International Journal of Data Science and Analytics (2024) 18:239–258
https://doi.org/10.1007/s41060-024-00540-x

REVIEW

A comprehensive and analytical review of text clustering techniques

Vivek Mehta[1] · **Mohit Agarwal**[1] · **Rohit Kumar Kaliyar**[1]

Received: 6 October 2023 / Accepted: 18 March 2024 / Published online: 8 April 2024

Abstract
Document clustering involves grouping together documents so that similar documents are grouped together in the same cluster and different documents in the different clusters. Clustering of documents is considered a fundamental problem in the field of text mining. With a high rise in textual content over the Internet in recent years makes this problem more and more challenging. For example, SpringerNature has alone published more than 67,000 articles in the last few years just on the topic of COVID-19. This high volume leads to the challenge of very high dimensionality in analyzing the textual datasets. In this review paper, several text clustering techniques are reviewed and analyzed theoretically as well as experimentally. The reviewed techniques range from traditional non-semantic to some state-of-the-art semantic text clustering techniques. The individual performances of these techniques are experimentally compared and analyzed on several datasets using different performance measures such as purity, Silhouette coefficient, and adjusted rand index. Additionally, significant research gaps are also presented to give the readers a direction for future research.

Keywords Text clustering · Semantic text clustering · Word Embeddings · Curse of dimensionality

1 Introduction

The rise in digital form of textual data has seen exponential growth in recent years. Organizations need a proper system in place to find meaningful insights from the available data. Whether it is social media analytics or it is cybercrime investigation, they all require some textual data analytics. For example, more than 67,000 new research articles only on COVID-19 have been published by SpringerNature [16]. More than 7800 peer-reviewed articles appeared on the Web of Science database just for the keyword search "coronavirus." Additionally, the expansion rate get doubled every 20 days [15]. So, it can be imagined how fast the textual data are getting accumulated in digital form. With so much available data, it is just impossible for a health official or a medical researcher to go through this data to make better decisions to halt the spread of disease.

In the analysis of large amounts of documents spread across multiple sites, document clustering has become increasingly significant. The difficult part is organizing the documents in a way that allows for better searching without adding a lot of extra cost and complexity. The Cluster Hypothesis is essential to the discussion of increased efficiency. It claims that relevant documents are more comparable to one another than non-related documents, and hence appear in similar clusters [45]. Relevant documents will be well differentiated from non-relevant documents if the cluster hypothesis holds true for a given document collection. Because it lacks some of the query terms, a related document may be placed low in a best-match search. Especially nowadays, an important application of clustering lies in text summarization which in turn has applications in various domains such as journalism for news summarization, book summarization, and legal data summarization. For example, in [44] text clustering has been successfully used to improve the summarization of lengthy documents by enhancing the document level scoring of each sentence.

The five major contributions of this review paper are as follows.

1. Various representation methods for text have been reviewed such as vector space model, lexical chains, and word embeddings.
2. Based on each representation method, techniques for clustering the text are reviewed such as partitioning-

✉ Vivek Mehta
vivekmehta27@gmail.com

[1] Bennett University, Greater Noida, Uttar Pradesh 201310, India

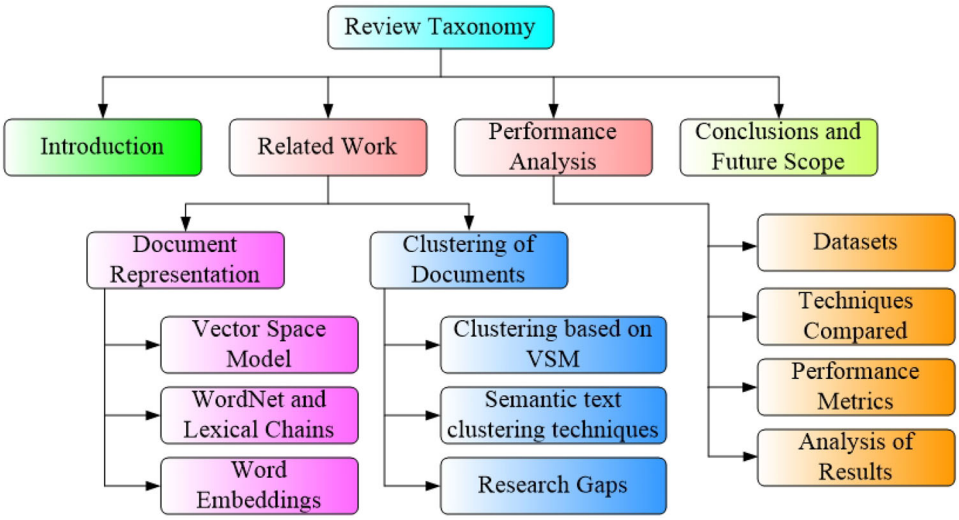

Fig. 1 Taxonomy of this review study

based, hierarchical-based, and other semantic clustering techniques.

3. Six independent factors have been identified which are synonymy, polysemy, dimensionality, language independency, and out of vocabulary (OOV) words based upon which several semantic text clustering techniques have been compared and analyzed.
4. Various research gaps have been analyzed and presented based on the review performed.
5. Experimental comparison and its analysis for semantic and non-semantic clustering techniques belonging to different categories are presented.

The organization of this research work is as follows. Section 2 contains the literature review which is subdivided into two subsections, text representation and text clustering. The first subsection consists of three different text representation paradigms, viz. vector space model, lexical chains, and word embeddings. The second subsection consists of clustering algorithms suitable for different text representations. Section 3 contains the performance analysis of various clustering methods presented in the literature review. Finally, the paper is concluded and the future scope is also presented. A diagrammatic overview of this document structure is shown in Fig. 1.

2 Research resources

To carry out literature review in this study, almost all the research papers have been taken from reputed publishers such as IEEE, Springer, ACM, and Science Direct. Additionally, various research papers from highly reputed conferences such as ACL, NEURIPS, EMNLP, KDD and KDDM are also included. These papers are counted in category "others" in Table 1. To include each of the papers counted in Table 1, a criterion has been followed which is mentioned in the following subsections.

Table 1 Distribution of research papers according to their resources

S.no	Digital library	Total papers
1	Elsevier	17
2	ACM	6
3	IEEE	6
4	Springer	9
5	Others (ACL, NEURIPS, EMNLP, etc.)	42

2.1 Inclusion criteria

A criteria for inclusion is established so that a thorough investigation could be conducted into limited number of papers that have been carefully chosen to satisfies the following requirements:

- IC 1: The article has extensive references and citations.
- IC 2: The article's scholarly credibility.
- IC 3:The article focuses on text clustering techniques.
- IC 4: The article talks about the use of innovative ideas such as lexical chains, word embeddings for text classification/clustering.

3 Literature review

The complete review is divided into two parts which include text representation and text clustering methods based on suitable representation. These are as follows.

Table 2 An example of a term-document matrix

	d_1	d_2
Mouse	1	0
Quicker	1	1
Cat	1	1
Dog	0	1

3.1 Text representation

A dataset containing a set of documents (also called a corpus) cannot be clustered in its raw form. Hence, the first step toward document clustering is representing the document dataset in a suitable form. The widely used representation for text is known as the vector space model. However, some more complex representations also exist in the relatively recent literature such as those using "lexical chains" and "word embeddings." Let us discuss these one by one.

3.1.1 Vector space model

In this representation, to perform the clustering of documents contained in a corpus, all the documents are converted into a numerical form. Generally, each document is converted into a numerical vector in which each numerical value reflects the importance of a word in deciding the category of document. It is also called a document vector. This representation of a document is also known as the bag-of-words (BOW) representation for a document. Based upon this representation, each document contained in the corpus is mapped to a common space of vectors. In a vector corresponding to a document, each vector component corresponds to a word contained in the vocabulary of the whole corpus. This set is generally formed by eliminating words of less importance such as stop words and other items such as punctuation and digits. This representation of a set of documents is known as vector space model (VSM). The output of this representation for a collection of N documents is an M x N matrix called as a term-document matrix having M words (also called dimensions) in vocabulary.

For example, let us assume that a corpus contains two documents d1 and d2 with the text "Mouse is quicker than cat" in d1 and "Cat is quicker than dog" in d2. Vocabulary formed from this set of two documents is the set mouse, quicker, cat, dog after eliminating stop words. A 4 term-document matrix resulting from this is shown in Table 2. The numerical values indexed at (i, j) are frequencies of the ith word in the jth document.

The pre-processing steps that are generally applied to get a better vocabulary of words from the whole corpus are as follows:

1. Punctuation removal: To perform discrimination among text documents, punctuation symbols like full stop (.), question mark (?), comma (,), semi-colon(;), colon (:), and quotation marks (" ") are not important. Hence, these are removed to make a process more accurate and computationally lighter.
2. Stop-words removal: A large number of words in any language are used to make the language grammatically correct but do not contribute to the actual theme of a text. These words include "the," "a," "and," "in," "of," "are," "that," "by," "for" and so on. These must be removed to compare any documents for their semantic aspect.
3. Tokenization: It is the process of converting a document (consisting of a large number of sentences concatenated together) into a list of words (or sentences). This is generally done for extracting features from any dataset on the basis of which any task such as categorization or classification can be performed.
4. Lowercasing: As the name suggests, it is the process of lowercasing each word in a document. This is generally done to reduce the size of corpus vocabulary.
5. Stemming: It is a process in which each word of a document is converted into its root word. For example, the three words *consist, consists, consisting* can be reduced only to the single word *consist*.

Term Scoring

In a document classification or clustering task, computing similarity between two documents is often done. As aforementioned, a document is represented as a vector that quantifies the importance of a word in that document. This quantification of the relative importance of each word can also be called *term scoring*. In a first attempt, term scoring can be done using the frequency of each term t occurring in a document d denoted by $TF(t, d)$. As two different words contribute differently to deciding the actual theme of a document, the term frequency approach suffers from the limitation of providing equal weights to every term. Hence, a mechanism is required that distinguishes the relative importance of each term in a document. For example, stop words occur too often in a document but contribute very little in deciding its similarity to another document.

