miss my cat very much
want eat cake night time
food picture made me hungry
wake up afternoon
excite present work conference
have not book plane ticket
Preprocessing
0.01, 0.00, 0.00, 0.91, 0.00, ...
0.00, 0.88, 0.00, 0.00, 0.02, ...
0.14, 0.00, 0.73, 0.00, 0.00, ...
Short Text Representation
Cluster1 : miss my cat very much
Cluster2: want eat cake night time
food picture made me hungry
Cluster3: wake up afternoon
Cluster4: excite present work conference
Cluster5: have not book plane ticket
Clustering Evaluation
1.65
1.17
1.39
2.80
2.16
Short Text Clustering
0.01, , 0.91, , ...
, 0.88, , , 0.02, ...
0.14, , 0.73, , , ...
Text Representation Adjustment
Using Vector Densification
Example of Short Text Document Clustering Process

Fig. 5. Example of the Short Text Document Clustering Process.

4. Experimental Results

4.1 Datasets

To validate the effectiveness of the proposed method, experiments were conducted on the following four publicly available short text document data sets:

BBC News is a corpus of the BBC website's news headlines collected in the years 2004–2005 by Greene and Cunningham and was firstly used as the baseline data set in their work [74]. There are altogether 2,225 short text documents surrounding 5 different areas: business, tech, politics, entertainment, and sport.

SearchSnippets is a corpus of Google's web search transactions collected by Phan et al. and was firstly used as the baseline data set in their work [72]. There are altogether 12,880 documents surrounding 8 different areas: business, computers, culture-arts-entertainment, education-science, engineering, health, politics-society, and sports.

StackExchange is a corpus of the questions, posts, and comments asked on the Stack Exchange community's web forum (https://archive.org/details/stackexchange). In the experiments, we randomly select short texts from 8 different classes. For this data set, we performed standard preprocessing techniques for the text as described in [75].

Twitter is a social media service where users can post "tweets", short status messages. According to Twitter's usage policies, each tweet is limited to no more than 140 characters [76]. Hence, Twitter represents one of a largest real-world user-generated social media data sets [6,77]. This dataset was randomly collected by the authors using Python's Tweepy (https://github.com/tweepy/tweepy) library to connect with the Twitter Streaming API. The data consists of only English tweets surrounding 5 different areas. The standard text preprocessing steps and basic text cleanups, such as removing symbols, URLs, stop-words (https://www.ranks.nl/stopwords), and user mentions, were conducted as in [6,75,78].

The characteristics of the data sets are summarized in Table 1.

Table 1. Text data set characteristics

Data set	No. of words		No. of vocab.	No. of classes	Description	Source
	Avg	Max				
BBC News	4.4	7	3,712	5	BBC News Headlines	[74]
SearchSnippets	17.9	38	30,646	8	Web search transactions	[72]
StackExchange	11.0	160	13,888	8	Questions, posts, comments	StackExchange
Twitter	8.7	21	65,454	5	Social Media Statuses	Twitter

4.2 Experimental Setup

The experiments were conducted by comparing the proposed method against other baseline techniques in terms of document representations, document similarity, and document clustering. For document representations, TF-IDF and Distributed Representations were used to represent text documents. TF-IDF represents text documents by vectors of the term frequencies in that document. The dimension of the vector representations generated by this method equals the size of the vocabulary. For the Distributed Representations, the documents were trained using the publicly available word2vec (https://code.google.com/p/word2vec) tool on a large document corpus of a related domain. The parameter settings for this training were set as in [22]. Text representations of BBC and SearchSnippets data sets were trained on Wikipedia dumps (https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2). For the StackExchange and Twitter data sets, the text representations were trained on their entire cleaned corpus. To cluster documents, K-means, CHAMELEON, and DBSCAN were used to cluster these text representations into groups based on their similarity. Euclidean distance, Cosine similarity, and WMD were used to find the similarities of documents.

The experimental results were compared against different baseline techniques. TF-IDF served as baseline for document representation. DBSCAN and CHAMELEON served as baseline for document clustering. For document similarity, Euclidean distance and Cosine similarity were the baseline methods.

For each data set, the experiments were performed using the different combinations of document representation, document similarity, and document clustering techniques. The experimental results were evaluated in terms of clustering quality, which includes precision, recall, F1, and adjusted Rand index (ARI).

