

## Analyzing semantic similarity amongst textual documents to suggest near duplicates

Viji Devarajan<sup>1,2</sup>, Revathy Subramanian<sup>3</sup>

<sup>1</sup>Department of Computer Science and Engineering, Faculty of Engineering and Technology, Sathyabama Institute of Science and Technology, Chennai, India

<sup>2</sup>Department of Computing Technologies, Faculty of Engineering and Technology, SRM Institute of Technology, Kattankulathur, India

<sup>3</sup>Department of Information Technology, Faculty of Engineering and Technology, Sathyabama Institute of Science and Technology, Chennai, India

### Article Info

#### Article history:

Received Aug 16, 2021

Revised Dec 18, 2021

Accepted Jan 11, 2022

#### Keywords:

BERT

Deep learning

GCN

Keyword extraction

Semantic-similarity

### ABSTRACT

Data deduplication techniques removing repeated or redundant data from the storage. In recent days, more data has been generated and stored in the storage environment. More redundant and semantically similar content of the data occupied in the storage environment due to this storage efficiency will be reduced and cost of the storage will be high. To overcome this problem, we proposed a method hybrid bidirectional encoder representation from transformers for text semantics using graph convolutional network hybrid bidirectional encoder representation from transformers (BERT) model for text semantics (HBTSG) word embedding-based deep learning model to identify near duplicates based on the semantic relationship between text documents. In this paper we hybridize the concepts of chunking and semantic analysis. The chunking process is carried out to split the documents into blocks. Next stage we identify the semantic relationship between documents using word embedding techniques. It combines the advantages of the chunking, feature extraction, and semantic relations to provide better results.

*This is an open access article under the [CC BY-SA](#) license.*



### Corresponding Author:

Viji Devarajan

Department of Computer Science and Engineering, Faculty of Engineering and Technology

Sathyabama Institute of Science and Technology

Chennai, India

Email: dviji2k@gmail.com

## 1. INTRODUCTION

In the digital world environment, everyone creates and uses digital data day by day. Due to this enormous amount of data generated and stored in the cloud environment. As per the statistics report of international data corporation (IDC) the data volume will reach 74 zettabytes by the end of 2021. The annual growth rate of digital data is 26 percent, so by the end of 2024 it will reach 149 zettabytes. This will be a tedious issue to maintain digital data in the future. The data deduplication technique is becoming a predominant methodology to maintain data in the cloud environment. Data deduplication removes duplicate content files and maintains unique content in the storage environment [1]-[3]. Data may be in a different format: text, images, audio, and video. Deduplication techniques will differ based on the data type. Majorly classified into two types text and multimedia data. Text data deduplication is carried out by file-level and content-based deduplication. File level deduplication working principle based on the file name, file type, and size of the file. Content-based deduplication splits the file into blocks. Blocks may be fixed or variable sizes. Deduplication achieves some highlighted points, it greatly increases the storage efficiency, network bandwidth, and decreases the cost. But to achieve these things deduplication needs extra resources to

calculate hash indexing and segmentation portion. Deduplication methodology performs based on the following features similarity and locality-based indexing [4], content-based inline deduplication [5], semantic aware deduplication [6], hash function [7], and record linkage deduplication [8] which all identifies duplicate content and reduces the storage space. Semantic similarity is calculated between words, or sentences, and documents.

Compared with word-level or sentence-level, the text at the document-level feature extraction gives better findings because document-level relationships contain a greater number of entities compared with sentence-level [9]. So, it becomes a harder task, to overcome this problem some pre-trained deep learning models are used. At the initial stage grouping similar content data using text clustering. Traditional algorithms work based on keyword and pattern matching, grouping similar documents in one cluster and non-similar documents in another cluster [10]. Semantic similarity is calculated by the distance between words by using cosine similarity [11]. Often, words may contain different meanings in two different sentences based on the context, but it is treated as both the sentences are similar.

The main contribution of this research work is i) fixed-size blocking mechanism was used to divide the documents into blocks to do the comparisons in the block wise manner, ii) keyword extraction done with the help of bidirectional encoder representation from transformers (BERT) and graph convolutional network (GCN) model, and iii) finally semantically similar documents are grouped together using K-means clustering algorithm. So, the proposed model blended the advantages of the chunking process, feature extraction and semantic relations.

Data deduplication generally focused on eliminating redundant content from the storage environment. The initial stage of research work focused on calculating fingerprints for blocks based on identifying repeated contents. Later focused on content-based deduplication methods like local maximum chunking (LMC), asymmetric extremum (AE), rapid asymmetric maximum algorithm (RAM), and parity check of interval (PCI) was proposed to avoid byte shifting problem and performance of chunking size [12].

