https://doi.org/10.33472/AFJBS.6.6.2024.5561-5576

African Journal of Biological Sciences

Journal homepage: http://www.afjbs.com

Research Paper Open Access

Article information

Article title

An Intensified text clustering algorithm for Noisy Big Data: Haphazardly Dense Clustering with more Noise (HDCN)

Authors

U. Vageeswari [a*], B. Lavanya[b]

Affiliations

[a] Research Scholar, Department of Computer Science, University of Madras, Chennai, India., uvageeswariphd@gmail.com

[b] Associate Professor, Department of Computer Science, University of Madras, Chennai, India., lavanmu@unom.ac.in

Corresponding author's email address and Twitter handle

uvageeswariphd@gmail.com

Abstract

Large Databases with multi-dimensions have large amounts of noise and sometimes only a small portion of it accounts for the clustering. Text Clustering involves stages of pre-processes since text data are semi-structured and unstructured. The method for transforming the text into another numerical form is referred to as text vectorization. TF-IDF and Word2Vec have widely used vectorization methods. This paper proposes a distance-based Text clustering algorithm, Haphazardly Dense Clustering with more Noise (HDCN) for the noisy and unknown dataset. Two datasets with two distinct vectorization approaches are used to test the algorithm. The same is compared with available state of art methods like K-Means, Hierarchical, and DBSCAN clustering. The DBSCAN algorithm detects 30.5 % of noise on average, but the HDCN algorithm detects 96.675 % of noise.

- *Method* that *is effective in clustering sentences that contain a greater quantity of noise*
- *Organizing sentences can aid in summarizing, categorizing, and analyzing sentiment in natural language processing tasks.*
- *The method helps categorize data that can be expressed using mathematical symbols and has a higher level of distortion.*

Keywords

TF-IDF, Word2Vec, KMeans, Agglorametive, DBSCAN, HDCN

Article Info

Volume6, Issue 6, June 2024

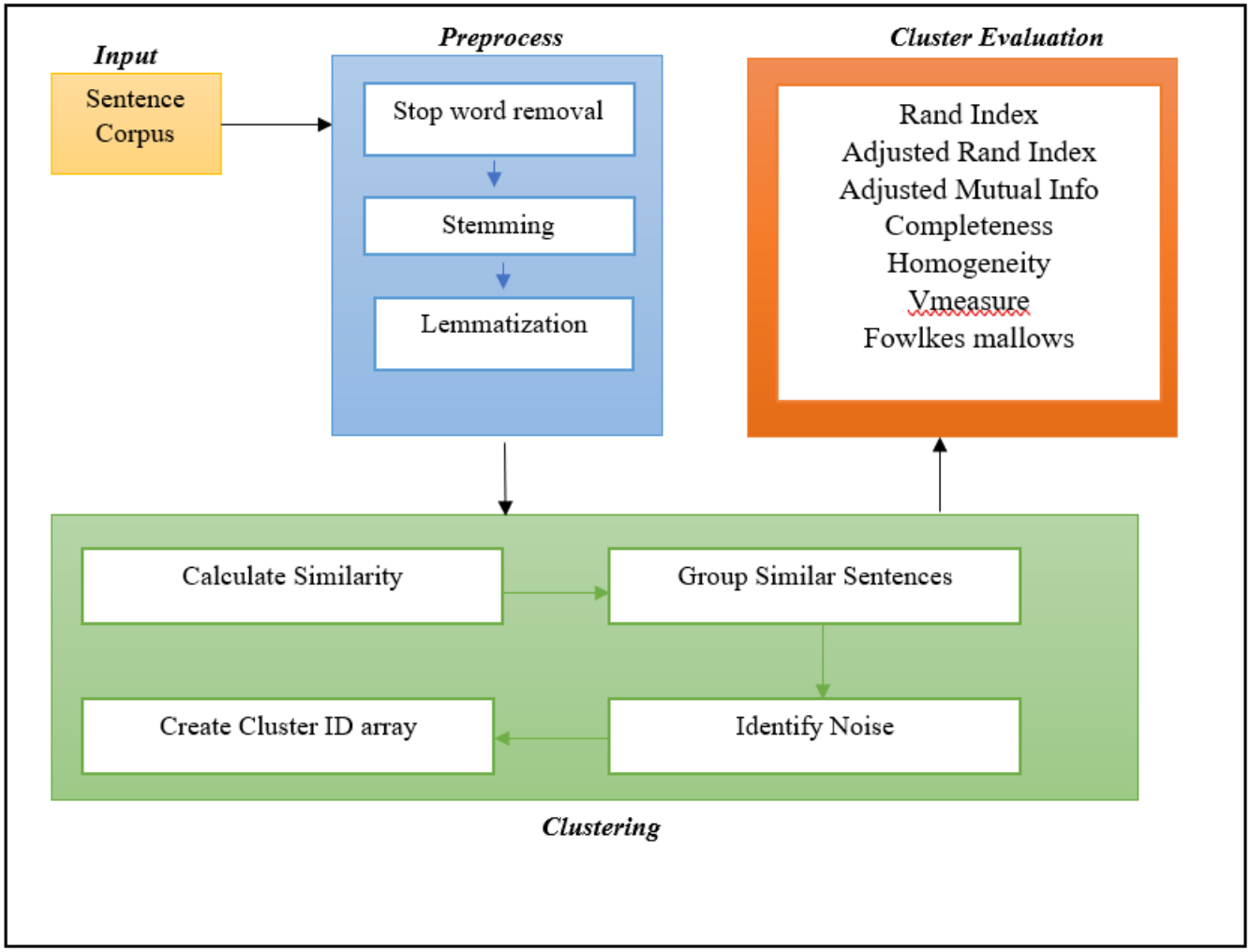
Received: 14 JAN 2024

Accepted: 21 April 2024

Published: 15 June 2024

doi.org/10.33472/AFJBS.6.6.2024.5561-5576

Graphical abstract



Specifications table

Subject area	Computer Science
More specific subject area	*Natural Language Processing, Clustering*
Name of your method	**Haphazardly Dense Clustering with more Noise (HDCN)**
Name and reference of original method	*Not Applicable*
Resource availability	*Not Applicable*

Method details

1.0 Introduction

The clustering problem is described as the quest for a homogeneous dataset in heterogeneous data sets. Clustering is a concept that can be applied to a variety of Text mining tasks like browsing and organizing documents, summarization of the corpus, classification of documents, document management, and indexing. Text data is unstructured and chaotic. As a result, considerable pre-processing is required before it can be used in any method that requires organized and well-defined fixed-length inputs (Ex: Machine Learning algorithm). The method for transforming the text into another numerical form is referred to as text vectorization. Count Vectorizer, Binary Term Frequency, Term Frequency Inverse Document Frequency (TF-IDF), and Word2Vec are some of the text vectorization methods.