4.3 Baseline Document Representation, Document Similarity, and Document Clustering

The following document representations, document similarity functions, and document clustering algorithms, in addition to the proposed method, were used in these experiments:

TF-IDF [31]: a text document representation that uses frequency of terms in a document divided by the frequency of each term in the entire document corpus.

Euclidean distance [43]: a standard metric for measuring the distance between two points, which is also used in text clustering. This is the default distance measure in the K-means clustering algorithm.

Cosine similarity [43]: a popular metric for measuring the cosine angle between two vectors, a representation for each text document. The cosine of the angle between vectors measures the similarity, closeness, and relatedness of the documents. A significant property of this metric is the independence of document length.

CHAMELEON hierarchical clustering [25]: one of agglomerative hierarchical clustering technique that operates on a sparse K-nearest neighbor graph. In this graph, nodes represent data items, and weighted edges represent their similarities. "Two-phase algorithm" is used to form clusters of these data. First, a graph partitioning algorithm is applied to the K-nearest neighbor graph to cluster data items into several small sub-clusters. Then, these sub-clusters are repeatedly merged to find the genuine clusters. The similarity of the clusters is determined by the two values: inter-connectivity and closeness.

DBSCAN [57]: one of the density-based clustering algorithms that groups data points into clusters by the density. DBSCAN performs clustering using three types of points: core points, border points, and noise points. The algorithm first identifies the core points by the density of points in a region of a specified radius around the point in which the density is higher than a certain threshold. The border points form the borders of clusters, which contain core points in their radii, but the number of neighbor points is lower than the specified minimum neighbor points. The noise points are the points that contain less points than the specified minimum neighbor points and do not contain any core points in their radius at all.

4.4 Clustering Quality

To validate the effectiveness of clustering, clustering measurements were used. Clustering measurements of the four data sets were measured by precision, recall, F1-score, and ARI. The experiments were conducted on the combinations of different document representations, document similarity functions, and document clustering methods. The clustering quality results are summarized and reported in Table 2.

Table 2. Clustering quality in terms of statistical average of precision, recall, F1-score, and ARI

	Precision			Recall			F1			ARI		
	K	**D**	**C**	**K**	**D**	**C**	**K**	**D**	**C**	**K**	**D**	**C**
Method (W)												
1	0.47	0.23	0.19	0.42	0.00	0.08	0.39	0.00	0.11	0.09	0.13	0.52
2	0.47	0.17	0.28	0.43	0.00	0.15	0.40	0.01	0.19	0.11	0.07	0.62
3	0.84	0.23	0.14	0.28	0.00	0.06	0.22	0.00	0.09	0.00	0.13	0.21
4	0.87	0.13	0.02	0.64	0.01	0.02	0.58	0.01	0.02	0.50	0.01	0.16
5	0.98	0.23	0.31	0.98	0.00	0.12	0.98	0.00	0.17	0.94	0.56	0.27
Method (X)												
1	0.89	0.00	0.25	0.89	0.00	0.14	0.89	0.00	0.18	0.77	0.00	0.69
2	0.89	0.12	0.17	0.89	0.01	0.10	0.89	0.01	0.12	0.77	0.04	0.41
3	0.89	0.00	0.17	0.36	0.00	0.16	0.36	0.00	0.16	0.05	0.00	0.50
4	0.93	0.08	0.04	0.93	0.02	0.07	0.80	0.02	0.50	0.87	0.06	0.19
5	0.99	0.06	0.00	0.99	0.00	0.00	0.99	0.00	0.00	0.97	0.50	0.42
Method (Y)												
1	0.27	0.25	0.25	0.26	0.01	0.19	0.25	0.01	0.21	0.06	0.01	0.79
2	0.49	0.09	0.19	0.46	0.01	0.13	0.47	0.01	0.15	0.15	0.04	0.38
3	0.81	0.00	0.12	0.21	0.00	0.09	0.19	0.00	0.10	0.00	0.00	0.48
4	0.76	0.06	0.08	0.68	0.01	0.09	0.66	0.01	0.08	0.47	0.08	0.40
5	0.85	0.06	0.11	0.77	0.00	0.08	0.76	0.00	0.09	0.55	0.50	0.47
Method (Z)												
1	0.33	0.00	0.30	0.18	0.00	0.60	0.22	0.00	0.09	0.160	0.001	0.25
2	0.41	0.22	0.38	0.13	0.02	0.10	0.18	0.02	0.14	0.064	0.033	0.16
3	0.98	0.15	0.30	0.56	0.00	0.11	0.58	0.00	0.15	0.840	0.006	0.16
4	0.99	0.12	0.16	0.87	0.02	0.06	0.87	0.03	0.08	0.959	0.067	0.12
5	0.99	0.31	0.23	0.99	0.00	0.08	0.99	0.00	0.10	0.987	0.336	0.22