Deduplication-assisted cloud-of-clouds (DAC) [13] uses data distribution in multiple independent cloud storage providers. DAC improves the performance and cost-efficiency significantly. Less security of cloud storage systems. Fast semantic duplicate detection [14] did automatic text data deduplication with French and English text in a particular region. But the classification doesn't select optimal features, increasing search complexity. Record linkage and various index methods performance have been compared in this survey [8]. Deduplication accuracy calculated only based on record level.

Major existing work focused on chunking or block size, and the indexing method. Based on this achieved deduplication on text data. There is a research gap in identifying the semantic relationships between text documents. In this paper, we focused on deduplication based on semantic relationships between documents. Semantic relationships are identified and applied in the following application in the previous days: text classification, document clustering, search engine queries, text summarization, and recommendation system.

Semantic relationships between documents identify similar content information between them. This can be done by the distance between terminology. In recent years natural language processing helps to calculate semantic similarity between words and sentences based on word co-occurrences, lexical database, or corpus. Word co-occurrence methods extract keywords from the documents based on that processing but, it does not take care about the word order of sentences and meaning of the word from the context. The lexical database contains the predefined word tree structure it represents the word, meaning, and relationship with other words. It identifies only best pair matching rather than identifying the exact meaning and also this method works based on the corpus. Corpus information may differ from one to another [15].

Word embedding techniques are more popular to find semantic similarities between documents. Early-stage latent semantic analysis (LSA) is used for word meaning identification. Later Word2vec became more popular than LSA because word2vec could handle large datasets. Meanwhile, LSA would be more suitable for a small-size corpus [16]. Zhou *et al.* [17] proposed a text similarity measure based on word vector distance decentralization input sentences were classified based on the labels and preprocessed by removing stop words, then fast semantic duplicate detection calculates a distance between word vectors to find similarities among sentences. Word vector distance decentralization (WVDD) performed well to identify the similarity of Chinese texts. WVDD is lagging to process long sentences and feature selection, a training set library not up to the level.

Ostendorff *et al.* [18] proposed a method to classify the semantic relationship between document pairs; they implemented six different word-embedding technologies. Global vectors for word representation (Glove), Doc2vec, deep contextual language models, BERT, XLNet, and Siamese architecture each method evaluated and explored in Wikipedia article pair dataset. The findings of this paper compared with vanilla transformers siamese architecture were not able to give good results in identifying semantic relations. vanilla transformers allow executing of two documents parallel. But transformers can use only 512 tokens from the text documents whereas, AvgGlove can evaluate entire text documents. The XLNet method can use lengthy sequences but it needs pretraining with long sequences.

A hybrid BERT model was proposed for multi-label text classification [19]. This model works based on four sub-task; first context-aware representation was developed using the word embedding technique (pre-trained BERT). The second module label graph embedding module focused on the semantic correlation between labels using GCN. GCN is used to represent finding the semantic relations among labels [20]. Each node is represented by labels, edges are represented by the semantic correlation between nodes. Third, the adjective attention module calculates a score between word and label. The fourth module aggregates the features from word embedding and label graph embedding and feeds into a bidirectional long short-term memory network (Bi-LSTM) for classification.

Sentence-level feature or relation extraction was easy compared with document-level. Document-level relation identification needs more effort since it contains many more entities. To resolve this issue document-level entity mask method with type information (DEMMT) was introduced by Han and Wang [9]. Each entity is masked by two tokens. The first token represents the type of entity and the second that entity that is linked to. BERT [21] encoder used to find the relationship among entities. The bilinear layer and softmax layer were used to identify relations of all entity pairs. DEMMT brings improvement results with all encoders like convolutional neural network (CNN), long short-term memory network (LSTM), Bi-LSTM, and BERT.

Reminder sections are organized as shown in: In section 2 we propose a method hybrid BERT model for text semantics (HBTSG) model. In section 3 we evaluated results and discussion. Finally, in section 4 we conclude our research work.

## 2. METHODOLOGY

In this section, we focused on identifying near duplicates based on semantic similarity between text documents. The proposed work consists of three sub-modules, in the first stage segmentation, documents are split into blocks, the second stage keyword extraction is done through GCN and word scoring method, and the last stage finds the distance relation between clusters. The workflow of the hybrid BERT model for text semantics using the GCN method is shown in Figure 1.