There are a number of different ways for determining how similar two phrases are. [1]. Each has its own unique range of applications, and the best similarity measure to choose is determined by the kind of data set and the information that must be retrieved from it. The two categories of text similarity algorithms are character-based algorithms and term-based algorithms. Longest Common Substring (LCS), Damerau-Levenshtein [2][3], Jaro [4] [5], Jaro Winkler [6], Needleman-Wunsch, Smith-Waterman [7], N-gram [8] are some of the character-based similarity methods most popularly used. The term-based similarity method is used to cluster sentences. Some term-based similarity measurements include Jaccard similarity, Euclidean distance, Dice's coefficient similarity, Cosine similarity, and Manhattan distance similarity.

A data point or value that stands out from the rest of the data in a dataset is known as a "noise" in most clustering algorithms, one or two noise points are considered while clustering. However, there may be more noise data points to consider throughout the clustering process. [9]. Clustering and trend identification in very big datasets is challenging due to the enormous quantity of noise in the databases and the fact that only a tiny fraction of the large databases are used for clustering [10].

This paper examines the applicability of several clustering [11] methods as well as their limitations [12]. It presented a novel Text clustering technique called Haphazardly Dense Clustering with More Noise (HDCN) for datasets with many noise elements and when the quantity of clusters is unknown. [13] This clustering algorithm can be used to find clusters of similar words from a corpus of utterances, identify similarly behaving clients, detection of identical sentences in plagiarism software, and assignment duplicate detection.

The rest of this paper is presented as follows: The review of clustering methods is presented in Section 2. Section 3 explains the proposed sentence clustering framework with the HDCN algorithm. In Section 4, a brief description of the dataset used in the experiment is provided. The performance of the HDCN algorithm and competing approaches are tabulated and examined in Section 5. The paper is concluded in Section 6.

2.0 Related Work

Clustering is an example of an unlabelled data problem because there is no predetermined category to which the data should be assigned. Text clustering may be done in a variety of ways, according to different scholars. For text clustering, some people utilize frequent item sets.[14]. Some use semi-supervised clustering methods.[15] No clustering algorithm outperforms all data sets. It is the responsibility of the researcher to choose the right clustering algorithm for the application they are analysing. Though clustering is a very old scientific problem. It started with the classification of Species[51]. It still has a very large research space and there are many unsolved problems. Because no single algorithm is suitable for every data set, even a single approach applied to the same dataset will not produce the desired results for a different type of research.[16]

The following key properties must be incorporated into any clustering algorithm:

- The Data points inside similar clusters should be like one another.
- As many diverse data points as possible should be included in each distinct cluster.

KMeans Algorithm [17] gives the best result when the given data set is distinct, well separated and normally distributed. Kmeans requires the number of clusters before starting the cluster [18], it is a very tricky problem, and several researchers proposed various methods [19] to identify the count of clusters scientifically.[20] By rule of thumb, the Elbow method (the point of inflexion on the curve) [21] [22], the Information Criterion Approach [23], the Information-Theoretic Approach [24], and Cross-validation [25] are some of the methods proposed by several researchers. It is unable to handle noisy data [26] [27] and outliers [28] [29][30].

The count of clusters is not needed for the hierarchical clustering process. It requires either a certain count of clusters to be generated or a certain radius threshold value to be utilised in cluster formation. Later using a dendogram [31] a meaningful number of clusters can be taken. However, determining the count of clusters using a dendogram might be tricky at times. It is sensitive to noise [32] and outliers [33]. There are two sorts of hierarchical clustering techniques: 1) AGNES (Agglomerative Hierarchical Clustering Algorithm) and DIANA (Divisive Hierarchical Clustering Algorithm). Agglomerative clustering is good at identifying small clusters. This method's fundamental tenet is that it treats every single value as a separate cluster and combines the cluster's closest pairings at each stage. Each cycle results in a cluster integrating with another cluster eventually only one is left. Large clusters can be identified using divisive clustering. This method's key idea is to regard every input value as belonging to the same cluster and to segregate them from groups that are not comparable in each subsequent iteration. N clusters remain at the conclusion.

DBSCAN stands for density-based spatial clustering with noise for programmes. Finding high-density areas that are divided from each other with low-density areas is the main goal of the DBSCAN algorithm. It does not require a lot of domain expertise, can find clusters of any structure, and works well for a big database (i.e., greater than several thousand). Two stages are required to determine the density of a region: Step 1: Number of points inside a circle with a radius of Eps (E) from point P. Step 2: Find the circle with the fewest number of points for each point in the cluster. What DBSCAN cannot do: 1. DBSCAN does not work well if the dataset has data points that are grouped in different ways. 2. Establishing (the distance parameter) and min pts can be hard, and you may need to test various values of e and min pts several times.