K=K-means, D=DBSCAN, C=CHAMELEON, Method 1=TF-IDF+Euclidean, Method 2=TF-IDF+Cosine Similarity, Method 3=Distributed Representation+Euclidean, Method 4=Distributed Representation+Cosine Similarity, Method 5=Distributed Representation+WMD, (W)=BBC News, (X)=SearchSnippets, (Y)=StackExchange, (Z)=Twitter.

TF-IDF is a statistical-based text document representation generated from the frequency of the word occurrence in the document. This representation heavily relies on the term overlap, which occurs extremely rarely for this type of document. Alternately, Distributed Representation is learning-based text document representation generated by analyzing the relationship of terms and their co-occurrences. Furthermore, this document representation is aggregated with contextual information from an external background knowledge. Thus, learning-based text representation always performs better than statistical-based.

The experiments were conducted on two types of document similarity: string-based and knowledge-based. As previously mentioned, Euclidean and Cosine similarity are string-based similarities, while WMD is knowledge-based similarity. String-based similarity compares the strings on a word-by-word basis, but WMD incorporate the background knowledge to determine the text similarity. Knowledge-based similarity can capture not only the syntactic similarity but also the semantics as well, especially considering the characteristics of short text documents (short and with very few words). The results show that suitable document similarity also alleviates the issues. The combination of learning-based text representation with knowledge-based document similarity always performs better than any other.

Considering document clustering methods, experiments on density-based, hierarchical-based, and partitioning clustering were also conducted. As previously mentioned, the main considered factors in selecting suitable clustering methods are the ability to handle large-dimensional data, cluster shape, cluster completeness, and time complexity. The experimental results have shown that a partitioning clustering method (K-means) is more suitable for clustering short text documents than density-based clustering (DBSCAN) due to the suitable clustering factors.

As experiments have been conducted on different combinations of document representations, document similarity functions, and document clustering methods, the experimental results are summarized in Table 2. Representing documents by TF-IDF with either Euclidean or cosine similarity as the document similarity function and clustering with either DBSCAN or CHAMELEON usually yield poorer clustering quality than K-means. For representing documents by Distributed Representation with either Euclidean or Cosine similarity as the document similarity function, clustering with either DBSCAN or CHAMELEON usually yield poorer clustering quality than K-means, as well. Comparing different text representations, representing documents by Distributed Representation with K-means as the document clustering method performs better than TF-IDF with K-means. Alternately, clustering with DBSCAN or CHAMELEON, regardless of document representation and document similarity function, usually provides lower clustering quality than K-means despite the document representation and similarity function. However, the use of Distributed Representation as document representation, WMD as document similarity function, and K-means as clustering method outperforms all other combinations for all data sets and clustering quality measures.

4.5 Impact of Data Set Characteristics

The experimental results have shown that, from different combinations of document representation, document similarity function, document clustering method, the use of Distributed Representation as document representation, WMD as document similarity function, and K-means as document clustering method outperforms any other technique in terms of different clustering quality measures. The experiments were conducted on four different publicly available short text data sets, which significantly confirmed that the proposed method performed better than the other baseline techniques. From Table 3, we can see that the data sets BBC News, SearchSnippets, and Twitter resulted in high clustering quality. These three data sets were composed of text documents of normal linguistic usage. However, Twitter data set sometimes contained symbolic characters but were cleaned up in the preprocessing step. For the StackExchange data set, the clustering method underperformed the other three techniques in terms of the clustering quality. By manually inspecting the data set, we could see that the documents from scientific web forums are usually composed of equations and variables. Although humans can distinguish the equations and variables from normal text in the documents and know that they convey some meaning, the clustering methods do not realize this difference and treat all of them the same, i.e., as normal text characters. Thus, this is the main factor that significantly impacts the clustering quality of this data set.