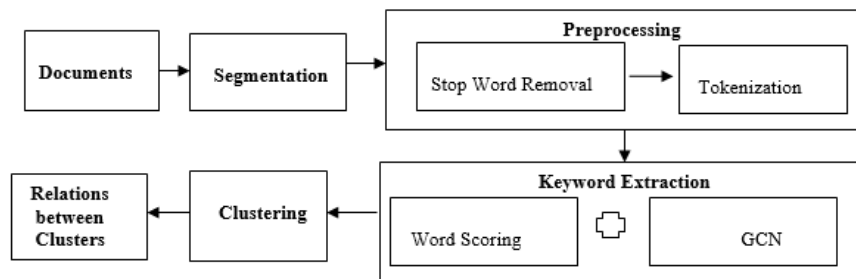


Figure 1. Research flow of HBTSG model

### 2.1. Segmentation

Files are segmented into several blocks, block-based deduplication is used to reduce redundancy levels and improve efficiency. Segmentation may occur in two ways: fixed size and variable size of blocks [3]. Fixed-size chunking process very simple to implement but lagging in byte shifting problem [12]. In our proposed method we are mainly focused on semantic relations between documents. So, we used the fixed chunking method for segmentation. The most common block size for the fixed chunking method is 4KB, it gives the optimal solution for deduplication as per the existing survey [12]. The documents are segmented into 4KB and this will be given as an input for the preprocessing.

### 2.2. Preprocessing

After segmentation, we need to do preprocessing to improve accuracy and efficiency. In general, for all data mining processes preprocessing is the first step. Good data preparation leads to efficient data analysis, eliminating errors and precise results during processing. Data preparation is also called data preprocessing. Data preparation is the job of cleaning and cleansing the raw data before processing and analysis. The main motive of this process makes our data ready to explore. Data preparation is a quite lengthy process, but it is essential for further processing. Usually, data preparation includes regulating the data formats, enhancing the raw data, and or getting rid of outliers with the help of data frames. We did stop word removal and tokenization using a natural language toolkit (NLTK). Stop words are in English for example is, I, an, and are. These words are not giving important features. If we remove stop words we can focus on the exact

feature and dataset size will be reduced greatly. Table 1 shows a sample of before and after stop word removal. Tokenization is the process of splitting sentences into individual words or terms that is also called tokens.

Table 1. Sample for without stop word

Sample Text	Without Stop words
I like food, so I am cooking	Like, food, cooking
He is studying computer science	Studying, computer science

### 2.3. Keyword extraction

In the second stage of work, keyword extraction can be done GCN, and word scoring based on semantic relationships using a BERT encoder. Word embedding is converting words into numbers by using one-hot encoding. In this method, all the words are treated as one feature of the vector i.e. each word represented by one column. The limitation of one-hot encoding method is if it contains more number of words then, it has to form more number of columns. As a result, this will create many zeros entry, it will create a sparse matrix shown in Table 2. It will be rigid to handle sparse matrices in a machine learning algorithm.

Table 2. One-hot representation of word

	I	am	doing	good
I	1	0	0	0
am	0	1	0	0
doing	0	0	1	0
good	0	0	0	1

If the sentence contains more similar words, those words are also treated as two different words. For example, good and great are more or less similar but in one hot representation it will be considered as two different words. Here there are no findings on the semantic relationship between words. So, the word2vec embedding algorithm was introduced [22]. Word2vec neural network model compares two vectors using cosine similarity. The main goal of word embeddings is to reduce the dimensionality and inter-word semantics must be captured. Word2vec and LSA does not account for co-occurrence statistics [23]. Global vectors for word representation (Glove) aim to improve context captured and co-occurrence probabilities. Its embedding relates to probabilities that two words appear together. Glove count-based models learn their vectors by dimensionality reduction on co-occurrence [24]. The glove does not rely on local context information of words [25]. Word2vec and Glove are context-independent. It combines all the different senses of words into one vector. Embeddings from language models (Elmo), and BERT are context-dependent. These models can generate a vector for words based on context. BERT generates its embedding differently compared with other word embedding methods [21]. Table 3 shows the comparison of characteristics between word embedding models. BERT takes the information forward and backward and combines it. BERT model used in the following applications: neural machine translation, question and answering, sentiment analysis, and text summarization.

Table 3. Different word embedding models

Word Embedding Models	Context Sensitive	Learnt representation
Word2vec	No	Words
Glove	No	Words
Elmo	Yes	Character-based
BERT	Yes	Sub words

#### 2.3.1. Word scoring based on semantic relationships

Word score calculated through the weight of edges, initially word score calculated based on word co-occurrence. Word score is based on word co-occurrence calculated by the weight of edges, but two words do not appear in the same window even though they have similar meanings. This limitation can be overcome through a word score calculated based on semantic relations. Many different ways we can calculate semantic relationships between words. In this paper, we calculated through the Word embedding method based on the pre-trained BERT model used for input features.