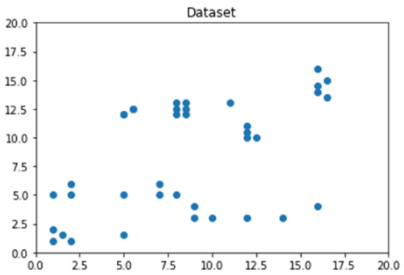


Fig 1: Dataset

According to the literature and study papers, when the number of clusters is known only, KMEANS offers superior results; If the data is hierarchical, agglomerative, or divisive is selected; for big and evenly distributed clusters of data, DBSCAN is the answer. This paper proposes a clustering algorithm for a dataset that is haphazardly dense with more noise and the number of clusters is unknown.

3.0 Proposed Sentence Clustering Framework

A framework for grouping sentences using Haphazardly Dense Clustering with more Noise is presented in this paper (HDCN). The flow of the sentence clustering process is depicted in Figure (2). The corpus of sentences is a collection of all sentences. All sentences have been pre-processed.

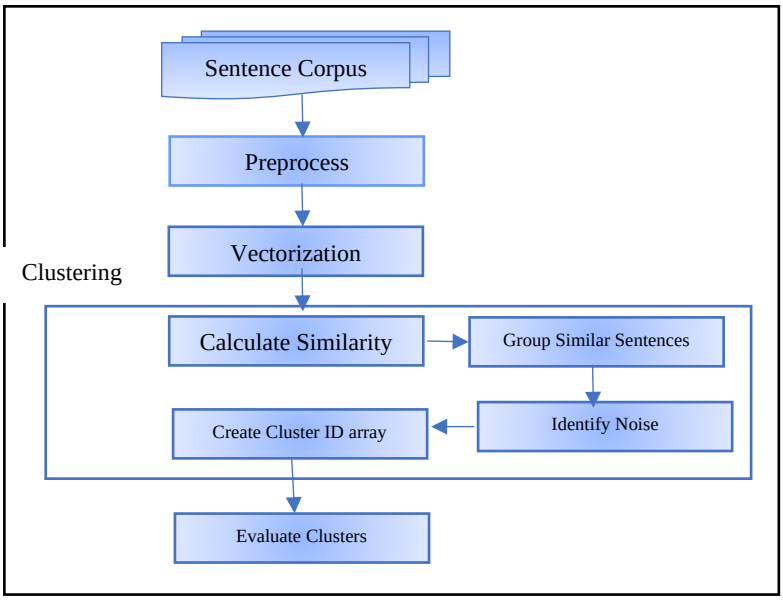


Fig 2: Proposed Sentence clustering Framework

3.1 Pre-process:

Pre-processing includes getting rid of stop words, lemmatizing, stemming, and changing all the letters to lowercase. Stop words are frequently employed in Text Mining and Natural Language Processing (NLP) to filter out terms that are so common that they do not tell us much. Stemming is the process of breaking a word down into its stable root word (Ex: Flying into Fly). The process of lemmatization is to identify the source word. (Example: Worst to Bad).

3.2. Vectorization:

There are various vectorization methods for text. In Binary Term Frequency 1 indicates the term is extant and indicates the term is not extant in a document, In Bag of Words (Bow) the rate of recurrence of terms in a document is captured. Count Vectorizer: [34] [35] A count vectorizer converts a string array into a frequency representation. Count vectors can help to determine what type of text it is by looking at the frequency of terms in a text .It examines words that show up a lot in a dataset as the most statistically important. One of its major weaknesses is that it does not consider connections among phrases, like linguistic similarities.

Term Frequency Inverse Document Frequency (TF-IDF): [36] [37] This is depending on the number of times a word appears in the corpus. It also gives a number that shows how important a phrase is for the descriptive statistic. TFIDF is better than Count Vectorizers because it not only looks at how often a word appears in the corpus but also how important it is. Then, we might

get rid of the phrases that are not as important for analysis. This would reduce the number of inputs to the model and make it simpler to build. The formula to calculate TF-IDF is shown in equation (1).

Word Frequency is a statistic that quantifies the frequency with which a term appears in a text, as demonstrated by the equation: (2). Because the length of each document is different, a phrase may show up a lot more often in textual information than in short ones. In most cases, the data is normalized by dividing the number of times a word is used by the total number of words in the text. The equation for the Inverse Document Frequency (IDF) shows how it can be used to figure out how important a word is (3). While estimating TF, each word is assigned the same amount of importance. Certain words may appear often yet have little meaning. IDF is used to scale down the basic words while scaling up the uncommon ones. TFIDF is based on the theory that terms that are simultaneously quite prevalent and too rare in a collection of documents are not statically important for finding a pattern. The Logarithmic factor in TFIDF penalises terms in the corpus that are too common or unusual by assigning them low TFIDF scores.

$$\mathrm{TF-IDF} = \mathrm{TermFrequency\ (TF)} * \mathrm{InverseDocumentFrequency\ (IDF)} \quad \ldots(1)$$

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \epsilon d} f_{t',d}} \quad \ldots(2)$$

$$df(t,D) = \log \frac{N}{|d \varepsilon D : t \varepsilon d|} \quad \ldots(3)$$

Where t stands for the term, d for the document (group of words), N for the number of terms in the collection of documents, and $f_{t,d}$ for the number of times term t appears in document d.

A technique called Word2Vec changes words into vectors [38]. Word2Vec begins with a single portrayal of all the words in the collection of documents and uses a large set of documents of datasets to train a Neural Network (NN) including one hidden layer. Most of the time, there are two ways to train the NN: Continuous Bag of Words (CBOW): Predict the vector representation of the centre/target word by looking at a window of words from the context. Skip-Gram (SG): Use the centre/target word to guess how a window of word embeddings will be represented as a vector. The vectorization method is selected based on the application's requirements. The vectorization algorithms TF-IDF and Word2Vec are regularly utilized. To evaluate the proposed algorithm's performance. Our framework uses the TF-IDF and Word2vec vectorization algorithms. Both of these vectorization methods are used to test the clustering algorithm's performance. After clean sentences have been vectorized, the clustering step begins.

3.3 Clustering:

Text similarity can be evaluated in several ways. In Dice's coefficient [42], Dice (Sa, Sb) stands for the number of similar phrases or the amount of tokens that are the same in the strings S_a and S_b.