For a term (or word) t, to give it a higher weight when it occurs in a few documents and a lower weight when it is frequent among all the documents, a scaling factor known as *inverse document frequency* [78] denoted by $IDF(t)$ is defined below.

$$IDF(t) = \log \frac{N}{df} \tag{1}$$

where N is the total number of documents in a corpus and df is the number of documents in which the term t occurs.

Now a weighting scheme known as $TF-IDF$ is defined by combining the term frequency and inverse document frequency that assigns a weight to a term t in a document d. It is given as follows.

$$TF - IDF = TF(t, d) \times IDF(t) \tag{2}$$

Thus $TF - IDF$ scoring mechanism gives the highest score to a term that is rare among documents but frequent within a document. Similarly, it assigns the lowest score to a term that is much frequent in a large number of documents in a collection of N documents. Other modifications in this scheme like $tf.rf$ where rf stands for $relevancefrequency$ are also defined in literature. [5, 53] presents a comparison of these kinds of schemes.

After a document is represented as a vector using the bag-of-words model and $TF - IDF$ scoring scheme, it is often required to calculate the similarity between two documents. Euclidean distance, for example, can be used as the simplest approach. However, it has a limitation that two documents that are very similar as per the semantics (meaning) of their content can have the Euclidean distance much higher. To overcome this limitation, cosine distance is generally used, which measures the cosine value of the angle between two document vectors. It is defined as follows for two documents d_1, d_2 having document vectors $\vec{d}_1, \vec{d}_2$ respectively.

$$sim(d_1, d_2) = \frac{\vec{d}_1.\vec{d}_2}{|\vec{d}_1||\vec{d}_2|} \tag{3}$$

Numerator in Eq. 3 is the dot product between the document vectors $\vec{d}_1, \vec{d}_2$ which is given by $\sum_{i=1}^{m} d_{1i}d_{2i}$, where m is the number of components in each vector. Denominator in Eq. 3 denotes the product of magnitude of each vector, which is given as $\sqrt{\sum_{i=1}^{m} x_i^2}$ for any vector $\vec{x}$. The denominator helps to normalize each document vector to unit length so as to neutralize the effect of document length on similarity.

As a final step in text clustering, any clustering algorithm shown in Fig. 4 is applied to a term-document matrix formed using statistical features like term frequency, $TF - IDF$ scoring, or any other suitable scoring mechanism. In [88], an experimental comparison of clustering algorithms for text data had been performed in which it came out that bisecting K-means performs better than K-means and hierarchical clustering algorithms. However, this kind of approach does not incorporate semantic features or characteristics of textual data. As a result of which many issues such as word sense disambiguation, polysemy and synonymy are not taken into account. In the next subsection, techniques involving semantic features of the text are reviewed.

3.1.2 WordNet and lexical chains

To discover semantic relations between different words in a text, an openly available[1] hierarchical database known as WordNet is generally used in various text processing tasks.

WordNet [65] is a database in which English nouns, verbs, adjectives, and adverbs are organized into sets of synonyms called synsets. These synsets are in turn linked through semantic relations such as synonymy (words with similar sense), Antonymy (words with opposite sense), and meronymy (part of). Table 3 shows different semantic relations in WordNet. In WordNet 2.1, there are a total of 155,327 words which are organized into 175,979 synsets. For each synset, WordNet provides a definition known as gloss and a set of example sentences. In papers such as [39, 81], document categorization is performed using techniques that used a WordNet based word sense disambiguation strategy. Specifically, in [39], WordNet was used to analyze three different settings in the VSM model which are: i. Disambiguation using part-of-speech tagging, ii. Including synonyms and iii. Including hypernyms. In another paper, WordNet was used to reduce dimensionality. [77]. It did so by mapping each word to one of the 41 lexical categories provided in the WordNet as the feature vector. Then in [28], a more complex Word Sense Disambiguation strategy was used and only nouns were used for building the feature vector. As a result, it was concluded that disambiguating polysemous nouns and synonyms gives better clustering results.

In addition to WordNet, lexical chains which originally were used for text summarization systems, have been used for text clustering as well [47]. A lexical chain is a group of related words which in turn help in identifying the topic and content of the document. In initial works, lexical chains were used to determine the test structure to identify its semantics. For example in [66], lexical chains computed using a thesaurus are used to identify the text structure. In a different paper [36], lexical chains are used to compute malapropisms (existing words with the same sound as the correct word but with a different meaning and which generally go undetected by a spell checker.). Lexical chains were also used in some papers for text clustering such as in [47, 48, 93].

3.1.3 Word embeddings

Representation of a word is a very trivial task required in almost all Natural Language Processing (NLP) tasks. Conventionally, given a fixed size vocabulary $V = w_1, w_2, w_3, \ldots, w_n$, each word w is represented as a vector $\vec{w}$ of length of vocabulary size $|V|$. Each index of this vector corresponds to a word in the vocabulary, thus for a word only one component

[1] https://wordnet.princeton.edu/.

Table 3 Semantic relations in WordNet

Relation Name	Syntactic category	Examples
Synonymy (similar)	Noun, Verb, Adjective, Adverb	(document, papers), (cluster, bunch)
Antonymy (opposing)	Noun, Verb, Adjective, Adverb	(tall, short), (light, dark)
Hyponymy (is-a relation)	Noun	(cluster, knot), (document, certificate)
Meronymy (part-of)	Noun	(car, engine), (tree, trunk)
Troponymy (manner-name)	Verb	(talk, orate), (eat, slurp)
Entailment	Verb	(eat, chew, masticate)

value is 1 and all others are zeros in its vector. Mathematically, it can be given

$$\vec{w_i} = \begin{cases} 1, & \text{if } w = w_i \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

This representation is called the *one-hot* representation and has been used widely in NLP tasks due to its simplicity. However, it has some drawbacks, firstly, it is high dimensional and sparse representation because most of the values remain zero. Secondly, it does not capture semantic relatedness among words. Thirdly, if new words get added to the dictionary, the size of the vector gets increases for each word, thus it is not flexible [56].

To overcome these limitations of one-hot representation and also to provide a mechanism by which words can be compared among themselves, a *distributed representation* of a word known as *word embeddings* was introduced by Tomas Mikolov in 2013 at Google [63]. Here, distributed representation means that the semantics is based on the context of the term, a word is mapped to a vector with continuous real values. This is based on the distributional hypothesis [26, 33] which says that words with similar contexts are semantically similar to each other. This representation is also $dense$, meaning that more than one dimension of the vector represents the semantics of a word. Additionally, this representation is flexible contrary to one-hot representation because the length of the vector for all words remains fixed even if a new word is added to the vocabulary. Formally, word embedding can be defined as "a dense, distributed and fixed-length vector used to represent the semantics of a word using its context."

The goal of optimizing an objective function for the production of word embeddings is to maximize the probability of a central word in a context window of fixed size m (in the case of Word2Vec algorithm [63]). This is accomplished by using a large corpus of text to train a neural network design. A numerical vector (or embedding) corresponding to a word is the output of the network design. Other word embedding algorithms include Stanford University's Glove [75]and Facebook's FastText (citebojanowski2017enriching). These are all open-source projects that may be downloaded for free [2,3]. Several recent studies present a good survey on various algorithms used to generate word embeddings [4, 8, 17, 91]. The dimensionality of this vector generally lies from one hundred to one thousand. Several algorithms proposed in the literature since 2013 are described below.

Word2Vec

Given a vast corpus of text in the form of a word sequence, Word2Vec finds word vectors (embeddings) such that they can best predict the context words $w(t+j)$ for a given center word $w(t)$ in a window to fixed-size m. This idea is represented mathematically in the form of an objective function given in the following equation [91].

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m \\ j\neq 0}} \log P(w_{t+j}|w_t;\theta) \tag{5}$$

where,

$$P(w_{t+j}|w_t;\theta) = \frac{\exp(\mu_o^T v_c)}{\sum_{w\in V}\exp(\mu_w^T v_c)} \tag{6}$$

The objective function in Eq. 5 is the average negative log-likelihood function in which θ denotes all the variables to be optimized. This optimization is generally done by minimizing it using the stochastic gradient descent function [13]. The probability of occurrence of a context word $w(t+j)$ given a center word $w(t)$ is given as shown in Eq. 6. The numerator in Eq. 6 contains the exponential value of the dot product of word vectors of a context word and the center word. This value is normalized (in the denominator) by summation of exponential values of the dot product of center word vector v_c and each word vector μ_w in the entire vocabulary V. However, the calculation of denominator becomes a computation process very slow; hence, the process of $negativesampling$ [64] is applied in which a modified objective function to make the process efficient. This type of model of Word2Vec is known as *Skip-gram* model. In the original paper, [63],

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

[3] https://fasttext.cc/.

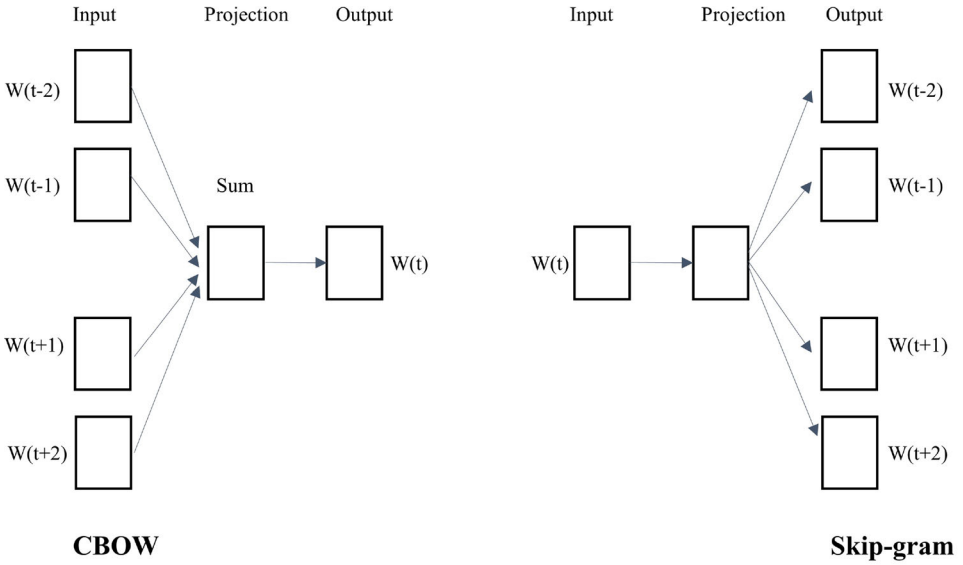

Fig. 2 Architectures of Word2Vec model [63]

the second variant of Word2Vec was also proposed which is known as the continuous bag-of-words (CBOW) model. The objective function in the CBOW model attempts to maximize the likelihood of a target center word given the context words. Figure 2 shows the two models of the Word2Vec algorithm.