BERT consists of three layers as input layer, BERT encoder, output layer. First input layer, sentences split into word set and indicated with ##. Second layer BERT encoder contains transformer blocks

and self-attention head added with input sequences. Structure of BERT encoder shown in Figure 2. Top of the BERT contains a SoftMax classifier to calculate conditional probability on predefined labels.

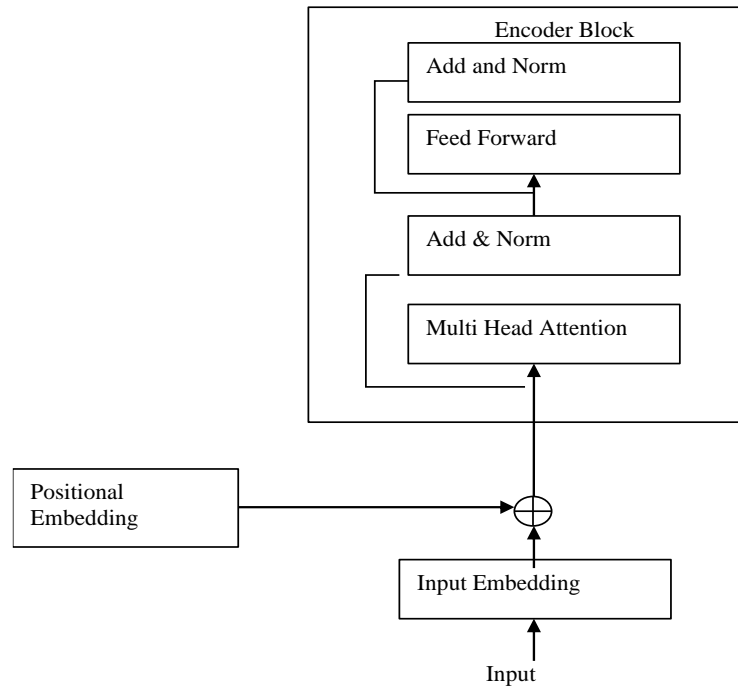


Figure 2. Structure of input encoder

Let assume  $A$  will be an input sequence containing  $n$  words, denoted as  $A_{1:n} = A_1, A_2, A_3, \dots, A_n$ , where  $A_i (1 \leq A_i \leq n)$  refers to the  $i^{th}$  word in the sequence. Input sequences are constructed without or with auxiliary sentences for example “He wrote the exam” with a label mentioned from the set of {“positive”, “negative”}. Sample input sequence construction mentioned in Table 4. The pseudo-sentence is made up of categorical labels and other words like,  $[CLS] A_1 \dots A_i, \dots A_n [SEP] a_1 \dots a_j \dots a_m [SEP]$ . The target labels are represented as  $\{0,1\}$  in the BERT4TC-AQ and BERT4TC-AA models [26]. The above models every input statement not containing more than 512 tokens. If the input sentence pairs  $(A_{1:n}, a_{1:m})$  fulfill the condition  $n+m+3 > 512$  means then only at most 509 tokens can be kept. Where the constant ‘3’ indicates one [CLS] token plus two [SEP] tokens. So, 512 tokens – 3 tokens ( $[CLS], [SEP], [SEP]$ ) = 509 tokens. The conditions specified for pretreatment in (1).

$$(A_{1:n}, a_{1:m}) = \begin{cases} [CLS]A_1 \dots A_n [SEP] a_1 \dots a_m [SEP] & \text{if } n + m < 509 \\ [CLS]A_1 \dots A_{n+m-509} [SEP] a_1 \dots a_m [SEP] & \text{if } n + m \geq 509 \end{cases} \quad (1)$$

Table 4. Construction of input sequence

Input Sequence	Label
[CLS] He wrote the exam [SEP]	{ <b>positive</b> , negative}
[CLS] He wrote the exam [SEP] What is the result [SEP]	{ <b>positive</b> , negative}
[CLS] He wrote the exam [SEP] The result is positive [SEP]	{ <b>1</b> , 0}
[CLS] He wrote the exam [SEP] The result is negative [SEP]	{1, <b>0</b> }

Computers don't know words; they can know numbers and vectors. Input embedding to map all words physically close to each other. The same word has different meanings in different sentences based on the position. To overcome this positional embedding aggregated with input embedding. A positional embedding vector gives a context-based position of the word in the sentence. For example, Sentence 1 Apple releases a new version. Sentence 2 Daily eat one apple. Here the word apple is the same but it gives a different meaning based on the position in the sentences. Positional embedding used the Sin, Cos function to generate a vector. Encoder blocks contain feed-forward and multi-head attention blocks. Feedforward transform attention vector to blocks. How does it work relevant to other words in the same sentence? This

will be resolved using an attention vector for every word generating attention vector based on the context relationship between words in the sentence. Additionally, we add normalization for each layer to make normalization across each feature instead of each sample. The stack of input encoder blocks used in BERT. BERT feature extraction explained in algorithm 1.