Table 3. Clustering quality of the proposed method (Distributed Representation + WMD + K-means) on different data sets

Data set	Precision	Recall	F1	ARI
BBC News	0.978	0.978	0.978	0.944
SearchSnippets	0.990	0.990	0.990	0.970
StackExchange	0.850	0.770	0.760	0.550
Twitter	0.995	0.995	0.995	0.987

Table 1 shows characteristics of the short text document data sets, including average number of words, maximum number of words, number of vocabularies, number of different classes of each data set, data set descriptions, and data set sources. We can see that documents that are too short and sparse, such as documents from BBC News data set, perform poorer, although acceptable, than those that are longer, such as Twitter. As a conclusion from Table 3, to achieve the best clustering quality, the minimum document length of incorporating the proposed technique should be at least, on average, 9 words per document.

4.6 Clustering Time

Table 4 presents the running time of each method combination. As previously mentioned, the time complexity of K-means is linear, but that of DBSCAN and CHAMELEON are non-linear. By comparing clustering methods, we can see that clustering by DBSCAN and CHAMELEON take much longer than by K-means. Furthermore, considering the running time with the data set characteristics in Table 1, we can see that the running time of the combination of Distributed Representation + WMD with K-means is directly proportional to the average number of words in the data set. More precisely, time complexity of the combination of Distributed Representation + WMD with K-means is of $O(kdpm)$ where $|k|$ is the

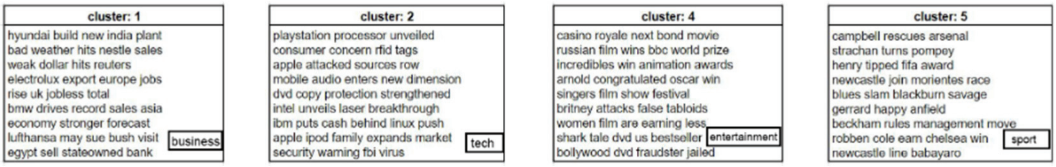
Table 4. Time spent running each method

	Time (sec)		
	K	**D**	**C**
Method (Data set: BBC News)			
Method 1: TF-IDF + Euclidean	5	186	73
Method 2: TF-IDF + Cosine Similarity	5	46	182
Method 3: Distributed Representation + Euclidean	1	166	100
Method 4: Distributed Representation + Cosine Similarity	1	23	230
Method 5: Distributed Representation + WMD	24	25,245	944
Method (Data set: SearchSnippets)			
Method 1: TF-IDF + Euclidean	8	48	17
Method 2: TF-IDF + Cosine Similarity	8	14	54
Method 3: Distributed Representation + Euclidean	1	35	30
Method 4: Distributed Representation + Cosine Similarity	1	19	77
Method 5: Distributed Representation + WMD	76	24,909	234
Method (Data set: StackExchange)			
Method 1: TF-IDF + Euclidean	7	27	10
Method 2: TF-IDF + Cosine Similarity	8	10	37
Method 3: Distributed Representation + Euclidean	1	17	21
Method 4: Distributed Representation + Cosine Similarity	1	3	41
Method 5: Distributed Representation + WMD	65	13,870	120
Method (Data set: Twitter)			
Method 1: TF-IDF + Euclidean	4	66	17
Method 2: TF-IDF + Cosine Similarity	5	6	36
Method 3: Distributed Representation + Euclidean	1	14	24
Method 4: Distributed Representation + Cosine Similarity	1	2	41
Method 5: Distributed Representation + WMD	34	11,025	147

number of clusters, |*d*| is the number of documents, |*p*| is the number of unique words in documents, |*m*| is the number of iterations before reaching condition of the stopping criteria. The smallest average number of words is from the BBC News data set, and the highest is SearchSnippets. In addition, the runtime of the proposed combination is smallest for BBC News and highest for SearchSnippets. Furthermore, the runtimes of DBSCAN and CHAMELEON as a clustering algorithm are much higher than that of K-means. Thus, K-means serves as a suitable document clustering method of this work.