Global Vectors (GloVe)

As compared to the one-hot representation of words, the Word2Vec model provides vectors of much smaller size (300) and better captures the semantics and relationship between words. However, Pennington et. al [75] observed that Word2Vec can capture information only from a local context of words ignoring the information from a global context. Hence, they proposed a different model for word vectors named *Global Vectors* (GloVe) [75]. In GloVe, information from the global context is used with the help of the global co-occurrence matrix X in which each element X_{ij} represents the co-occurrence frequency of the i^{th} word and j^{th} word. This observation is formulated into the following objective function to calculate word vectors.

$$J = \sum_{i,j} f(X_{ij})(v_i^T v_j + b_i + b_j - log X_{ij}) \tag{7}$$

In above equation [75], v_i and v_j are word vectors corresponding to i^{th} and j^{th} word, respectively, which are to be learned. b_i and b_j are word-specific biases that also need to be learned. All these parameters are learned by minimizing J for all these parameters. Function $f(x)$ is used as a weight function to not over-weight very frequent words and rare words. It is defined as follows.

$$f(x) = \begin{cases} (\frac{x}{x_{\max}})^{\alpha}, & \text{if } x < x_{\max} \\ 1, & \text{if } x \geq x_{\max}. \end{cases} \tag{8}$$

where $x_{\max}$ (the upper cutoff for co-occurrence frequency) and α can be tuned for a given dataset.

FastText

Word2Vec and GloVe models fail to provide word vectors for words that are out of vocabulary (OOV). This refers to words that aren't found in the training dataset. A model for word representation and text classification problems was introduced in the article [12] and made publicly available as a library named fastText.[4] The architecture of the model is similar to Word2Vec but uses subwords (or character n-grams) to find vector representations and related words for queries of OOV words. For example, the word "gearshift" which may not exist on Wikipedia is broken down into subwords which are all the substrings contained in it like "gea," "ear," "rsh," and "shi." The length of subwords is controlled using *minn* and *maxn* parameters. Each of these subwords gets a vector corresponding to them and the final vector for the complete word is achieved by summing up the vectors of these subwords. In comparison with Word2Vec and GloVe models, fastText is fast in its training on large corpora for classification tasks such as tag prediction and sentiment analysis [12, 49].

Bidirectional Encoders Representations using Transformers (BERT)

One of the major challenges in the field of Natural Language Processing is to represent words and sentences in some form so that they can be compared to each other based on the context in which they appear. Late 2018 saw a milestone achievement in this area with the introduction of BERT.

BERT is a Transformer [90] based deep learning model that has set new benchmarks in language-based tasks like sentiment analysis next sentence prediction, question answering,

[4] https://github.com/facebookresearch/fastText.

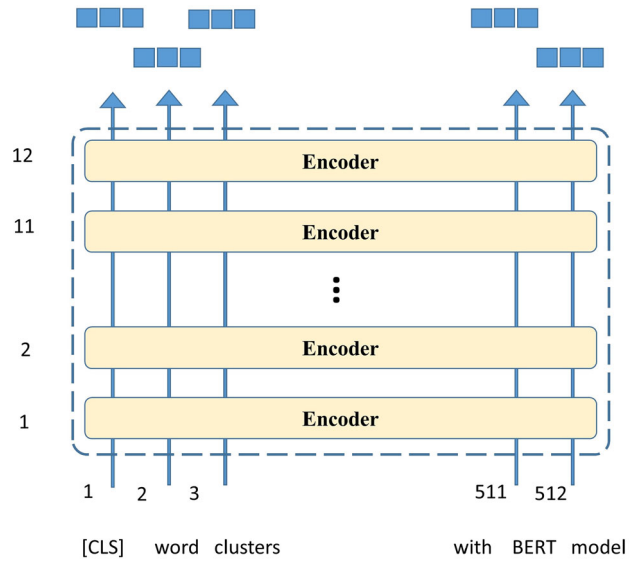

Fig. 3 Architecture of BERT [3]

and Named Entity Recognition (NER) [21]. Various versions of BERT were open-sourced. [5] These versions have already been pre-trained on the massive book and Wikipedia datasets. As a result, these models can be used as-is or fine-tuned for the various supervised tasks outlined before.

Figure 3 depicts the BERT model's high-level architecture. It is basically a stack of transformer (encoder) layers stacked on top of each other. The original paper offered two architectures, $\text{BERT}_{\text{base}}$ and $\text{BERT}_{\text{large}}$, with 12 and 24 encoder layers, respectively. The model receives a string of words as input, the first of which is the special token "[CLS]." The input sequence might be as short as 1 and as long as 512 characters. Each BERT encoder layer produces a vector that is fed into the layer above it as input.

The $\text{BERT}^{\text{base}}$ and $\text{BERT}_{\text{large}}$ models produce a vector of length 768 and 1024 for each individual word of the input sequence, respectively. The semantics and relationships between them are encoded in these vectors. These vectors can be utilized for a variety of supervised downstream applications, including question answering and sentiment analysis. This is usually accomplished by including an additional neural network layer at the end of the model, as well as a softmax function. In comparison with other state-of-the-art models, the original research offers remarkable outcomes for these types of tasks.

3.2 Text clustering

3.2.1 Clustering based on VSM

Once a numerical representation of the text corpus is achieved using any of the aforementioned techniques, conventional clustering techniques can be applied to get the document clusters. In this section, a brief description of various clustering algorithm categories along with their advantages and disadvantages is provided.

Classification of clustering techniques Clustering algorithms can be broadly classified as shown in Fig. 4 [32]. Various categories are briefly described below:

(a) **Partitioning based**
Suppose n objects are present in a dataset D. These n objects can be divided into k clusters $C_1, C_2, \ldots, C_k$ (where $k \leq n$) using partitioning methods. The obtained partitions should satisfy these conditions: (1) None of the clusters should be empty and (2) every object must be a part of anyone cluster for Hard c-means or multiple clusters for Fuzzy c-means. Initially, random or some more complex methods are used to choose k cluster centers, subsequently, a migration approach is utilized to move the cluster centers toward a better precise solution. For example, (1) in K-means [58], all the cluster's data points average is employed to calculate the new cluster center here (2) in k-medoids [72] a cluster is represented by an object which is placed close to the center of that cluster. An objective function is employed to calculate the quality of the clustering approach. The design of this objective function is such that it achieves low intercluster similarity and high intracluster similarity. Other popular algorithms present in this category are: K-modes [41], CLARA [71], FCM [10] and PAM [70].

(b) **Hierarchical based**
These methods can be classified as divisive and agglomerative, in which a hierarchical breakdown of a dataset is made. The agglomerative clustering approach utilizes every object as a cluster and then progressively merged till they satisfy a cessation condition. The contentious clustering strategy, on the other hand, treats the entire set of items as a single large cluster that is gradually split down into smaller clusters until it meets a termination constraint. The agglomerative approach is also known as the bottom-up method whereas the divergent approach is called the top-down method. The agglomerative clustering algorithm can be described as follows.

[5] https://github.com/google-research/bert.

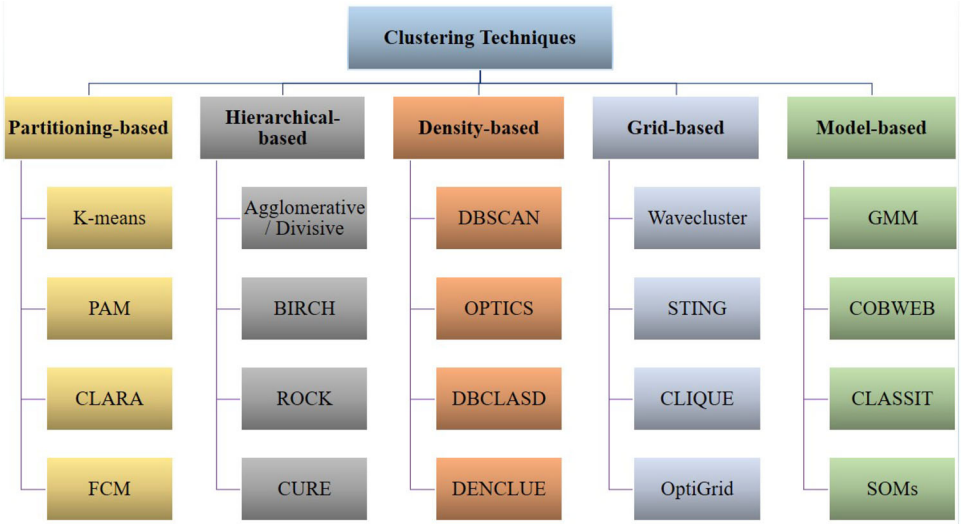

Fig. 4 Broad classification of clustering algorithms

Agglomerative clustering algorithm

(a) Let every data point be assumed as a cluster on its own.
(b) The vicinity matrix of individual points is calculated.
(c) Two nearest clusters are merged and then the vicinity matrix is updated.
(d) Repeat step (c) until a one cluster is formed.