---

**Algorithm 1: BERT feature extraction**


---

```

Input  $R(f)$ : Set consisting of input sequence  $f$ .
S( $f$ ): Similarity scores set related to input sentence  $f$ .
Output  $J$ : Extractions Set. Each  $j \in J$  denotes the tuple of (feature, similarity scores)
pre-processing the data;
for  $s \in S(s)$  do
  Remove  $s$ ;
  Tag the Features with the Pre-trained BERT model  $s$ ;
end
reconfigure;
 $J = \alpha$ 
for  $s \in S(f)$  do
for each and every sentence in  $s$  do
   $M =$  BERT Tag nouns of  $s$ ;
  for  $r \in R(f)$  do
     $N = rUM$ ;
     $RES =$  Original Features - BERT Features ;
     $j = (N, RES)$ ;
    Add  $j$  to  $J$  ;
  end
end
end
 $J =$ Feature Extracted Normalized Data;
return  $J$ ;

```

---

### 2.3.2. Graph embedding

Graph embedding technology converts graphs into lower-dimensional before sending them into a machine learning algorithm. That transforms nodes, edges, and relations (features) into vector space. Graph analytics process majorly used in the following area: node classification used to find a label of nodes, link prediction to predict missing link and future occurrence, clustering used to identify same node type and group them, visualization improves the structure of the network [27].

### 2.3.3. Deep learning-based model

Deep neural network-based models structural deep network embedding (SDNE) and deep neural networks for graph representation (DNNGR) are expensive and not efficient for large sparse graphs. Graph convolutional networks overcome this problem [20]. In recent years GCN was used in most of the research work to specify unique labels for all nodes. It gives overall structure information of the graph, especially semantic relations among labels from the context. As shown in (2) [20] layer-wise propagation rule used in the multi-layer GCN.

$$H^{(l+1)} = \sigma \left( \tilde{D} - \frac{1}{2} \tilde{A} \tilde{D} - \frac{1}{2} H^{(l)} W^{(l)} \right) \quad (2)$$

where,

- $\tilde{A} = A + I_N$ , is an adjacency matrix of graph G.
- $I_N$ , is an identity matrix.
- $\tilde{D}_{ii} = \sum_j \tilde{A}_{i,j}$  and  $W^{(l)}$ , is a layer specific trainable weight matrix.
- $\sigma(\cdot)$ , is an activation function. i.e., [ ReLU ( $\cdot$ ) = max(0,  $\cdot$ ) ]

In two-layer GCN, node classification on a graph with adjacency matrix A. First in preprocessing stage calculate the value for  $\hat{A} = \tilde{D} - \frac{1}{2} \tilde{A} \tilde{D} - \frac{1}{2}$ .

$$Z = f(X, A) = \text{softmax}(\hat{A} \text{ReLU}(\hat{A} X W^{(0)}) W^{(1)}) \quad (3)$$

As shown in (3) [20] explains the simple form of forward model. Figure 3 explains the process of GCN, C indicates input channels and F denotes features,  $Y_i$  denotes labels, edges are represented by black lines in the Figure 3. GCN collects the values of all neighboring nodes to evaluate the current node. ReLU activation function used here [19], [27].

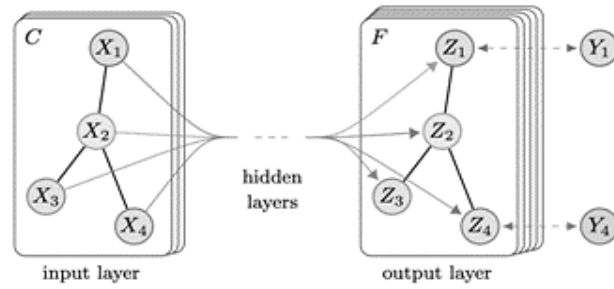


Figure 3. Graph convolutional network

**2.3.4. Clustering**

The clustering concept is used to group the objects based on similar characteristics. Document clustering or text grouping is majorly focused based on grouping the documents based on similar content or the same category of the files. Similarity can be identified using semantic relations between documents through natural language processing techniques. Keyword extraction is done by word scoring using the BERT and GCN model, then the documents are clustered by using the K-mean algorithm. K-means clustering algorithm most popular algorithm for clustering the documents based on semantic similarity [28], [29].