4.7 Clustering Outputs

A sample of clustered documents are from data set BBC News headlines, shown in Fig. 6. From the figure, we can see that documents in cluster 1 talk about business, cluster 2 about technology, cluster 4 about entertainment, and cluster 5 about sports. By manually inspecting the clustering outputs against the ground truths by authors, it is confirmed that the documents in each cluster are semantically and contextually related, which corresponds to the clustering quality of the proposed method. The manual inspection was not only conducted on the BBC News headlines data set but also on the SearchSnippets, StackExchange, and Twitter data sets as well. The proposed clustering methods contextually group related documents together in the same group. From the clustering quality in Table 3, it seems that, although the clustering process did not result in totally pure clusters of documents, the clustering output reasonably represents the group members.



cluster: 1

hyundai build new india plant
bad weather hits nestle sales
weak dollar hits reuters
electrolux export europe jobs
rise uk jobless total
bmw drives record sales asia
economy stronger forecast
lufthansa may sue bush visit
egypt sell stateowned bank

business

cluster: 2

playstation processor unveiled
consumer concern rfid tags
apple attacked sources row
mobile audio enters new dimension
dvd copy protection strengthened
intel unveils laser breakthrough
ibm puts cash behind linux push
apple ipod family expands market
security warning fbi virus

tech

cluster: 4

casino royale next bond movie
russian film wins bbc world prize
incredibles win animation awards
arnold congratulated oscar win
singers film show festival
britney attacks false tabloids
women film are earning less
shark tale dvd us bestseller
bollywood dvd fraudster jailed

entertainment

cluster: 5

campbell rescues arsenal
strachan turns pompey
henry tipped fifa award
newcastle join morientes race
blues slam blackburn savage
gerrard happy anfield
beckham rules management move
robben cole earn chelsea win
newcastle line babayaro

sport

Fig. 6. Clustering outputs of clustered documents.

5. Conclusions and Future Research

This paper proposes an approach for grouping short text documents into different clusters of related contexts. However, because short text documents usually contain very few words, this leads to sparsity issues, while normal text representation techniques do not provide acceptable results. Thus, suitable document representations, document similarity functions, and document clustering techniques are presented. To attain the best results, Distributed Representation of Words are used to represent the vocabularies in documents in the form of vectors that incorporate background knowledge by using external knowledge sources. The document clustering is performed with K-means clustering, and the WMD serves as document similarity measurement. To validate the effectiveness of the proposed method, the clustering quality is conducted on the experimental results in terms of precision, recall, F1-scores, and ARI. In addition to the clustering quality, a manual inspection on the clustering results by the authors also confirms the effectiveness of this proposed method. Nevertheless, there are still issues that can be further improved, such as performing clustering in a real-time manner to cluster text documents that arrive as stream, which should be conducted in future works.

References

[1] V. K. R. Sridhar, "Unsupervised topic modeling for short texts using distributed representations of words," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, Denver, CO, 2015, pp. 192-200.

[2] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "A brief survey of text mining: classification, clustering and extraction techniques," 2017, https://arxiv.org/abs/1707.02919.

[3] V. Gupta and G. S. Lehal, "A survey of text mining techniques and applications," *Journal of Emerging Technologies in Web Intelligence*, vol. 1, no. 1, pp. 60-76, 2009.

[4] A. M. Jadhav and D. P. Gadekar, "A survey on text mining and its techniques," *International Journal of Science and Research*, vol. 3, no. 11, pp. 2110-2113, 2014.

[5] C. C. Aggarwal, "Mining text and social streams: a review," *ACM SIGKDD Explorations Newsletter*, vol. 15, no. 2, pp. 9-19, 2014.

[6] L. F. S. Coletta, N. F. F. da Silva, E. R. Hruschka, and E. R. Hruschka, "Combining classification and clustering for tweet sentiment analysis," in *Proceedings of 2014 Brazilian Conference on Intelligent Systems*, Sao Paulo, Brazil, 2014, pp. 210-215.

[7] S. Sharma and V. Gupta, "Recent developments in text clustering techniques," *International Journal of Computer Applications*, vol. 37, no. 6, pp. 14-19, 2012.

[8] L. Rokach and O. Maimon, *Clustering Methods*. Boston, MA: Springer, 2005.