Hierarchical clustering algorithms create the familiar tree-like structure known as a *dendogram* as the output. This *dendogram* can be broken at various levels to obtain corresponding varying data clusterings. Depending on the definition of inter-cluster similarity, three main agglomerative hierarchical clustering approaches are *single-linkage, average linkage* and *complete linkage*. Distance between the *closest* pair of data points in clusters is used as a measure of inter-cluster similarity in single linkage algorithms. Distance between the *group average* of all data points present in a cluster is used as the proximity measure between clusters for average linkage algorithms. Whereas the distance between the *afar* pair of data points is used as the inter-cluster similarity in complete linkage algorithms. A sample dendogram created using seven data points for single-linkage clustering is shown in Fig. 5 [43]. BIRCH [97], ROCK [30], CURE [31] and Chameleon [50] are few more popular hierarchical algorithms.

(c) **Density based**

Methods described above find it difficult to cluster when they are of arbitrary shape [32] as they find similarity based on a proximity measure. On the other hand, density-based methods can find clusters of random shapes as they find clusters depending on density. Here, a cluster keeps growing till the count of data objects in the vicinity surpasses some baseline value. Density at any data point p is defined as the number of data points within a circle of radius *eps* around p as a center. A cluster inside a circle of radius *eps* is called a dense region if it has some least number of data points *minpts*. A point with a dense circular region around it is called a *key point*. Similarly, a border point is a point that itself is inside the vicinity of a key point but which has points lesser than *minpts* surrounding it. Lastly, a point that cannot be called a key point or a boundary point is defined as a noise point. A basic density-based algorithm DBSCAN [25] can be described as follows [80].

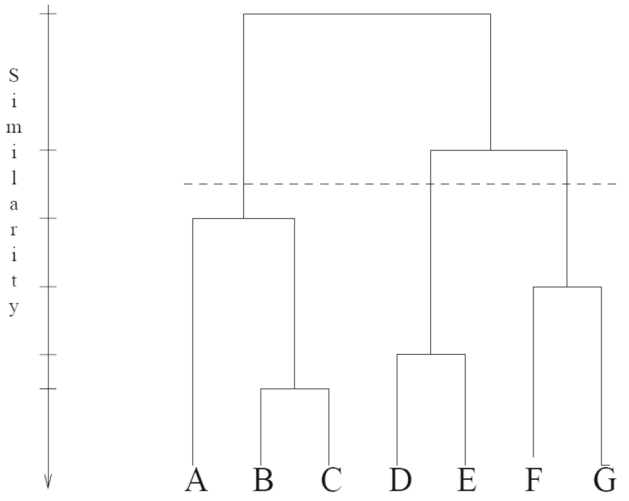

Fig. 5 A simple dendogram depicting hierarchy of clusters on 7 points [43]

DBSCAN algorithm

(a) All core points are identified.
(b) Neighboring key points are assigned to one cluster.

(c) **For every** non-key point **do**

Assign that point as a boundary point to the nearest core point cluster if possible else mark it as a noise point.
OPTICS [6], DBCLASD [95] and DENCLUE [35] are other algorithms of this category.

(d) **Grid based**
Here, a grid structure is formed between dimensions by dividing each dimension into several cells. On this quantized space clustering operations are then performed. The processing time of these methods does not depend on the number of objects. Rather it is determined by the number of cells in the lattice structure. Few well-known examples of this category are STING [92], Wavecluster [83], CLIQUE [43] and OptiGrid [34].
One of the early algorithms in this category (CLIQUE) can be described as follows [32].

(a) The d-dimensional data space is divided into non-overlaying rectangular cells (or units) and dense components in all subspaces are identified based on a density baseline l.
(b) In each subspace dense cells are then used to form clusters. This is done by beginning with a random dense cell followed by finding the maximal region covering the cell and operating on the rest of the dense cells.

In [2] its more elaborate details can be found.

(e) **Model based**
Clustering is done in these methods by first hypothesizing a mathematical model which follows a step of finding its closest fit for a given dataset. The EM algorithm, for example, performs an expectation-maximization analysis [20], A probability analysis and a neural network-based method is performed by COBWEB [27], similarly high-dimensional data are mapped onto a 2-D or 3-D feature map using self-organizing maps (SOMs) [52] to perform clustering. CLASSIT [29] algorithm extends COBWEB for continuous-valued data.
Table 4 summarizes the advantages and disadvantages of the above-mentioned clustering algorithms which are determined from the literature. A performance comparison of these algorithms according to the time taken for cluster formation and the size of datasets is presented in [82]. A comparison of the performance of partitioning algorithms such as K-medoids and K-means, based on different proximity measures was done in [85]. However, the data of a much larger dimension (such as textual data) was not covered. Experiments to review and contrast the performance of different clustering algorithms of various categories are presented (Sect. 5) in this study. A substantially wide range of data dimensionality is covered in these experiments. Partitioning-based FCM algorithm is reviewed in the next subsection especially for proximity measures after having reviewed vicinity measures in Sect. 2 and clustering algorithms in Sect. 3.1.

3.2.2 Semantic text clustering techniques

Apart from the clustering techniques mentioned in Sect. 3.2.1 which are directly applicable to the VSM representation of text, various semantic text clustering techniques have been defined in the literature. Most of these techniques exploit WordNet to extract the semantics of text, while some use the Wikipedia database as well. A few other techniques use some domain-specific ontology such as defined by L.Yue et al. [96]. A complete summary based on several important factors identified from the related research papers is provided in Table 5. Important fields used in Table 5 are defined below.

1. Solves synonymy
   A checkmark (✓) indicates that the technique uses the synonymy relation between words to be more efficient, while a cross mark (✕) indicates that it does not use synonymy relation.
2. Solves polysemy
   A checkmark (✓) indicates that the technique uses the polysemy relation between words to tackle the ambiguity problem, while a cross mark (✕) indicates the non-usage of the same.
3. Semantic source
   This field tells the usage of any semantic source (Wikipedia, WordNet, etc.) by the technique.
4. Dimensionality
   This field tells about the actual dimensionality used by the technique for clustering the documents. The "High" value implies no mechanism is used to reduce the dimensionality, while the "Medium" value implies some mechanism has been used to reduce the dimensionality.
5. Language independency
   In this field, a checkmark (✓) indicates that the technique is applicable for any language dataset while a cross mark (✕) indicates that it is for a specific language only (mostly English).
6. OOV words
   This field tells whether the technique is capable of handling the out of vocabulary (OOV) words. OOV words are those words that are not contained in the vocabulary of the semantic source used.

A more detailed review of each of the works mentioned in Table 5 is as follows.

Table 4 Advantages and disadvantages of different types of clustering techniques

Clustering technique	Advantages	Disadvantages
Partitioning based	Algorithms are robust to noise and converge faster	They generally produce only convex clusters and are difficult to work with ordinal/minimal attributes
Hierarchical based	The number of clusters is not needed and can find non-convex clusters	It needs large memory space for large datasets and is more time taking and fails in the presence of noise
Density-based	Has the capability to treat outliers as noise robustly and can discover arbitrary shaped clusters also does not require the number of clusters	Appropriate selection of parameters highly decides the quality of clustering. Algorithms do not perform well in case of varying density (in the case of DBSCAN). Curse of dimensionality is also present in these algorithms
Grid-based	High dimensional datasets suit these algorithms more. There is no negative effect based on the order of input of records	Size and number of grid cells highly decide the clustering quality
Model-based	Appropriate statistical models can be chosen to capture latent clusters. Explicit assignment of data points is not present instead they have a probability of being part of a number of clusters	If there are a large number of distributions then EM algorithm can be considerably expensive and the algorithm also does not ensure to settle on a global optimum (this can be a more serious worry in high dimensions)

Patil and Atique [74] have demonstrated how key terms from a document could be retrieved for documents clustering. The process starts by removing stop words, followed by stemming using the Porter Stemming algorithm. In the next step, Wordnet is used to assign English lexical categories such as nouns and verbs. to each stemmed word. Finally feature selection methods such as tf2, tf-df, and tf-idf are used to find key terms based on a threshold value. Authors found that tf-idf is a better way of dimensionality reduction by finding key terms without much data loss for efficient document clustering. Li et al. [54] have used the BERT model to generate sentence embedding based on context then they have used two methods WA (Weighted Average) and WR (Weighted Removal) for weighing scheme to organize sentence embedding. Finally, k-means clustering is used to give better performance as compared to previous clustering methods. Park et al. [73] have shown that state-of-the-art contextualized document vector creation methods for extracting syntactic and semantic features of documents can be used for efficient document clustering. Authors experimented using cosine similarity and Euclidean distance measure for comparing contextualized vectors and update cluster centroids using mini-batches of the dataset. Authors showed that the proposed method can achieve 85.11% accuracy for SQuAD 1.1 dataset for clustering which is better than previous methods of k-means using TF-IDF, GMM, BIRCH, and DEC.

Shi and Wang [84] have proposed two methods for document clustering SCL (self-supervised contrastive learning) and FCL (few-shot contrastive learning). Authors tuned BERT model using their learning methods and used for clustering. Authors used back translation and random masking augmentation methods with SCL and found that it outperformed all state-of-the-art methods in accuracy and NMI for document clustering. FCL was also able to achieve performance close to supervised methods and along with unsupervised data augmentation(UDA) authors achieved further improvement in clustering short texts. Sinoara et al. [86] proposed two semantic representations of document collection NASARI+Babel2Vec and Babel2Vec. These use neighboring work sense to obtain more interpretable document information. The authors compared the performance with BOW, LDA, and Word2Vec text classification methods. Using Macro averaging and micro averaging for calculating F1-score they found that Babel2Vec was second-ranked for Micro F1-score and first ranked for Macro F1-score.

Elhadad et al. [24] have used an ontology-based method to reduce the dimensionality of the feature set of documents. The words which do not have any lexical meaning according to WordNet lexical categories are removed and the tf-idf and vector space model is used for text mining. Authors show that the proposed ontology based method achieves high accuracy and F1-score as compared to PCA based feature reduction for document classification. The authors calculated the results using the Reuters-21578 dataset.

Abbasi-Moud et al. [1] have proposed a tourism recommendation system by performing semantic clustering and sentiment analysis of user reviews of tourism attractions. In the next step, nearby attractions are ranked based on the user's preference similarity and contextual information such as location, time, and weather. Sentiment analysis could help gather user preferences and contextual information could also enhance recommendations. Authors show a higher F1 score for giving tourism recommendations as compared to previous similar previous studies.