The Jaccard similarity [43] is calculated by deducting the amount of the intersection of these two sets from the amount of the union of those sets. When the text is quite long, Jaccard will solve the similarity through the set; the similarity will be smaller. As a result, when Jaccard is used to calculate similarity, it is typically first normalised. Words in English can be reduced to the same root, but words in Chinese can be reduced to synonyms.

In Cosine similarity [39] [40] [41] In lieu of computing the angles and distances, the cosine distance is transformed into a vector space angle problem relating to the two points. The similarity is calculated using the cosine of the angle formed by two vectors. Due to the large size of the text, the cosine distance is a superior method for determining similarity, even if the Euclidean distance between two equivalent documents is great. This can also be used to assess the relevancy of a document's point of view. Cosine similarity uses the angle between data points, the lesser the angle more similar the data points are, shown in Eqn (4).

$$cos\theta = \frac{(\vec{a}).(\vec{b})}{|\vec{a}||(\vec{b})|} \quad \ldots(4)$$

Where $(\vec{a}).\left(\vec{b}\right) = \sum_1^n a_1 b_1 + a_2 b_2 + \cdots \ldots \ldots + a_n b_n$

The optimum similarity method for our framework must be selected from among the many similarity methods. To discover this, similarity checks on a set of sentences is performed, with each set containing a set of similar sentences
belonging to the same cluster. A study was conducted using two sets of sentences shown in table (1). Three similar approaches are used in the experiment: Jaccard, Dice, and Cosine. The Cosine similarity technique yields a greater number for a similar cluster group of sentences, as presented in Table (2). In the suggested framework, sentence similarity is computed using cosine similarity. Clustering is accomplished in four phases. Similarity calculation: From a corpus, a sentence is picked, and a similarity value for all the sentences with that sentence is calculated. Group Similar Sentences: If the similarity values of the sentences are greater than the threshold value, they are grouped. Identify Noise: This phase identifies sentences that do not belong to any of the cluster groups. The cluster ID is set to -1 for them. Create Cluster-ID Array: The clustered sentences are given a unique ID for each cluster group.

Table 1: Example sentences

Example	Sentence 1	Sentence 2
1	Eligibility	Eligibility for registration for ph.d. programmes
2	section i: Loss of or damage to the vehicle insured	section 1: Loss or damage

Table 2: Similarity values by various similarity methods

Similarity Method	Similarity values for Example 1	Similarity values Example 2
Jaccard	0.25	0.42
Dice	0.36	0.63
Cosine	**0.5**	**0.73**

3.4 Cluster Evaluation

Rand index, Completeness, Homogeneity, Adjusted Rand index, Vmeasure, Adjusted mutual information, and Fowlkes- Mallow's score is employed to evaluate clusters.

3.5. Haphazardly Dense Clustering with more Noise (HDCN)

Algorithm (1) shows Haphazardly Dense Clustering with more Noise (HDCN). The steps in this algorithm are as follows:

Step 1: Start at the arbitrary point which is not visited, and calculate the similarity score with all other non-visited data points.

Step 2: When the similarity score exceeds the limit, the data point is recognized as visited and added to the cluster.

Step 3: Repeat step 1 for all non-visited data points.

Step 4: Create a list of cluster ids for all data points and assign a value of -1 to those that are not clustered and identified as noise.

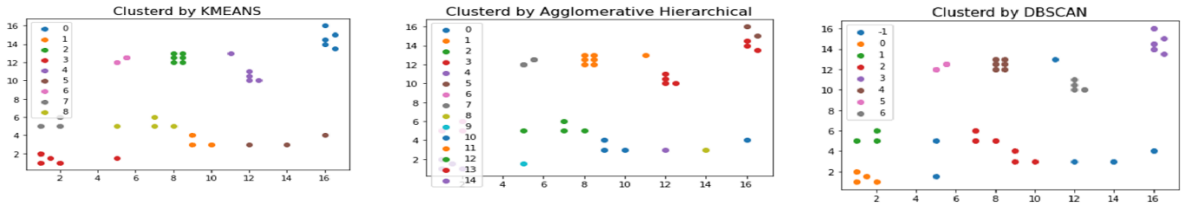
```
Input: X, TS                          /* X:Multidimensional data,TS:Threshold */
Output: CID                           /* ClusterIds */
Function Cluster_Text_Dense (X, TS)
    VL [];                            // VisitedList
    CL [];                            // CurrentList
    FL [];                            // FinalList
    CID [];                           // ClusterIds
    for x <-- 0 to len(X) do
        for y <-- x to len(X) do
            if y not in VL then
                if sim(x, y) > TS then
                    CL.append(y)
                    VL.append(y)
        if len(CL) > 1 then
            FL.append(x)
            CL:clear ()
    /* ClusterID Array Creation */
    for x <-- 0 to len(X) do
        if x not in VL then
            CID:append(-1)
        else
            CID:append(FL[x])
    return CID
```

Algorithm 1: Haphazardly Dense Clustering with more Noise (HDCN)

The difficulty with the suggested approach is deciding on a core data point. Choosing the core point is handled by sorting a text by the length in descending order. The Long sentences are chosen first as the core point.

3.6 Cluster result visualization:

The proposed algorithm is tested on randomly generated data points to visualise its performance. Figure (1) illustrated the Haphazardly dense data points with greater noise and the clusters are nearby. These data points are randomly generated points. The results of three state of art cluster algorithms are shown in the below Figure (3). The figure presents the results of applying the KMeans clustering algorithm (3a). The count of clusters is fixed to eight, however, the problem with this method is that all non-cluster points are allocated to clusters that are near together. The outcome of Agglomerative Hierarchical Clustering is shown in figure (3b), Cluster numbers are not assigned; instead, the distance value is used for clustering. It found a total of 15 clusters. However, the problem is that each non-cluster point is either allocated to a nearby cluster or a new cluster. Figure (3c) shows the result of DBSCAN clustering, the problem with this approach is that two nearby dense clusters are recognised as a single cluster. It discovered six clusters. Vectorized texts are used as input in the proposed technique. Instead of employing a number of groups, clustering is done using a distance measure. There were eight clusters discovered. The proposed method accurately identifies near clusters as two separate clusters, which is the fundamental difference between DBSCAN and the recommended technique. The results of the suggested procedure can be seen in the following Figure (4). Clusters 2 and 3 are recognised as a single cluster in DBSCAN but as two distinct clusters in the proposed technique. In DBSCAN, clusters 8 and 4 were likewise recognised as the same cluster, but in the proposed method, they were detected as separate clusters.