[9] C. C. Aggarwal and C. Zhai, "A survey of text clustering algorithms," in *Mining Text Data*. Boston, MA: Springer, 2012, pp. 77-128.

[10] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval,*" Journal of Documentation*, vol. 28, no. 1, pp. 11-21, 1972.

[11] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY: McGraw-Hill Inc., 1986.

[12] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the 1st Instructional Conference on Machine Learning,* Washington, DC, 2003, pp. 133-142.

[13] J. Xu, P. Wang, G. Tian, B. Xu, J. Zhao, F. Wang, and H. Hao, "Short text clustering via convolutional neural networks," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing,* Denver, CO, 2015, pp. 62-69.

[14] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," 2014, https://arxiv.org/abs/1404.2188.

[15] J. Xu, B. Xu, P. Wang, S. Zheng, G. Tian, and J. Zhao, "Self-taught convolutional neural networks for short text clustering," *Neural Networks*, vol. 88, pp. 22-31, 2017.

[16] C. Ma, Q. Zhao, J. Pan, and Y. Yan, "Short text classification based on distributional representations of words," *IEICE Transactions on Information and Systems*, vol. 99, no. 10, pp. 2562-2565, 2016.

[17] Y. Yan, R. Huang, C. Ma, L. Xu, Z. Ding, R. Wang, T. Huang, and B. Liu, "Improving document clustering for short texts by long documents via a Dirichlet multinomial allocation model," in *Web and Big Data*. Cham: Springer, 2017, pp. 626-641.

[18] L. Hong and B. D. Davison, "Empirical study of topic modeling in twitter," in *Proceedings of the 1st Workshop on Social Media Analytics*, Washington, DC, 2010, pp. 80-88.

[19] J. Weng, E. P. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, New York, NY, 2010, pp. 261-270.

[20] R. Mehrotra, S. Sanner, W. Buntine, and L. Xie, "Improving LDA topic models for microblogs via tweet pooling and automatic labeling," in *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Dublin, Ireland, 2013, pp. 889-892.

[21] X. Quan, C. Kit, Y. Ge, and S. J. Pan, “Short and sparse text topic modeling via self-aggregation,” in *Proceedings of the 24th International Conference on Artificial Intelligence*, Buenos Aires, Argentina, 2015, pp. 2270-2276.

[22] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Advances in Neural Information Processing Systems*, vol. 26, pp. 3111-3119, 2013.

[23] M. Steinbach, G. Karypis, and V. Kumar, “A comparison of document clustering techniques,” in *Proceedings of the International KDD Workshop on Text Mining*, Boston, MA, 2000.

[24] P. B. Nagpal and P. A. Mann, “Comparative study of density based clustering algorithms,” *International Journal of Computer Applications*, vol. 27, no. 11, pp. 421-435, 2011.

[25] G. Karypis, E. H. Han, and V. Kumar, “Chameleon: hierarchical clustering using dynamic modeling,” *Computer*, vol. 32, no. 8, pp. 68-75, 1999.

[26] K. Mumtaz and K. Duraiswamy, “A novel density based improved k-means clustering algorithm - Dbkmeans,” *International Journal on Computer Science and Engineering*, vol. 2, no. 2, pp. 213-218, 2010.

[27] A. Karami and R. Johansson, “Choosing DBSCAN parameters automatically using differential evolution,” *International Journal of Computer Applications*, vol. 91, no. 7, pp. 1-11, 2014.

[28] W. H. Gomaa and A. A. Fahmy, “A survey of text similarity approaches,” *International Journal of Computer Applications*, vol. 68, no. 13, pp. 13-18, 2013.

[29] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, “From word embeddings to document distances,” in *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, 2015, pp. 957-966.

[30] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA, 1967, pp. 281-297.

[31] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information Processing and Management*, vol. 24, no. 5, pp. 513-523, 1988.

[32] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” 2013, https://arxiv.org/abs/1301.3781.

[33] C. De Boom, S. Van Canneyt, S. Bohez, T. Demeester, and B. Dhoedt, “Learning semantic similarity for very short texts,” in *Proceedings of 2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, Atlantic City, NJ, 2015, pp. 1229-1234.

[34] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391-407, 1990.