Wei et al. [93] have used WordNet for differentiating words with similar meanings and lexical chaining to extract

Table 5 Summarized analysis of various semantic text clustering techniques

S.no	Authors	Solves Synonymy	Solves Polysemy	Semantic source	Dimensionality	Language independency	OOV words	References
1	Mehta et al. (2021)	✓	✓	BERT embeddings	Low	✓	✓	[62]
1	Wei et al. (2015)	✓	✓	WordNet	Medium	×	×	[93]
2	Li et al. (2015)	×	×	Nil	High	✓	×	[55]
3	Yue et al. (2015)	✓	×	Dairy Ontology	Medium	×	×	[96]
4	Nasir et al. (2013)	✓	×	WordNet/ Wikipedia	Medium	×	×	[69]
5	Bouras et al. (2012)	✓	×	WordNet	Medium	×	×	[14]
6	Fodeh et al. (2011)	✓	✓	WordNet	Medium	×	×	[28]
7	Huang et al	✓	✓	Wikipedia	High	×	×	[40]
8	Baghel et al. (2010)	✓	×	WordNet	Medium	×	×	[7]
9	Luo et al. (2009)	×	×	Nil	High	✓	×	[57]
10	Recupero et al. (2007)	✓	×	WordNet/ ANNIE	Medium	×	×	[77]
11	Jayarajan et al. (2007)	✓	×	WordNet	Medium	×	×	[47]
12	Patil and Atique (2013)	×	×	WordNet	Medium	×	×	[74]
13	Park et al. (2019)	✓	✓	Contextual embeddings	Medium	✓	✓	[73]
14	Li et al. (2020)	✓	✓	BERT embeddings	Medium	✓	✓	[54]
15	Shi and Wang (2020)	✓	✓	BERT embeddings	Medium	✓	✓	[84]
16	Abbasi-Moud et al. (2021)	×	×	WordNet	High	×	×	[1]

main semantic features. Authors used these techniques for dimensionality reduction, clustering, and assigning topics to the clusters generated. Authors experimented on the Reuters-21578 dataset by sampling subsets based on several documents in a class and obtained an F1-score of 0.728 on such a subset made of the 3 largest classes of the Reuters-21578 dataset.

Li et al. [55] have created a parallel k-means clustering of documents based on neighbors. Neighbors are calculated based on the tf-idf vector of documents and cosine similarity, when this similarity is more than a threshold then it is taken as a neighbor document. The parallel algorithm takes advantage of parallelism in 3 steps calculating neighbors and assigning initial centroids based on ranks and finally updating and assigning documents to clusters in a loop. A one to thirteen nodes Linux cluster was used to test the performance and it was found speedup was almost linear with an increase in the number of processors. Yue et al. [96] have used domain ontology to find low dimensionality feature vectors. Next, they transform the vector space model to concept space using singular value decomposition (SVD). Finally, fuzzy equivalence is used to cluster documents based on max-min transitive closure. Thus, authors could achieve both dimensionality reduction and better clustering results.

Nasir et al. [69] have created a document clustering model named S-VSM (semantically smoothened Vector Space Kernel) and showed its performance is better than VSM (Vector Space Model). In this, the authors used semantic relatedness of words for smoothing document similarity. Authors used 3 types of similarity, one based on WordNet, one on Wikipedia, and one based on corpus-based similarity. Authors also demonstrated that a variant of S-VSM in which top k semantically related terms were used can have better performance as compared to VSM with efficient time and space requirements. Bouras and Tsogkas [14] have used a bag-of-words frequency to compare the similarity of documents based on Euclidean distance and cosine similarity. Further using k-means clustering authors found that better results were obtained as compared to hierarchical clustering. Authors have also used a WordNet-based k-means clustering in which a hypernym graph is created for keywords in a document and 20% most important keywords are extracted from each document to reduce the dimensionality and then k-means clustering is used for documents clustering. It was reported that it could give 10 times better results than normal k-means clustering. Fodeh et al. [28] have shown that clustering based on words identified as nouns using WordNet can give good results similar to word sense disambiguation (WSD). Authors also showed that polysemous and synonymous words are important in a document and using Information Gain by the unsupervised method they can help to identify core semantic features and reduce dimensionality by 90%. This process of dimensionality reduction can maintain or improve clustering performance.

Huang et al. [40] have used a bag of concepts (BOC) using Wikipedia to cluster documents based on Euclidean distance and cosine similarity. Authors found that using two datasets Reuters-21578 and OHSUMed improved F1-score was obtained as compared to other clustering methods such as bag of words (BOW). Baghel and Dhir [7] have shown frequent concepts- based document clustering (FCDC) which uses WordNet-based ontology which uses semantic relations of words and helps to create a low dimensional vector that can give more accurate clustering results than other methods such as bisecting k-means, UPGMA and FIHC. Luo et al. [57] have used tf-idf vectors cosine similarity and a threshold to form a neighborhood matrix. Authors have used a new way to find initial centroids of clusters using the rank of documents. Authors demonstrated that this way of document clustering can provide improved results as compared to k-means and bisecting k-means clustering.

Recupero [77] have used WordNet lexical categories (WLC) and WordNet Ontology (WO) to create low and more structured vector space models. In the next step, the author has used bisecting k-means and Multipole tree methods for document clustering. The author found that WO gave the best results but it was more computationally expensive as compared to WLC. Jayarajan et al. [47] have shown that traditional bag of words do not use semantic features of a document and have used lexical chaining to reduce the dimensionality of a document tremendously and provides reduced time and space complexity with similar or better performance.

Apart from the above mentioned research works, several other recent research works have emerged, for example, Cecchini et al. [18] have proposed a new graph-based word sense induction (WSI) method on two pseudoword ego word graph datasets. Authors have compared the proposed model performance with six WSI methods: Markov Cluster Algorithm (MCL), Chinese Whispers (CW), MaxMax (MM), Gangplank clustering (GP), Aggregative clustering (AGG), Curvature-based clustering (CURV), and HyperLex(HL). The performance was compared by three measures: normalized mutual information (NMI), BCubed F-measure, and a newly defined measure TOP2.

Roul [79] have proposed a novel feature selection method that generates term and document feature vectors and then merges these vectors. A novel web document clustering technique is discussed which uses latent semantic indexing (LSI) and the min-cut algorithm of graph theory. Finally set of important topics is generated and inverse-mapped to the documents which are ranked based on semantic space weights. Jasinska et al. [46] have proposed clustering of open heart failure datasets using k-means clustering by feature selection in two ways. The first was domain based in which features

are selected by careful recommendation of doctors. The second was done using Principal Component Analysis (PCA) which automatically selects important features based on the PCA algorithm. It was found that the second approach gave better results as it was free from human bias and also gave insights into features that were considered unimportant by physicians.

Hosseini and Varzaneh [38] have proposed a new BERT model for representing text called ParsBERT, usage of Autoencoder for providing robust features for providing to the k-means clustering algorithm. The authors evaluated the performance on the Barez unlabeled dataset and Silhouette Score is used to evaluate documents clustering into seven topics. It was found that the proposed method outperforms the existing clustering methods.Urkude and Pandey [89] have proposed an ontology-based clustering of the BBCSports dataset and compared it with traditional K-means clustering and density-based clustering (DBC) methods. Authors developed an ontology for the sports domain and found that the ontology-based method outperforms both traditional methods as they were based on a collection of characters and the ontology-based method considered the semantics of the words.

Mustafi and Mustafi [67] have proposed the usage of the differential evolution (DE) algorithm for clustering of a large corpus. The fitness function used was the ratio of intra-cluster distance and summation of inter-cluster distances. After this step, the authors re-evaluated the points lying on the fringes of clusters and reassigned them to different clusters as necessary. The result of these two steps on a number of standard corpora demonstrated that the results were highly accurate. Dey et al. [22] have shown the usage of quantum-inspired differential evolution for clustering 'of size real-life public datasets. Authors found that by using some concepts of quantum gates they could obtain better results than four other variants of quantum and DE algorithms in terms of standard error and convergence time.

Naik et al. [68] have clustered Twitter data based on graph structure generated by Twitter data and content of Tweets. Individually these methods fail to generate effective clusters in large datasets such as Twitter. Thus a pipeline-based method of structural clustering followed by content-based clustering was evaluated and found to give better results than existing clustering techniques.

Based on this comprehensive literature review, important research gaps are mentioned in the next subsection.

3.2.3 Research gaps

Based on the literature review presented in previous sections, the following research gaps have been found in text clustering techniques.

1. **High dimensionality and sparsity**
   Vectors resulting from the BOW model lead to a very high-dimensional representation of documents, for example, it may reach in order of 10^5 for some thousands of documents. This also leads to a very sparse term-document matrix that affects the accuracy of clustering techniques [9, 87]. Techniques such as *latent semantic indexing* (LSI) [19] and topic modeling [11, 37] reduces the original space to a lower-dimensional space. However, dimensionality reduction techniques exploiting more recent distributed [64] representation of words do not exist.
2. **Semantic ambiguities**
   Dealing with problems such as synonymy (different words with the same meaning) and polysemy (the same word with a different meaning) often decreases the accuracy of text clustering. Various approaches have explored methods to derive semantic relations using WordNet ontology [39, 81, 93]. However, these approaches are highly dependent on word coverage and the design of WordNet [65]. Additionally, these approaches are mainly useful for only a few languages. Hence, text clustering methods covering these limitations are required.
3. **Use of distributed representation of words**
   Almost all the text clustering techniques use a Bag of Words model in which every single word is treated as a dimension. As a result, the term-document matrix becomes very sparse. However, with the invention of word embeddings in which a word is represented with a dense vector, semantic relations between words can use to combine the words around a single topic. This area of research has not been much explored especially for text clustering tasks except for only a few research studies [23, 51].
4. **Clustering of large text datasets**
   When the number of the document becomes very large in a corpus, above mentioned problems like high dimensionality, sparsity, and semantic ambiguities, etc. become even more challenging for a text clustering task. For large datasets, efficient clustering methods that can use recently proposed context-based dense representations (based on deep learning) such as embeddings from language models (ElMo) [76] and BERT [21] are required. These types of embeddings can capture the semantics of a word based on its context.