(a) KMeans (b) Agglomerative Hierarchical (c) DBSCAN

Fig 3: Clustered results

3.7 Performance analysis of DBSCAN and HDCN

HDCN is a clustering technique that is based on distance. Another distance-based clustering approach is DBSCAN. In HDCN, core points are no longer picked from inside the same cluster, and cosine similarity is employed again, instead of Euclidean distance. Because the points visited are not used as core it can manage the neighbouring two dense clusters and handle high-dimensional points. It is capable of dealing with a larger number of noise sources. It can handle haphazardly dense clusters.

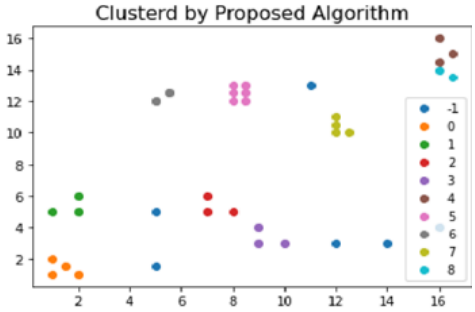


Fig 4: HDCN Result

4.0 Dataset Description

Data Set 1: It is a collection of titles extracted from Indian two-wheeler insurance policy paperwork. It has a total of 118 titles, 72 of which are part of one of the clusters. Others aren't clustered data points since they don't match any sentences. There are a total of 15 cluster groups. There are 119 distinct features that have been discovered.

Data Set 2: It's a collection of 849 titles collected from various Indian institutions' Notification of Ph.D. Admission papers. There are a total of 441 non-clustered locations. The remaining titles are grouped into one of the 119 clusters. There are 547 distinct features that have been discovered. Two data sets are used in the implementation.

The number of sentences in each corpus, the number of clusters, and the amount of noise are all listed in the table (3)

Table 3: Dataset

Data Set	Number of Sentences	Number of Clusters	Number of noise	Percentage of Noise
Insurance	118	15	54	46.55172414
Notification	849	119	441	51.9434629

5.0 Performance Evaluation

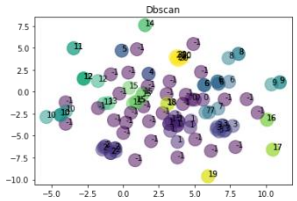


Fig 5: DBSCAN Clustering

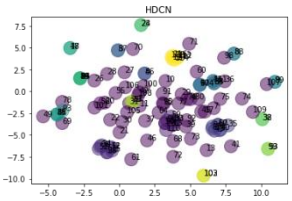


Fig 6: HDCN clustering

It is common to practise employing t-Distributed Stochastic Neighbor Embedding, a non-linear dimensionality reduction approach when investigating data sets with a large number of dimensions. Figure (5) shows the output for the insurance dataset by the HDCN algorithm. Figure (6) shows the output of the DBSCAN algorithm for the insurance dataset. The HDCN method clusters sentence more accurately than the DBSCAN algorithm, as shown in figures (5) and (6).

Metrics such as the Rand index, Completeness, Homogeneity, Adjusted Rand index, Vmeasure, Adjusted mutual information, and Fowlkes- Mallow's score is used to judging how well clustering algorithms work. The values of these evaluation metrics range from 0 to 1. And 1 indicates that the algorithm is completely accurate. As a result, the greater the value, the better the algorithm.

Rand index: The Rand index examines how comparable two different clustering algorithms' results are.[44] It is denoted by R. Equation (5) shows the formula to calculate the Rand index.

$$R = \frac{a+b}{nC2} \quad \ldots(5)$$

a: The amount of occasions a pair of items are in the same cluster across two clustering methods.
b: The amount of occasions a pair of items are in the distinct cluster across two clustering methods.
nC2: The amount of unordered pair relationships that exist inside a set that has n elements. Haphazardly given label values are not guaranteed to be near to zero, as shown by the Rand index. By establishing the modified Rand index, we could minimize the anticipated RI E[RI] of ARI.

Adjusted Rand index: The adjusted Rand index (ARI) is thus guaranteed to be near to 0.0 for random labelling, regardless of the number of clusters and samples, and to be exactly 1.0 when clustering is identical.[45]

$$ARI = \frac{RI - ExpectedRI}{\max(RI) - ExpectedRI} \quad \ldots(6)$$

Adjusted mutual information: The Adjusted Mutual Information (AMI) may then be determined using the predicted value in a like way to that of the modified Rand index:

$$AMI = \frac{MI - E(MI)}{mean\left(H(U), H(V)\right) - E(MI)} \quad \ldots(7)$$

Normalizing values are usually some kind of average of the probabilities of each clustering for both normalised and adjusted mutual information. There are many generalized ways to say something, but there are no hard and fast rules for choosing one over the others.