The K-mean algorithm is simple and faster than others [30], some research papers used bisecting K-mean it can work with large dataset and accuracy was improved. To finetune our results after grouping the similar content of documents, we calculated the relations between clusters based on the cosine distance vector.

**3. RESULTS AND DISCUSSION**

In this section, discussion on datasets and evaluating the effect of the proposed method. Comparisons made among existing semantic relations identification techniques. Inspc dataset are used for the collection of abstracts. Inspc dataset was first introduced by Hulth [31] in 2003. Inspc dataset about the collection of abstracts nearly 2000 abstracts are available in this. Hulth divided that into three parts: 1000 training datasets, 500 validation datasets, 500 testing datasets. Reference keywords from the corpus 3829. DUC01 dataset was released in 2008 by Wan and Xiao [32]. These dataset collections of news articles total 308 articles are available. reference keywords from the corpus are 2488. BibTxt dataset [33] contains a large number of articles downloaded from the web. The dataset contains a total of 4387 BibTxt files with more than 6 million article records available.

**3.1. Evaluation measures**

The most common evaluation measures are purity and normalized mutual information (NMI). These techniques are used for clusters to check the correctness of each cluster. Cluster represented as  $C = \{C1, C2, C3, \dots, Cj\}$  and partitioned,  $P = \{P1, P2, P3, \dots, Pi\}$  such as  $i$ , and  $j$  represented the number of cluster and cluster classes. Clusters can achieve high purity shown in (4), and NMI calculated between pairs of clusters and individual classes.

$$purity(C, P) = 1/N \sum \max |Cj \cap Pi| \tag{4}$$

For keyword extraction evaluation measure calculated by precision, recall, and F1 measure shown in (5)-(7):

$$P = |K \cap MK| / |K| \tag{5}$$

$$R = |K \cap MK| / |MK| \tag{6}$$

$$F1 = 2PR / P + R \tag{7}$$

where the P represents a precision measure, K represents an extracted keyword, and MK represents the manually assigned keyword. R denoted as recall.

**3.2. Analysis**

From the experiment results precision, recall, and F1 Measure values in Table 5. Table 6 described a comparison among three different datasets with semantic relations technologies like third combination normalized google distance (TCNGD), probabilistic feature patterns (PFP), and our method hybrid BERT model for text semantics using gcn (HBTSG). Figures 4 and 5 represent the same described in Table 5.

Table 5. Comparison of keyword extraction result

Dataset Method	Inspec			BipTxt			DUC01		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
TCNGD	0.391	0.752	0.501	0.401	0.759	0.522	0.292	0.374	0.322
PFP	0.395	0.771	0.532	0.411	0.791	0.552	0.333	0.392	0.363
HBTSG (Proposed)	0.411	0.801	0.551	0.423	0.834	0.593	0.354		0.396
								0.394	

Table 6. Analysis of accuracy

Accuracy %	TCNGD	PFP	HBTSG
Inspec	86	87	90
BipTxt	84	85	87
DUC01	82	83	85

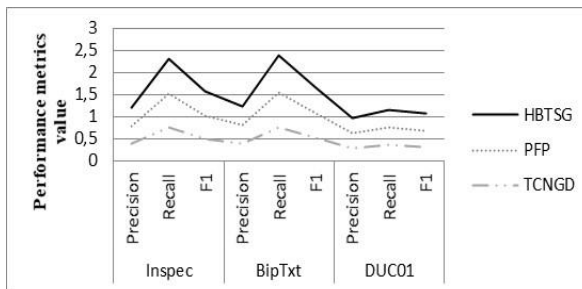


Figure 4. Keyword extraction evaluation measure

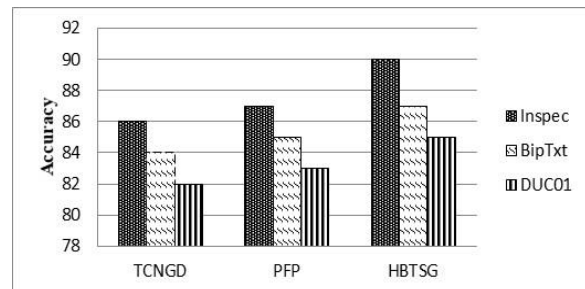


Figure 5. Analysis of accuracy

#### 4. CONCLUSION

In this paper, we focused on identifying near duplicates based on semantic relations between text documents using NLP techniques. Various methods discussed semantic relations identification. Documents are split into fixed blocks 4KB then preprocessing done. In the third stage, keyword extraction is done through a combination of the BERT and GCN models. This hybrid method gives better keyword extraction. Further similar contents are grouped using the clustering technique, finally the distance calculated between clusters to get fine-tuned results of semantic similarity between text documents. Through this deduplication can be done easily and it will give great results to identify similar content files stored in the storage environment.