4 Performance analysis

In this section, an analysis of five different types of text clustering techniques is presented based on three different

performance metrics. The performance is analyzed on seven datasets that are of different sizes and domains. The results presented in tabular formats in this section majorly form a subset of those covered in one of our previous works [62]. Other details are as follows:

4.1 Datasets used

To validate the performance of different clustering algorithms experimentally, the following datasets have been used.

i. Articles-253
The Articles[6] dataset consists of 253 research articles of 5 different categories which are as follows:
   a. Mobile Computing
   b. Political Science
   c. Weather
   d. Food
   e. DNA

   Each document consists of a title, abstract, and references.
ii. Scopus
This dataset[6] consists of 500 articles taken from Scopus database. Each document belong to one of the following 5 categories.
   a. Concrete
   b. Hyperactivity
   c. Investment
   d. Photosynthesis
   e. Tectonicplates
iii. 20NG
This is a subset of a widely used 20 Newsgroups dataset[7] which consists of news articles of 20 different categories. The subset used in our experiments consists of 5 different non-overlapping categories namely:
   a. Alt.atheism
   b. Talk.religion.misc
   c. Comp.graphics
   d. Sci.space

   The total number of documents are 700.
iv. Classic4
This corpus[6] consists of 800 documents which belong to the following 4 categories.
   a. Aerodynamics
   b. Medical
   c. Computing algorithms
   d. Information Retrieval
v. Scopus-long
This corpus is a collection of 2800 research articles from Scopus database[6]. There are 7 categories in this dataset each of which contains 400 articles. The categories are:
   a. Investment
   b. Neural Network
   c. Hyperactivity
   d. Concrete
   e. Proton
   f. Photosynthesis
   g. TectonicPlates
vi. Classic4-long
This is an extended dataset of Classic4[6] which contains 3891 documents divided into 4 categories.
vii. 20NG-long
This is also a large part of the 20 Newsgroups dataset[7] containing 8131 documents. These documents belong to 9 different categories.

4.2 Techniques compared

To perform an exhaustive analysis of different text clustering techniques, one from each of the following categories has been considered.

i. Embedding-based
In this category, a recently proposed text clustering technique based on BERT word embeddings is included [62]. This technique aims to solve two challenges. One is to deal with the problem of high dimensionality which often occurs in any text mining task due to a large number of unique words in the dataset vocabulary. The second is to improve accuracy based on context-based semantics captured by word embeddings based on BERT. It was shown in [62] that it performs much better than other traditional as well as state-of-the-art text clustering techniques in terms of accuracy, especially for large-sized datasets.

ii. WordNet-based
In this category, a recently proposed technique is used that uses WordNet to capture lexical-chains [61]. Lexical chains are a group of words that are semantically related. With the formation of these groups, a text corpus is represented with reduced dimensionality. Hence, it can produce better clustering results than other tech-

[6] Available at: https://vhasoares.github.io/downloads.html. Accessed: 2022-04-08.

[7] Available at: http://qwone.com/~jason/20Newsgroups/. Accessed: 2022-04-08.

niques which are based on the vector space model representation of a text corpus.

iii. Partitioning-based
In this category, the widely used K-means clustering [58] is used to perform clustering based on the VSM representation of text documents containing TF-IDF scores.

iv. Hierarchical-based
In this category, the well-known Agglomerative clustering [94] is used which performs clustering in a bottom-up approach. In this approach, initially, the VSM representation of each text document is considered as a single data point which are subsequently merged to form larger groups until a desired group of clusters is made.

v. Density-based
Here, a fairly recent clustering algorithm known as HDBSCAN [60] is used which basically extends the original DBSCAN [94] algorithm into a hierarchical form so as to come up with a more robust clustering algorithm.

4.3 Performance metrics

Metrics for assessing the performance of any clustering algorithm are classified into two categories: External and Internal. External metrics require external ground truth labels for each data point. Using these truth labels, these metrics compute the efficiency of the clustering results. Internal metrics are independent of external ground truth labels because these metrics check the quality of clusters using the internal attributes of each cluster such as separation among the clusters and the density of an individual cluster. The metrics used in this paper are as follows.

1. Silhouette coefficient: This is an internal clustering metric that is defined by the following equation.

$$s = (b - a)/\max(b - a) \tag{9}$$

a denotes the average distance between a single data point and all other data points in a cluster. b denotes the average distance between a single data point and all other data points in the next nearest cluster. The silhouette coefficient for a set of data points is given as the average value of the silhouette coefficient of all the data points in the set. It lies in the range [-1, +1], where a higher value indicates dense and well-separated clusters.

2. Adjusted Rand Index: Adjusted Rand Index(ARI) measures the quality of clusters given the external ground truth labels. It compares two different clustering assignments that ignore different permutations of the same clustering. It ranges between [-1, +1], where two similar clusterings achieve a score of $+1$ and two different clusterings achieve a score of -1. Its detailed mathematical formulation can be found in [42].

3. Purity: Purity is also an external metric that first assigns all the data points to a class to which the maximum number of data points in that cluster belongs. The correct number of class labels is summed over all the clusters and then normalized by the total number of data points [59].

4.4 Analysis of results

In this section, clustering algorithms of different types are analyzed based on their performance on several datasets. Algorithms of following types are analyzed: embedding-based (WEClustering [62]), WordNet-based(semantic clustering [61]), partitioning-based (K-means [58]), hierarchical-based (agglomerative [94]) and density-based (HDBSCAN [60]).

All three metrics, i.e., purity, Silhouette coefficient, and ARI indicate that embedding-based clustering is giving the best result in almost all datasets as per Tables 6, 7 and 8. This can be attributed to the fact that they can well capture the context-based semantics of text better than other techniques. For comparatively small-sized datasets, i.e., Articles-253, although embedding-based and partitioning-based clustering are giving the best score according to all three metrics, however, others also do not lag much behind. But, as one goes toward larger datasets such as 20NG and 20NG-long, the performance difference is larger. Density-based clustering is lagging much behind for large-sized datasets as can be seen in Figs. 6, 7, and 8 for the values of purity, ARI and Silhouette coefficient, respectively. It can also be observed that for the dataset Scopus-long, WordNet-based clustering is giving the best purity score. This can be attributed to the fact that it can also capture context-based semantics to some extent with the help of WordNet.

A second perspective of observing the results is looking at the variation of results of each clustering algorithm among themselves with respect to dataset size. A common trend can be easily seen that as the dataset size is growing, the performance of each algorithm is declining. For example, in Articles-253, ARI value of Embedding-based technique is 0.970 and for 20NG-long, it is 0.165. Similarly, for Density-based the values are 0.734 and 0.001, respectively. Hence, it can be concluded that high dimensionality (in case of large sized datasets) has a good impact over the performance of each clustering algorithm.

Table 6 Purity values of different clustering techniques on different datasets

S.no	Datasets	Embedding-based	WordNet-based	Partitioning-based	Hierarchical-based	Density-based
1	Articles-253	0.988	0.986	**1.0**	0.992	0.893
2	Classic4	**0.970**	0.785	0.887	0.961	0.830
3	Classic4-long	**0.961**	0.745	0.845	0.966	0.802
4	Scopus	**0.956**	0.750	0.916	0.866	0.378
5	Scopus-long	**0.765**	**0.793**	0.750	0.729	0.439
6	20NG	**0.637**	0.345	0.607	0.506	0.339
7	20NG-long	**0.409**	0.221	0.365	0.363	0.120

sparaThe best values are shown in bold

Table 7 ARI values of different clustering techniques on different datasets

S.no	Datasets	Embedding-based	WordNet-based	Partitioning-based	Hierarchical-based	Density-based
1	Articles-253	0.970	0.989	**1.0**	0.978	0.734
2	Classic4	**0.912**	0.458	0.737	0.886	0.456
3	Classic4-long	**0.888**	0.467	0.641	0.902	0.477
4	Scopus	**0.893**	0.579	0.807	0.702	0.108
5	Scopus-long	**0.600**	0.258	0.518	0.524	0.010
6	20NG	**0.344**	0.180	0.272	0.149	0.003
7	20NG-long	**0.165**	0.098	0.079	0.085	0.001

The best values are shown in bold

Table 8 Silhouette coefficient values of different clustering techniques on different datasets

S.no	Datasets	Embedding-based	WordNet-based	Partitioning-based	Hierarchical-based	Density-based
1	Articles-253	**0.458**	0.247	0.097	0.097	0.087
2	Classic4	**0.264**	0.254	0.013	0.011	0.010
3	Classic4-long	0.259	0.026	0.010	0.010	0.008
4	Scopus	**0.314**	0.056	0.017	0.015	0.005
5	Scopus-long	**0.240**	0.047	0.012	0.012	−0.016
6	20NG	**0.130**	0.032	0.002	0.009	0.002
7	20NG-long	**0.086**	0.025	0.007	0.004	0.001

The best values are shown in bold

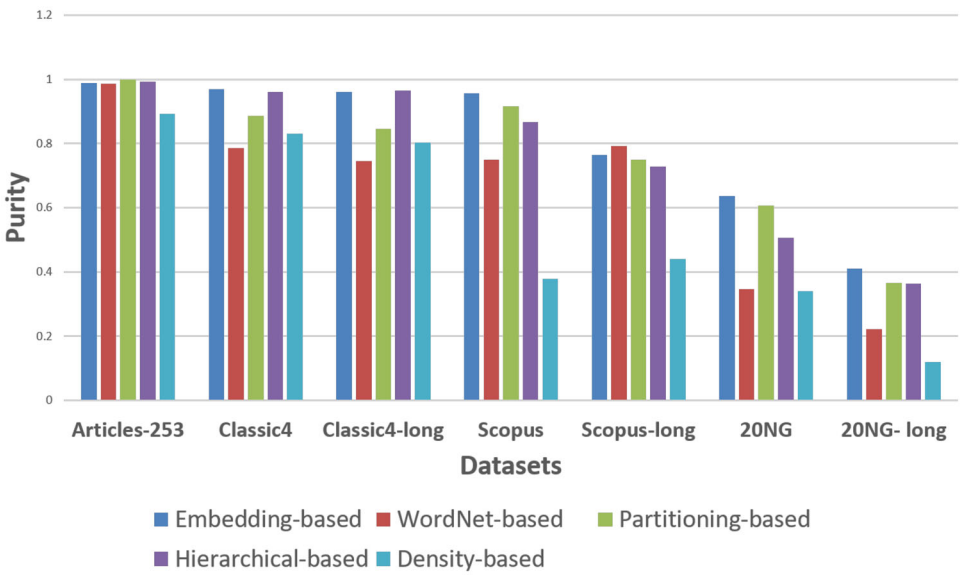

Fig. 6 A bar chart showing purity values attained by different clustering algorithms on different datasets