Homogeneity: The homogeneity criteria confirm that only individuals belonging to one class are present in each cluster. Homogeneity evaluates the extent to which each cluster consists of examples from a specific class of C. This is found by taking the conditional probability of the class distribution of the benchmark, H(C|K), and dividing it by the probability of the class distribution in the benchmark, H. (C).

$$h = 1 - \frac{H(C|K)}{H(c)} \quad \ldots(8)$$

Completeness: During a completeness check, all of the members of a certain class are put in the same cluster. Completeness measures how well each class fits into a single cluster. This is calculated as the conditional entropy of the cluster distribution found by the algorithm, H(K|C), divided by the cluster distribution's entropy, H. (K).

$$c = 1 - \frac{H(K|C)}{H(K)} \quad \ldots(9)$$

V measure: The quality of the clustering may be evaluated based on something called vmeasure, which is the harmonic mean of completeness and homogeneity [46].

$$V\beta b = \frac{(1+\beta b) * h * c}{(\beta b * h) + c} \quad \ldots(10)$$

Fowlkes-Mallow's score: Use of the Fowlkes-Mallows criterion The FMI is calculated by averaging the paired recall and precision: [47]

$$FMI = \frac{TP}{\sqrt{(TP+FP)*(TP+FN)}} \quad \ldots(11)$$

In this formula, TP represents the number of pairs of points that share the same clusters in both the true labels and the predicted labels, FP represents the number of pairs of points that share the same clusters in the true labels but not in the predicted labels, and FN represents the number of pairs of points that share the same clusters in the true labels but not in the predicted labels (i.e the number of sets of data points that are incorrectly placed in the same clusters by the predicted labels). [48]

5.1 Rand Index (RI), Adjusted Mutual Information (AMI), Adjusted Rand Index (ARI)

Table 4: RI, ARI, AMI Values using TF-IDF vectorization method

Dataset	Algorithm	Rand Index	Adjusted_rand score	Adjusted_mutual info score
Insurance	**HDCN**	**0.848037085**	**0.603157**	**0.682904**
	Kmeans	0.848906273	0.514849	0.652799
	Hierarchical	0.774735622	0.129931	0.410153
	DBSCAN	0.456757931	-0.08911	0.1926
Notification	**HDCN**	**0.980676**	**0.950135**	**0.96613**
	Kmeans	0.762329	0.264718	0.471121
	Hierarchical	0.725221	0.017558	0.284669
	DBSCAN	0.328452	-0.08308	0.047699

Table (4) shows the RI, ARI, AMI values for the Insurance and Notification dataset. The evaluation was done on Kmeans, Agglomerative, DBSCAN and HDCN algorithms. From table it is clear that HDCN performs well in two data sets.Table (5) shows the Adjusted rand Index, Adjusted mutual info score, Rand Index for the insurance and notification dataset wthe ith Word2Vec vectorization method.

Table 5: RI, ARI, AMI Values using Word2Vec vectorization

Dataset	Algorithm	Rand Index	Adjusted_rand score	Adjusted_mutual info score
Insurance	**HDCN**	**0.932928**	**0.793418**	**0.8409**
	Kmeans	0.798928	0.2137	0.521634
	Hierarchical	0.82935	0.374354	0.626054
	DBSCAN	0.88773	0.554313	0.744847
Notification	**HDCN**	**0.932928**	**0.793418**	**0.8409**
	Kmeans	0.798928	0.2137	0.521634
	Hierarchical	0.82935	0.374354	0.626054
	DBSCAN	0.88773	0.554313	0.744847

5.2 Completeness, Homogeneity, Vmeasure, Fowlkes mallows score

Table 6: Completeness, Homogeneity, Vmeasure, Fowlkes mallows score with TF-IDF vectorization

Dataset	Algorithm	Completeness	Homogeneity score	V measure Score	Fowlkes mallows score
Insurance	**HDCN**	**0.785094009**	0.753799	**0.769128**	**0.707448**
	Kmeans	0.698634836	0.821634	0.755159	0.617667
	Hierarchical	0.52292951	0.838779	0.644223	0.263629
	DBSCAN	0.464482645	**0.838779**	0.339766	0.274236
Notification	**HDCN**	**0.957404**	**0.998816**	**0.977671**	**0.963864**
	Kmeans	0.595712	0.744642	0.661903	0.419666
	Hierarchical	0.475089	0.760359	0.584789	0.103426
	DBSCAN	0.390223	0.760359	0.137492	0.399496

Table (6) shows the Homogeneity, Completeness, Vmeasure, Fowlkes-Mallow's score (FMI) values for the Insurance and Notification dataset with TF-IDF vectorization compared with Kmeans, Agglomerative, DBSCAN. From table (6) it is clear that HDCN performs well in two data set according to Completeness, V-Meaasure, Fowlkes mallow score. The table (7) shows the measures of Completeness, Homogeneity, V measure, Fowlkes mallow score of the insurance and Notification dataset with Word2Vec as the vectorization method.

Table 7: Completeness, Homogeneity, Vmeasure, Fowlkes mallows score with Word2Vec vectorization

Dataset	Algorithm	Completeness	Homogeneity score	V measure Score	Fowlkes mallows score
Insurance	**HDCN**	**0.907983**	0.874143	**0.890742**	**0.836429**
	Kmeans	0.645298	0.684402	0.664275	0.335151
	Hierarchical	0.73442	0.732995	0.733707	0.481011
	DBSCAN	0.766368	**0.91949**	0.835975	0.650279
Notification	**HDCN**	**1**	**0.994905117**	**0.995034015**	**0.997511**
	Kmeans	0.566360648	0.150907719	0.813426173	0.667774
	Hierarchical	0.575047996	0.173585549	0.822599961	0.676901
	DBSCAN	0.636723238	0.304127416	0.355131644	0.455955

Noise identification is only performed with DBSCAN and HDCN. Table (8) shows the noise identified by two algorithms using two vectorization methods. The DBSCAN algorithm detects 30.5 percent of noise, while the HDCN
method detects 96.675 percent. On a dataset with more noise, it demonstrates the superiority of the HDCN method over the DBSCAN technique.

Table 8: Noise Identified by DBSCAN and HDCN

Data set	Number of Sentences	Number of Noise	Noise identified by DBSCAN		Noise identified by HDCN	
			TF-IDF Vectorization	Word2Vec Vectorization	TF-IDF Vectorization	Word2Vec Vectorization
Insurance	118	54	13	32	**61**	**50**
Notification	849	441	41	133	**425**	**379**

5.3 Time Complexity

The time complexity of HDCN consists of two sections. Section 1 comprises of step 1 to 14, for cluster creation. Section 2 comprises of step 15 to 20, for ID creation. Let n be the number of sentences to be clustered. For Section 1, Best case is when all data points belong to one cluster. In the best situation, the time complexity is O(n). The time complexity of Section two is O. (n). Therefore, in the best situation, the time complexity is O(n + n).