[35] T. Hofmann, “Probabilistic latent semantic indexing,” in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley CA, 1999, pp. 50-57.

[36] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet allocation,” *Journal of Machine Learning Research*, vol. 3, pp. 993-1022, 2003.

[37] D. E. Rumelhart, J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition (Volume 1: Foundations)*. Cambridge, MA: MIT Press, 1986.

[38] J. L. Elman, “Distributed representations, simple recurrent networks, and grammatical structure,” *Machine Learning*, vol. 7, no. 2-3, pp. 195-225, 1991.

[39] T. Kenter and M. de Rijke, “Short text similarity with word embeddings,” in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, Melbourne, Australia, 2015, pp. 1411-1420.

[40] J. Qiang, P. Chen, T. Wang, and X. Wu, “Topic modeling over short texts by incorporating word embeddings,” in *Advances in Knowledge Discovery and Data Mining*. Cham: Springer International Publishing, 2017, pp. 363-374.

[41] A. Karandikar, “Clustering short status messages: a topic model based approach,” M.S. thesis, Faculty of the Graduate School, University of Maryland Baltimore County, Baltimore, MD, 2010.

[42] A. Barron-Cedeno, P. Rosso, E. Agirre, and G. Labaka, "Plagiarism detection across distant language pairs," in *Proceedings of the 23rd International Conference on Computational Linguistics*, Stroudsburg, PA, 2010, pp. 37-45.

[43] A. Huang, "Similarity measures for text document clustering," in *Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC)*, Christchurch, New Zealand, 2008, pp. 49-56.

[44] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Proceedings of the 6th International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, Bombay, India, 1998, pp. 59-66.

[45] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99-121, 2000.

[46] J. A. Hartigan, *Clustering Algorithms*. New York, NY: John Wiley & Sons Inc., 1975.

[47] P. N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Boston, MA: Pearson Education Inc., 2006.

[48] J. Soler, F. Tence, L. Gaubert, and C. Buche, "Data clustering and similarity," in *Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference (FLAIRS'13)*, St Pete Beach, FL, 2013, pp. 492-495.

[49] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, MA, 2000.

[50] D. A. Ingaramo, M. L. Errecalde, and P. Rosso, "Density-based clustering of short-text corpora," *Procesamiento del Lenguaje Natural*, vol. 41, pp. 81-87, 2008.

[51] A. Rangrej, S. Kulkarni, and A. V. Tendulkar, "Comparative study of clustering techniques for short text documents," in *Proceedings of the 20th International Conference Companion on World Wide Web*, Hyderabad India, 2011, pp. 111-112.

[52] N. Singh and N. S. Chaudhari, "A novel clustering technique for short texts," in *Proceedings of 2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, India, 2016, pp. 228-232.

[53] T. S. Madhulatha, "An overview on clustering methods," 2012, https://arxiv.org/abs/1205.1117.

[54] H. Singh, "Clustering of text documents by implementation of k-means algorithms," *Streamed Info-Ocean*, vol. 1, pp. 53-63, 2016.

[55] Y. Chen and L. Tu, "Density-based clustering for real-time stream data," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Jose, CA, 2007, pp. 133-142.

[56] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Upper Saddle River, NJ: Prentice-Hall Inc., 1988.

[57] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, OR, 1996, pp. 226-231.

[58] Z. Miller, B. Dickinson, W. Deitrick, W. Hu, and A. H. Wang, "Twitter spammer detection using data stream clustering," *Information Sciences*, vol. 260, pp. 64-73, 2014.

[59] E. K. Ikonomakisa, D. K. Tasoulisa, and M. N. Vrahatisa, "Density based text clustering," in *Recent Progress in Computational Sciences and Engineering.* Boca Raton, FL: Taylor & Francis, 2006, pp. 218-221.

[60] S. Yang and Y. Wang, "Density-based clustering of massive short messages using domain ontology," in *Proceedings of 2009 Asia-Pacific Conference on Information Processing*, Shenzhen, China, 2009, pp. 505-508.

[61] M. T. H. Elbatta and W. M. Ashour, "A dynamic method for discovering density varied clusters," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 6, no. 1, pp. 123-134, 2013.