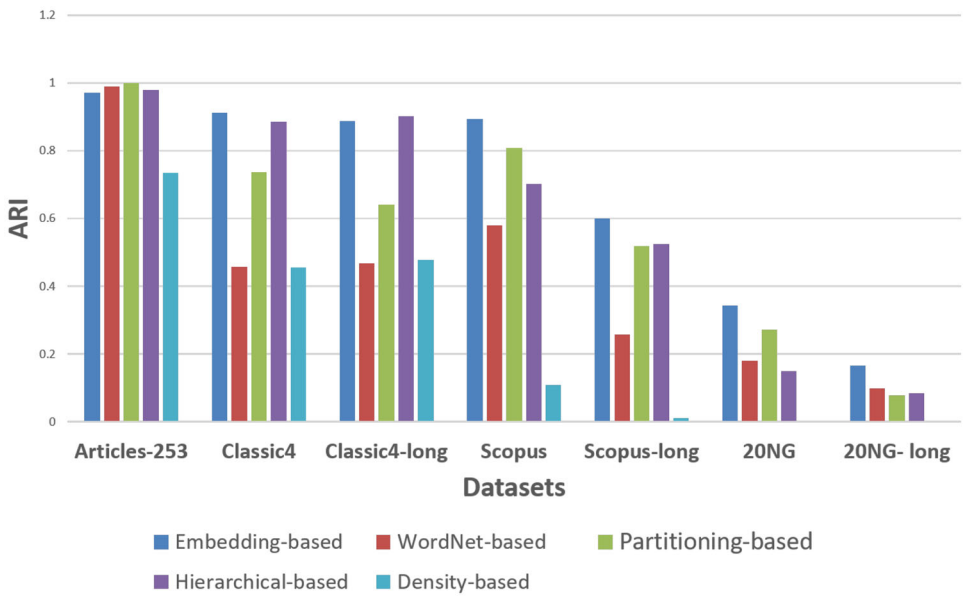

Fig. 7 A bar chart showing ARI values attained by different clustering algorithms on different datasets

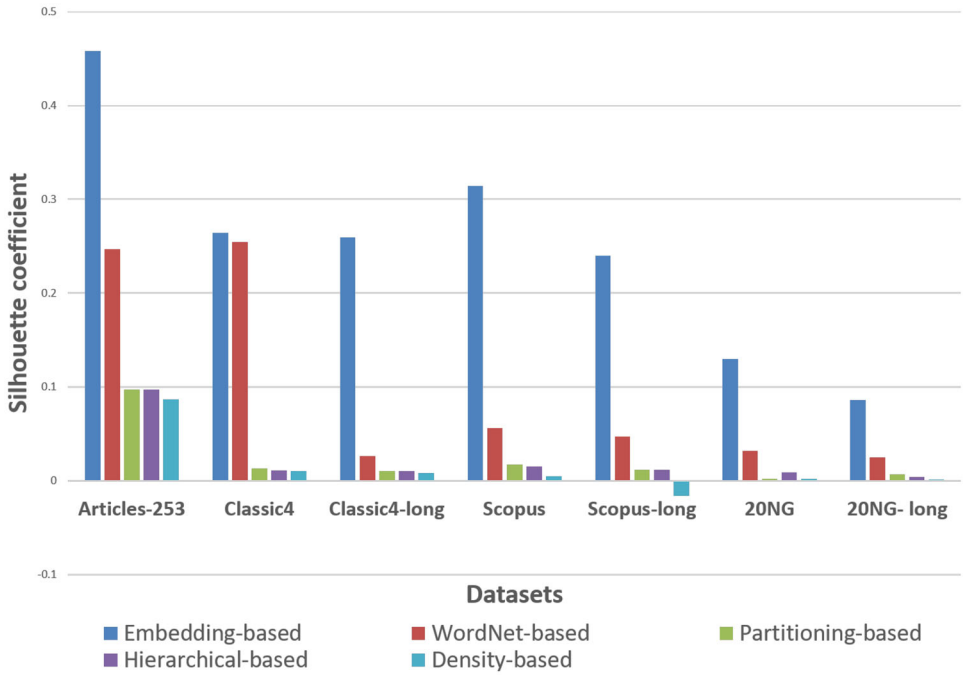

Fig. 8 A bar chart showing purity values attained by different clustering algorithms on different datasets

5 Conclusions and future scope

Looking at the experimental results of this research, it can be concluded that for small sized datasets, all types of clustering algorithms are almost equally good, however, high dimensionality affects the accuracy of each clustering algorithm significantly. Hence, more sophisticated clustering techniques can be designed that consider contextual information as well as can deal high dimensionality, so that better performance can be achieved for large sized datasets. Specifically, text clustering based on state- of-the-art word embeddings can be applied for tasks such as text classification, text summarization across various domains such as legal documents, scientific documents, book summarization and sentiment analysis as an extension of this research.

Data availability All the datasets analyzed in this paper are publicly available, links to which have been included as footnotes in the Sect. 3.1.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Abbasi-Moud, Z., Vahdat-Nejad, H., Sadri, J.: Tourism recommendation system based on semantic clustering and sentiment analysis. Expert Syst. Appl. **167**, 114324 (2021)
2. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data. Data Min. Knowl. Disc. **11**(1), 5–33 (2005)
3. Alammar, J.: The illustrated Bert, Elmo, and Co. http://jalammar.github.io/illustrated-bert/ (2018). Accessed 25 Jan 2021
4. Almeida, F., Xexéo, G.: Word embeddings: a survey (2019). arXiv preprint arXiv:1901.09069
5. Altınçay, H., Erenel, Z.: Analytical evaluation of term weighting schemes for text categorization. Pattern Recogn. Lett. **31**(11), 1310–1323 (2010)
6. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: ordering points to identify the clustering structure. In: ACM Sigmod Record, vol. 28, pp. 49–60. ACM (1999)
7. Baghel, R., Dhir, R.: Text document clustering based on frequent concepts. In: 2010 First International Conference On Parallel, Distributed and Grid Computing (PDGC 2010), pp. 366–371. IEEE (2010)
8. Bakarov, A.: A survey of word embeddings evaluation methods (2018). arXiv preprint arXiv:1801.09536
9. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is "nearest neighbor" meaningful? In: International Conference on Database Theory, pp. 217–235. Springer, Berlin (1999)
10. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: the fuzzy c-means clustering algorithm. Comput. Geosci. **10**(2–3), 191–203 (1984)
11. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
12. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017)
13. Bottou, L.: Stochastic gradient descent tricks. In: Neural networks: Tricks of the Trade, pp. 421–436. Springer, Berlin (2012)
14. Bouras, C., Tsogkas, V.: A clustering technique for news articles using wordnet. Knowl.-Based Syst. **36**, 115–128 (2012)
15. Brainard, J.: Scientists are drowning in covid-19 papers. can new tools keep them afloat. Science **13**(10), 1126 (2020)
16. Covid-19 research highlights. https://www.springernature.com/in/researchers/campaigns/coronavirus (2020). Accessed 06 May 2022
17. Camacho-Collados, J., Pilehvar, M.T.: From word to sense embeddings: a survey on vector representations of meaning. J. Artif. Intell. Res. **63**, 743–788 (2018)
18. Cecchini, F.M., Riedl, M., Fersini, E., Biemann, C.: A comparison of graph-based word sense induction clustering algorithms in a pseudoword evaluation framework. Lang. Resour. Eval. **52**, 733–770 (2018)
19. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. J. Am. Soc. Inf. Sci. **41**(6), 391–407 (1990)
20. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B (Methodol.) 1–38 (1977)
21. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2018). arXiv preprint arXiv:1810.04805
22. Dey, A., Bhattacharyya, S., Dey, S., Platos, J., Snasel, V.: A quantum inspired differential evolution algorithm for automatic clustering of real life datasets. Multimedia Tools Appl. 1–30 (2023)
23. Duan, T., Lou, Q., Srihari, S.N., Xie, X.: Sequential embedding induced text clustering, a non-parametric Bayesian approach. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 68–80. Springer, Berlin (2019)
24. Elhadad, M.K., Badran, K.M., Salama, G.I.: A novel approach for ontology-based dimensionality reduction for web text document classification. Int. J. Software Innov. (IJSI) **5**(4), 44–58 (2017)
25. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, vol. 96, pp. 226–231 (1996)
26. Firth, J.R.: A synopsis of linguistic theory, 1930-1955. Studies in linguistic analysis (1957)
27. Fisher, D.H.: Knowledge acquisition via incremental conceptual clustering. Mach. Learn. **2**(2), 139–172 (1987)
28. Fodeh, S., Punch, B., Tan, P.N.: On ontology-driven document clustering using core semantic features. Knowl. Inf. Syst. **28**(2), 395–421 (2011)
29. Gennari, J.H., Langley, P., Fisher, D.: Models of incremental concept formation. Artif. Intell. **40**(1–3), 11–61 (1989)
30. Guha, S., Rastogi, R., Shim, K.: Cure: an efficient clustering algorithm for large databases. In: ACM Sigmod Record, vol. 27, pp. 73–84. ACM (1998)
31. Guha, S., Rastogi, R., Shim, K.: ROCK: a robust clustering algorithm for categorical attributes. Inf. Syst. **25**(5), 345–366 (2000)
32. Han, J., Pei, J., Kamber, M.: Data Mining: Concepts and Techniques. Elsevier, New York (2011)
33. Harris, Z.S.: Distributional structure. Word **10**(2–3), 146–162 (1954)
34. Hinneburg, A., Keim, D.A.: Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering. In: Proceedings of the 25th International Conference on Very Large Databases, pp. 506–517 (1999)
35. Hinneburg, A., Keim, D.A., et al.: An efficient approach to clustering in large multimedia databases with noise. In: KDD, vol. 98, pp. 58–65 (1998)
36. Hirst, G., St-Onge, D., et al.: Lexical chains as representations of context for the detection and correction of malapropisms. WordNet Electronic Lexical Database **305**, 305–332 (1998)
37. Hofmann, T.: Probabilistic latent semantic analysis (2013). arXiv preprint arXiv:1301.6705
38. Hosseini, S., Varzaneh, Z.A.: Deep text clustering using stacked AutoEncoder. Multimedia Tools Appl. **81**(8), 10861–10881 (2022)
39. Hotho, A., Staab, S., Stumme, G.: Ontologies improve text document clustering. In: Third IEEE International Conference on Data Mining, pp. 541–544. IEEE (2003)
40. Huang, A., Milne, D., Frank, E., Witten, I.H.: Clustering documents using a wikipedia-based concept representation. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 628–636. Springer, Berlin (2009)
41. Huang, Z.: A fast clustering algorithm to cluster very large categorical data sets in data mining. DMKD **3**(8), 34–39 (1997)
42. Hubert, L., Arabie, P.: Comparing partitions. J Classif **2**, 193–218 (1985)
43. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs (1988)
44. Jain, D., Borah, M.D., Biswas, A.: A sentence is known by the company it keeps: improving legal document summarization using deep clustering. Artif. Intell. Law 1–36 (2023)
45. Jardine, N., van Rijsbergen, C.J.: The use of hierarchic clustering in information retrieval. Inf. Storage Retrieval **7**(5), 217–240 (1971)
46. Jasinska-Piadlo, A., Bond, R., Biglarbeigi, P., Brisk, R., Campbell, P., Browne, F., McEneaney, D.: Data-driven versus a domain-led approach to k-means clustering on an open heart failure dataset. Int. J. Data Sci. Anal. **15**(1), 49–66 (2023)
47. Jayarajan, D., Deodhare, D., Ravindran, B.: Document clustering using lexical chains (2007)