#### REFERENCES




- [1] R. Kaur, I. Chana, and J. Bhattacharya, "Data deduplication techniques for efficient cloud storage management: a systematic review," *The Journal of Supercomputing*, vol. 74, no. 5, pp. 2035–2085, May 2018, doi: 10.1007/s11227-017-2210-8.
- [2] J. Paulo and J. Pereira, "A Survey and Classification of Storage Deduplication Systems," *ACM Computing Surveys*, vol. 47, no. 1, pp. 1–30, Jul. 2014, doi: 10.1145/2611778.
- [3] D. Viji and S. Revathy, "Various Data Deduplication Techniques of Primary Storage," in *2019 International Conference on Communication and Electronics Systems (ICCES)*, Jul. 2019, pp. 322–327, doi: 10.1109/ICCES45898.2019.9002185.
- [4] W. Xia, H. Jiang, D. Feng, and Y. Hua, "Similarity and Locality Based Indexing for High Performance Data Deduplication," *IEEE Transactions on Computers*, vol. 64, no. 4, pp. 1162–1176, Apr. 2015, doi: 10.1109/TC.2014.2308181.
- [5] A. Khan, P. Hamandawana, and Y. Kim, "A Content Fingerprint-Based Cluster-Wide Inline Deduplication for Shared-Nothing Storage Systems," *IEEE Access*, vol. 8, pp. 209163–209180, 2020, doi: 10.1109/ACCESS.2020.3039056.
- [6] Y. Tan, H. Jiang, D. Feng, L. Tian, Z. Yan, and G. Zhou, "SAM: A Semantic-Aware Multi-tiered Source De-duplication Framework for Cloud Backup," in *2010 39th International Conference on Parallel Processing*, Sep. 2010, pp. 614–623, doi: 10.1109/ICPP.2010.69.
- [7] A. T. Clements, I. Ahmad, M. Vilayannur, and J. Li, "Decentralized deduplication in SAN cluster file systems," *USENIX Annual Technical Conference*, vol. 9, pp. 101–114, 2009.
- [8] P. Christen, "A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 9, pp. 1537–1555, Sep. 2012, doi: 10.1109/TKDE.2011.127.
- [9] X. Han and L. Wang, "A Novel Document-Level Relation Extraction Method Based on BERT and Entity Information," *IEEE Access*, vol. 8, pp. 96912–96919, 2020, doi: 10.1109/ACCESS.2020.2996642.
- [10] R. K. Ibrahim, S. R. M. Zeebaree, and K. F. S. Jacksi, "Survey on Semantic Similarity Based on Document Clustering," *Advances in Science, Technology and Engineering Systems Journal*, vol. 4, no. 5, pp. 115–122, 2019, doi: 10.25046/aj040515.
- [11] S. Tongphu, "Toward semantic similarity measure between concepts in an ontology," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, no. 3, p. 1356, Jun. 2019, doi: 10.11591/ijeecs.v14.i3.pp1356-1372.
- [12] D. Viji and D. S. Revathy, "Comparative Analysis for Content Defined Chunking Algorithms in Data Deduplication," *Webology*, vol. 18, no. 2, pp. 255–268, Apr. 2021, doi: 10.14704/WEB/V18SI02/WEB18070.