[62] M. Parimala, D. Lopez, and N. Senthilkumar, "A survey on density based clustering algorithms for mining large spatial databases," *International Journal of Advanced Science and Technology*, vol. 31, pp. 59-66, 2011.

[63] K. Sawant, "Adaptive methods for determining DBSCAN parameters," *International Journal of Innovative Science, Engineering & Technology*, vol. 1, no. 4, pp. 329-334, 2014.

[64] A. K. Pujari, *Data Mining Techniques*. Hyderabad, India: Universities Press (India) Private Limited, 2001.

[65] V. K. Singh, N. Tiwari, and S. Garg, "Document clustering using k-means, heuristic k-means and fuzzy c-means," in *Proceedings of the 2011 International Conference on Computational Intelligence and Communication Networks*, Gwalior, India, 2011, pp. 297-301.

[66] S. C. Punitha and M. Punithavalli, "A comparative study to find a suitable method for text document clustering," *International Journal of Computer Science & Information Technology (IJCSIT)*, vol. 3, no. 6, pp. 49-59, 2011.

[67] S. T. Deokar, "Text documents clustering using k means algorithm," *International Journal of Technology & Engineering Science (IJTES)*, vol. 1, no. 4, pp. 282-286, 2013.

[68] R. Rehurek and P. Sojka, "Software framework for topic modelling with large corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta, Malta, 2010, pp. 45-50.

[69] D. Sailaja, M. Kishore, B. Jyothi, and N. R. G. K. Prasad, "An overview of pre-processing text clustering methods," *International Journal of Computer Science & Information Technologies*, vol. 6, o. 3, pp. 3119-3124, 2015.

[70] A. I. Kadhim, Y. N. Cheah, and N. H. Ahamed, "Text document preprocessing and dimension reduction techniques for text document clustering," in *Proceedings of the 2014 4th International Conference on Artificial Intelligence with Applications in Engineering and Technology*, Kota Kinabalu, Malaysia, 2014, pp. 69-73.

[71] Y. Song and D. Roth, "Unsupervised sparse vector densification for short text similarity," in *Proceedings of 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, CO, 2015, pp. 1275-1280.

[72] X. H. Phan, L. M. Nguyen, and S. Horiguchi, "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *Proceedings of the 17th International Conference on World Wide Web*, Beijing, China, 2008, pp. 91-100.

[73] P. Shrestha, "Corpus-based methods for short text similarity," in *Proceedings of the 17th Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues*, Caen, France, 2011.

[74] D. Greene and P. Cunningham, "Practical solutions to the problem of diagonal dominance in kernel document clustering," in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006, pp. 377-384.

[75] S. Vijayarani, M. J. Ilamathi, and M. Nithya, "Preprocessing techniques for text mining-an overview," *International Journal of Computer Science & Communication Networks*, vol. 5, no. 1, pp. 7-16, 2015.

[76] S. Seifzadeh, A. K. Farahat, M. S. Kamel, and F. Karray, "Short-text clustering using statistical semantics," in *Proceedings of the 24th International Conference on World Wide Web*, Florence, Italy, 2015, pp. 805-810.

[77] M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge, "Twitter polarity classification with label propagation over lexical links and the follower graph," in *Proceedings of the 1st Workshop on Unsupervised Learning in NLP*, Stroudsburg, PA, 2011, pp. 53-63.

[78] S. Baillargeon, S. Halle, and C. Gagne, "Stream clustering of tweets," in *Proceedings of 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, San Francisco, CA, 2016, pp. 1256-1261.

Supavit Kongwudhikunakorn **https://orcid.org/0000-0003-0834-5209**

He received his B.Eng. (1st class) degree in Software and Knowledge Engineering from Kasetsart University in 2016. Later in 2018, he was conferred M.Eng. in Computer Engineering from the same university. Currently, he is pursuing his Ph.D. in Information Science and Technology at Vidyasirimedhi Institute of Science and Technology, Thailand. His current research interests include deep learning and data mining.

Kitsana Waiyamai **https://orcid.org/0000-0002-2970-9553**

He is an associate professor in Department of Computer Engineering, Kasetsart University. He received his Ph.D. degree in Computer Science from Universite de Clermont-Ferrand II, France in 1999. His current research interests include data mining data warehousing, decision support systems, and database management system.