For Section 1, Since at a time 2 sentences are compared to find the similarity therefore it is a problem of combination. The formula to calculate nCr is shown in equation (12).

$$nCr = \frac{n!}{r!(n-r)!} \quad \ldots(12)$$

Therefore, the time complexity for section 1 is $O({}_nC_2)$ which is $\frac{n^2}{2} - \frac{n}{2}$ at its average and worst case. For Section 2, the time complexity is O(n). Therefore, the total time complexity is $\frac{n^2}{2} - \frac{n}{2} + n$ at its worst and average case. Table (9) demonstrates the space and time complexity of several methods, where n- Number of sentences; m- Number of characteristics; k- Number of clusters; I -Number of iterations.

Table 13: Time and Space Complexity of clustering algorithms

Algorithm	Time Complexity	Space Complexity
K-means	*O (i * k * n * m)*	*O((n+k)m)*
Hierarchical	$O(n^2)$	$O(n^2)$
DBSCAN	$O(n^2)$	*O(n)*
HDCN	$O((n^2/2 - n/2) + n)$	*O(n)*

6. Conclusion

Text clustering [49] at the sentence level is the focus of research. First, there is a discussion of sentence clustering methods. Their benefits and drawbacks in generating clusters are carefully investigated. This work provided a unique text clustering approach for grouping sentences in a dataset with greater noise. In terms of assessment measures, the recommended HDCN method outperforms K-means, DBSCAN [50], and Hierarchical Agglomerative for data with more noise. It may be used on any dataset with a higher level of noise. Clustering is accomplished in four stages. The text is converted into vectors using TF-IDF. The similarity between the sentences is determined using cosine similarity. In terms of time and spatial complexity, the technique might be improved. The effectiveness has been evaluated and judged to be satisfactory.

Ethics statements

This research does not involve human subjects, animal experiments, not data collected from social media.

CRediT author statement

Both authors have made equal contributions to the invention and writing of this article.

Acknowledgments

Under the Junior Research Fellowship programme, University Grants Commission is providing financial assistance for this work (JRF).

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

References

[1] Huang et al., "Similarity measures for text document clustering," in Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand, vol. 4, pp. 9- 56, 2008.

[2] P. A. Hall and G. R. Dowling, "Approximate string matching," ACM computing surveys (CSUR), vol. 12, no. 4, pp. 381-402, 1980.

[3] J. L. Peterson, "Computer programs for detecting and correcting spelling errors," Communications of the ACM, vol. 23, no. 12, pp. 676-687, 1980.

[4] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida," Journal of the American Statistical Association, vol. 84, no. 406, pp. 414- 420, 1989.

[5] M. A. Jaro, "Probabilistic linkage of large public health data files," Statistics in medicine, vol. 14, no. 5-7, pp. 491- 498, 1995.

[6] W. E. Winkler, "String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage.," 1990.

[7] T. F. Smith, M. S. Waterman, et al., "Identification of common molecular subsequences," Journal of molecular biology, vol. 147, no. 1, pp. 195 - 197, 1981.

[8] 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 (Coling 2010), pp. 37- 45, 2010.

[9] L. Duan, L. Xu, Y. Liu, and J. Lee, "Cluster-based outlier detection," Annals of Operations Research, vol. 168, no. 1, pp. 151- 168, 2009.

[10] Hinneburg and D. A. Keim, "A general approach to clustering in large databases with noise," Knowledge and Information Systems, vol. 5, no. 4,pp. 387- 415, 2003.

[11] M. C. Thrun and Q. Stier, "Fundamental clustering algorithms suite," SoftwareX, vol. 13, p. 100642, 2021.

[12] O. A. Abbas, "Comparisons between data clustering algorithms.," International Arab Journal of Information Technology (IAJIT), vol. 5, no. 3, 2008.

[13] Nagpal, A. Jatain, and D. Gaur, "Review based on data clustering algorithms," in 2013 IEEE conference on information & communication technologies, pp. 298 - 303, IEEE, 2013.

[14] W. Zhang, T. Yoshida, X. Tang, and Q. Wang, "Text clustering using frequent itemsets," Knowledge-Based Systems, vol. 23, no. 5, pp. 379 - 388, 2010.

[15] W. Zhang, X. Tang, and T. Yoshida, "Tesc: An approach to text classification using semi-supervised clustering," Knowledge-Based Systems, vol. 75, pp. 152{160, 2015.

[16] K. Jain, "Data clustering: 50 years beyond k-means," Pattern recognition letters, vol. 31, no. 8, pp. 651- 666, 2010.

[17] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," IEEE transactions on pattern analysis and machine intelligence, vol. 24, no. 7, pp. 881- 892, 2002.

[18] P. S. Bradley and U. M. Fayyad, “Refining initial points for k-means clustering.," in ICML, vol. 98, pp. 91-99, Citeseer, 1998.

[19] T. M. Kodinariya and P. R. Makwana, “Review on determining number of cluster in k-means clustering,” International Journal, vol. 1, no. 6, pp. 90 - 95, 2013.

[20] Likas, N. Vlassis, and J. J. Verbeek, “The global k-means clustering algorithm,” Pattern recognition, vol. 36, no. 2, pp. 451- 461, 2003.

[21] Bholowalia and A. Kumar, “ Ebk-means: A clustering technique based on elbow method and k-means in wsn,” international Journal of Computer Applications, vol. 105, no. 9, 2014.

[22] M. Syakur, B. Khotimah, E. Rochman, and B. D. Satoto, “Integration kmeans clustering method and elbow method for identification of the best customer profile cluster,” in IOP Conference Series: Materials Science and Engineering, vol. 336, p. 012017, IOP Publishing, 2018.