- [13] S. Wu, K.-C. Li, B. Mao, and M. Liao, "DAC: Improving storage availability with Deduplication-Assisted Cloud-of-Clouds," *Future Generation Computer Systems*, vol. 74, pp. 190–198, Sep. 2017, doi: 10.1016/j.future.2016.02.001.
- [14] I. M. Nguena and A.-M. O. C. Richeline, "Fast Semantic Duplicate Detection Techniques in Databases," *Journal of Software Engineering and Applications*, vol. 10, no. 06, pp. 529–545, 2017, doi: 10.4236/jsea.2017.106029.
- [15] A. Pawar and V. Mago, "Calculating the similarity between words and sentences using a lexical database and corpus statistics," Feb. 2018, [Online]. Available: <http://arxiv.org/abs/1802.05667>.
- [16] E. Altszyler, M. Sigman, S. Ribeiro, and D. F. Slezak, "Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database," *Consciousness and Cognition*, vol. 56, pp. 178–187, Oct. 2016, doi: 10.1016/j.concog.2017.09.004.
- [17] S. Zhou, X. Xu, Y. Liu, R. Chang, and Y. Xiao, "Text Similarity Measurement of Semantic Cognition Based on Word Vector Distance Decentralization With Clustering Analysis," *IEEE Access*, vol. 7, pp. 107247–107258, 2019, doi: 10.1109/ACCESS.2019.2932334.
- [18] M. Ostendorff, T. Ruas, M. Schubotz, G. Rehm, and B. Gipp, "Pairwise multi-class document classification for semantic relations between wikipedia articles," *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, Mar. 2020, pp. 127–136, doi: 10.1145/3383583.3398525.
- [19] L. Cai, Y. Song, T. Liu, and K. Zhang, "A Hybrid BERT Model That Incorporates Label Semantics via Adjustive Attention for Multi-Label Text Classification," *IEEE Access*, vol. 8, pp. 152183–152192, 2020, doi: 10.1109/ACCESS.2020.3017382.
- [20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, Sep. 2017, [Online]. Available: <http://arxiv.org/abs/1609.02907>.
- [21] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, pp. 4171–4186, Oct. 2019.
- [22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Jan. 2013, [Online]. Available: <http://arxiv.org/abs/1301.3781>.
- [23] A. Rozeva and S. Zerkova, "Assessing semantic similarity of texts - Methods and algorithms," in *AIP Conference Proceedings*, vol. 1910, 2017, p. 060012, doi: 10.1063/1.5014006.
- [24] S. M. Mohammed, K. Jacksi, and S. R. M. Zeebaree, "A state-of-the-art survey on semantic similarity for document clustering using GloVe and density-based algorithms," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 1, p. 552, Apr. 2021, doi: 10.11591/ijeecs.v22.i1.pp552-562.
- [25] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014, pp. 1532–1543, doi: 10.3115/v1/d14-1162.
- [26] S. Yu, J. Su, and D. Luo, "Improving BERT-Based Text Classification with Auxiliary Sentence and Domain Knowledge," *IEEE Access*, vol. 7, pp. 176600–176612, 2019, doi: 10.1109/ACCESS.2019.2953990.
- [27] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, May 2018, doi: 10.1016/j.knosys.2018.03.022.
- [28] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," *Proceedings of the International KDD Workshop on Text Mining*, 2000.
- [29] S. Sharma and R. K. Gupta, "Improved BSP Clustering Algorithm for Social Network Analysis," *International Journal of Grid and Distributed Computing*, vol. 3, no. 3, pp. 67–76, 2010.
- [30] M. Fariss, N. El Allali, H. Asaidi, and M. Bellouki, "A semantic web services discovery approach integrating multiple similarity measures and k-means clustering," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 2, p. 1228, Nov. 2021, doi: 10.11591/ijeecs.v24.i2.pp1228-1237.
- [31] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, vol. 10, 2003, pp. 216–223, doi: 10.3115/1119355.1119383.
- [32] X. Wan and J. Xiao, "Single Document Keyphrase Extraction Using Neighborhood Knowledge," in *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, 2008, pp. 855–860, doi: 10.5555/1620163.1620205.
- [33] "BibText dataset," *IESL*. <https://sites.google.com/a/iesl.cs.umass.edu/home/data/bibtex> (accessed Feb. 18, 2020).

## BIOGRAPHIES OF AUTHORS



**Viji Devarajan**    received her Bachelor of Engineering degree in Computer Science and Engineering in 2010. She also received her Master of Engineering Degree in Computer Science and Engineering from Adhi parasakthi engineering Engineering College, Melmaruvathur in 2015. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai India. She is currently an Assistant Professor in the Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai India. Her work includes over 19 Journal publications and 21 conference publications within her working experience of 5 years so far. She can be contacted at email: [dviji2k@gmail.com](mailto:dviji2k@gmail.com).



**Dr. Revathy Subramanian**    is presently working as an Associate Professor in the Department of Information Technology, Sathyabama Institute of Science and Technology, Chennai India. Her research interest includes Machine Learning, Data Analytics and Big Data. She has published over 41 papers in refereed journals. She can be contacted at email: [ramesh.revathy@gmail.com](mailto:ramesh.revathy@gmail.com).