48. Jayarajan, D., Deodhare, D., Ravindran, B.: Lexical chains as document features. In: Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I (2008)
49. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification (2016). arXiv preprint arXiv:1607.01759
50. Karypis, G., Han, E.H., Kumar, V.: Chameleon: Hierarchical clustering using dynamic modeling. Computer **32**(8), 68–75 (1999)
51. Kim, J., Yoon, J., Park, E., Choi, S.: Patent document clustering with deep embeddings. Scientometrics 1–15 (2020)
52. Kohonen, T.: The self-organizing map. Neurocomputing **21**(1–3), 1–6 (1998)
53. Lan, M., Sung, S.Y., Low, H.B., Tan, C.L.: A comparative study on term weighting schemes for text categorization. In: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., vol. 1, pp. 546–551. IEEE (2005)
54. Li, Y., Cai, J., Wang, J.: A text document clustering method based on weighted Bert model. In: 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), vol. 1, pp. 1426–1430. IEEE (2020)
55. Li, Y., Luo, C., Chung, S.M.: A parallel text document clustering algorithm based on neighbors. Clust. Comput. **18**(2), 933–948 (2015)
56. Liu, Z., Lin, Y., Sun, M.: Representation Learning for Natural Language Processing. Springer Nature, Berlin (2020)
57. Luo, C., Li, Y., Chung, S.M.: Text document clustering based on neighbors. Data Knowl. Eng. **68**(11), 1271–1288 (2009)
58. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. Oakland, CA, USA (1967)
59. Manning, C.D., Schütze, H., Raghavan, P.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
60. McInnes, L., Healy, J., Astels, S.: hdbscan: Hierarchical density based clustering. J. Open Source Softw. **2**(11), 205 (2017)
61. Mehta, V., Bawa, S., Singh, J.: Stamantic clustering: Combining statistical and semantic features for clustering of large text datasets. Expert Syst. Appl. **174**, 114710 (2021)
62. Mehta, V., Bawa, S., Singh, J.: Weclustering: word embeddings based text clustering technique for large datasets. Complex Intell. Syst. **7**(6), 3211–3224 (2021)
63. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013). arXiv preprint arXiv:1301.3781
64. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. Adv. Neural. Inf. Process. Syst. **26**, 3111–3119 (2013)
65. Miller, G.A.: Wordnet: a lexical database for English. Commun. ACM **38**(11), 39–41 (1995)
66. Morris, J., Hirst, G.: Lexical cohesion computed by thesaural relations as an indicator of the structure of text. Comput. Linguist. **17**(1), 21–48 (1991)
67. Mustafi, D., Mustafi, A.: A differential evolution based algorithm to cluster text corpora using lazy re-evaluation of fringe points. Multimedia Tools Appl. 1–25 (2023)
68. Naik, A., Maeda, H., Kanojia, V., Fujita, S.: Scalable Twitter user clustering approach boosted by Personalized PageRank. Int. J. Data Sci. Anal. **6**(4), 297–309 (2018)
69. Nasir, J.A., Varlamis, I., Karim, A., Tsatsaronis, G.: Semantic smoothing for text clustering. Knowl.-Based Syst. **54**, 216–229 (2013)
70. Ng, R.T., Han, J.: Efficient and effective clustering methods for spatial data mining. In: Proceedings of VLDB, pp. 144–155 (1994)
71. Ng, R.T., Han, J.: CLARANS: a method for clustering objects for spatial data mining. IEEE Trans. Knowl. Data Eng. **14**(5), 1003–1016 (2002)
72. Park, H.S., Jun, C.H.: A simple and fast algorithm for k-medoids clustering. Expert Syst. Appl. **36**(2), 3336–3341 (2009)
73. Park, J., Park, C., Kim, J., Cho, M., Park, S.: ADC: advanced document clustering using contextualized representations. Expert Syst. Appl. **137**, 157–166 (2019)
74. Patil, L.H., Atique, M.: A semantic approach for effective document clustering using wordnet (2013). arXiv preprint arXiv:1303.0489
75. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
76. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations (2018). arXiv preprint arXiv:1802.05365
77. Recupero, D.R.: A new unsupervised method for document clustering by using wordnet lexical and conceptual relations. Inf. Retrieval **10**(6), 563–579 (2007)
78. Robertson, S.: Understanding inverse document frequency: on theoretical arguments for IDF. J. Doc. **60**(5), 503–520 (2004)
79. Roul, R.K.: An effective approach for semantic-based clustering and topic-based ranking of web documents. Int. J. Data Sci. Anal. **5**, 269–284 (2018)
80. Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X.: DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. ACM Trans. Database Syst. (TODS) **42**(3), 1–21 (2017)
81. Sedding, J., Kazakov, D.: Wordnet-based text document clustering. In: proceedings of the 3rd Workshop on Robust Methods in Analysis of Natural Language Data, pp. 104–113. Association for Computational Linguistics (2004)
82. Sehgal, G., Garg, D.K.: Comparison of various clustering algorithms. Int. J. Comput. Sci. Inf. Technol. **5**(3), 3074–3076 (2014)
83. Sheikholeslami, G., Chatterjee, S., Zhang, A.: Wavecluster: a multi-resolution clustering approach for very large spatial databases. In: VLDB, vol. 98, pp. 428–439 (1998)
84. Shi, H., Wang, C., Sakai, T.: Self-supervised document clustering based on Bert with data augment (2020). arXiv preprint arXiv:2011.08523
85. Shirkhorshidi, A.S., Aghabozorgi, S., Wah, T.Y.: A comparison study on similarity and dissimilarity measures in clustering continuous data. PLoS ONE **10**(12), e0144059 (2015)
86. Sinoara, R.A., Camacho-Collados, J., Rossi, R.G., Navigli, R., Rezende, S.O.: Knowledge-enhanced document embeddings for text classification. Knowl.-Based Syst. **163**, 955–971 (2019)
87. Steinbach, M., Ertöz, L., Kumar, V.: The challenges of clustering high dimensional data. In: New Directions in Statistical Physics, pp. 273–309. Springer, Berlin (2004)
88. Steinbach, M., Karypis, G., Kumar, V., et al.: A comparison of document clustering techniques. In: KDD Workshop on Text Mining, vol. 400, pp. 525–526. Boston (2000)
89. Urkude, G., Pandey, M.: Design and development of density-based effective document clustering method using ontology. Multimedia Tools Appl. **81**(23), 32995–33015 (2022)
90. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
91. Wang, S., Zhou, W., Jiang, C.: A survey of word embeddings based on deep learning. Computing **102**(3), 717–740 (2020)
92. Wang, W., Yang, J., Muntz, R., et al.: Sting: a statistical information grid approach to spatial data mining. In: VLDB, vol. 97, pp. 186–195 (1997)
93. Wei, T., Lu, Y., Chang, H., Zhou, Q., Bao, X.: A semantic approach for text clustering using wordnet and lexical chains. Expert Syst. Appl. **42**(4), 2264–2275 (2015)
94. Xu, D., Tian, Y.: A comprehensive survey of clustering algorithms. Ann. Data Sci. **2**(2), 165–193 (2015)

95. Xu, X., Ester, M., Kriegel, H.P., Sander, J.: A distribution-based clustering algorithm for mining in large spatial databases. In: 14th International Conference on Data Engineering, 1998. Proceedings, pp. 324–331. IEEE (1998)
96. Yue, L., Zuo, W., Peng, T., Wang, Y., Han, X.: A fuzzy document clustering approach based on domain-specified ontology. Data Knowl. Eng. **100**, 148–166 (2015)
97. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large databases. In: ACM Sigmod Record, vol. 25, pp. 103–114. ACM (1996)