[23] W. Arnold, “Uninformative parameters and model selection using akaike's information criterion,” The Journal of Wildlife Management, vol. 74, no. 6, pp. 1175 - 1178, 2010.

[24] K. P. Burnham and D. R. Anderson, “Practical use of the information theoretic approach,” in Model selection and inference, pp. 75- 117, Springer, 1998.

[25] M. Krieger and P. E. Green, “A cautionary note on using internal cross validation to select the number of clusters,” Psychometrika, vol. 64, no. 3, pp. 341- 353, 1999.

[26] W. Tang and T. M. Khoshgoftaar, “Noise identification with the k-means algorithm,” in 16th IEEE International Conference on Tools with Artificial Intelligence, pp. 373-378, IEEE, 2004.

[27] S.-S. Yu, S.-W. Chu, C.-M. Wang, Y.-K. Chan, and T.-C. Chang, “Two improved k-means algorithms,” Applied Soft Computing, vol. 68, pp. 747 - 755, 2018.

[28] Hautamaki, S. Cherednichenko, I. Karkkainen, T. Kinnunen, and P. Franti, “Improving k-means by outlier removal,” in Scandinavian Conference on Image Analysis, pp. 978- 987, Springer, 2005.

[29] G. Gan and M. K.-P. Ng, “K-means clustering with outlier removal,” Pattern Recognition Letters, vol. 90, pp. 8- 14, 2017.

[30] S. Shukla and S. Naganna, “A review on k-means data clustering approach,”International Journal of Information and Computation Technology, vol. 4, no. 17, pp. 1847-1860, 2014.

[31] M. Forina, C. Armanino, and V. Raggio, “Clustering with dendrograms on interpretation variables,” Analytica Chimica Acta, vol. 454, no. 1, pp. 13 - 19, 2002.

[32] W. S. Chan, W. T. Leung, and D. L. Lee, “Clustering search engine query log containing noisy clickthroughs,” in 2004 International Symposium on Applications and the Internet. Proceedings., pp. 305 - 308, IEEE, 2004.

[33] J. Almeida, L. Barbosa, A. Pais, and S. Formosinho, “Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering,” Chemometrics and Intelligent Laboratory Systems, vol. 87, no. 2, pp. 208 - 217, 2007.

[34] Sethy and B. Ramabhadran, “Bag-of-word normalized n-gram models,” in Ninth Annual Conference of the International Speech Communication Association, 2008.

[35] L. Al Qadi, H. El Rifai, S. Obaid, and A. Elnagar, “Arabic text classification of news articles using classical supervised classifiers,” in 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), pp. 1- 6, IEEE, 2019.

[36] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” Information processing & management, vol. 24, no. 5, pp. 513- 523, 1988.

[37] Aizawa, “An information-theoretic perspective of tf- idf measures,” Information Processing & Management, vol. 39, no. 1, pp. 45-65, 2003.

[38] R. Bartusiak, L. Augustyniak, T. Kajdanowicz, P. Kazienko, and M. Piasecki, “Wordnet2vec: Corpora agnostic word vectorization method,” Neurocomputing, vol. 326, pp. 141- 150, 2019.

[39] K. Park, J. S. Hong, and W. Kim, “ A methodology combining cosine similarity with classifier for text classification," Applied Artificial Intelligence, vol. 34, no. 5, pp. 396 - 411, 2020.

[40] J. Murthy, “Clustering based on cosine similarity measure,” 2012.

[41] R. Lahitani, A. E. Permanasari, and N. A. Setiawan, “Cosine similarity to determine similarity measure: Study case in online essay assessment,” in 2016 4th International Conference on Cyber and IT Service Management,pp. 1 - 6, IEEE, 2016.

[42] L. R. Dice, “Measures of the amount of ecologic association between species,” Ecology, vol. 26, no. 3, pp. 297 -302, 1945.

[43] P. Jaccard, “Etude comparative de la distribution florale dans une portion des alpes et des jura,” Bull Soc Vaudoise Sci Nat, vol. 37, pp. 547 - 579, 1901.

[44] J. M. Santos and M. Embrechts, “On the use of the adjusted rand index as a metric for evaluating supervised classification,” in International conference on artificial neural networks, pp. 175-184, Springer, 2009.

[45] P. Shrestha, C. Jacquin, and B. Daille, “Clustering short text and its evaluation,” in International Conference on Intelligent Text Processing and Computational Linguistics, pp. 169 - 180, Springer, 2012.

[46] Vlachos, A. Korhonen, and Z. Ghahramani, “ Unsupervised and constrained dirichlet process mixture models for verb clustering,” in Proceedings of the workshop on geometrical models of natural language semantics, pp. 74 - 82, 2009.

[47] M. Gholamian, S. Jahanpour, and S. Sadatrasoul, “A new method for clustering in credit scoring problems,” Journal of mathematics and computer Science, vol. 6, pp. 97 - 106, 2013.

[48] Y. Lu, Y.Wu, J. Liu, J. Li, and P. Zhang, “Understanding health care social media use from different stakeholder perspectives: a content analysis of an online health community,” Journal of medical Internet research, vol. 19, no. 4, p. e109, 2017.

[49] E. Laxmi Lydia, “CLUSTERING AND INDEXING OF MULTIPLE DOCUMENTS USING FEATURE EXTRACTION THROUGH APACHE HADOOP ON BIG DATA”, MJCS, pp. 108–123, Nov. 2020.

[50] S. O. Al-mamory and I. S. Kamil, “A NEW DENSITY BASED SAMPLING TO ENHANCE DBSCAN CLUSTERING ALGORITHM”, MJCS, vol. 32, no. 4, pp. 315–327, Oct. 2019.

[51] Ketcham, Mahasak, Thittaporn Ganokratanaa, and Nattapat Sridoung. "Classification of broadband network devices using text mining technique." *MethodsX* 11 (2023): 102346.