ISSN: 2229-6956 (ONLINE)
DOI: 10.21917/ijsc.2025.0509

SEMANTIC BASED EXTRACTIVE DOCUMENT SUMMARIZATION USING DEEP LEARNING MODEL

S. Divya[1] and N. Sripriya[2]
[1]*Department of Computer Science and Engineering, Shiv Nadar University, India*
[2]*Department of Information Technology, Sri Sivasubramaniya Nadar College of Engineering, India*

Abstract

The rapid growth of web documents led to the entailment of automatic document summaries. Extractive summarization designates certain principle features from the input document and groups them together to generate a summary. This empowers readers to quickly browse the document and unveil the information in it. The focus of this work is to propose a clustering algorithm that suits for the summarization of both Tamil and English documents. Transformer mechanism that is trained on 104 languages (which includes Tamil and English language) is used to represent each sentence in the source document as features in the high dimensional space. Feature vectors are exposed to clustering with a notion of ignoring outliers and group similar features. A hybrid clustering algorithm is proposed to generate efficient clustering that aims in forming clusters that are densely coupled and massive clusters are divided as sub-clusters to facilitate sentence selection from each cluster. An identical number of sentences are picked from each cluster/sub-clusters and are included in the summary until the summary size outreaches the threshold. The performance of the proposed clustering algorithm is evaluated on both Tamil and English document. The proposed clustering algorithm is applied on the CNN/DailyMail dataset and is evaluated in terms of ROUGE metrics. In addition to this, the summary generated for the Tamil documents are shared with readers for evaluating based on the reader's perspective. ROUGE and the Mean Opinion Score prove that the clusters generated by the proposed model are well-organized and the summary is precise and informative. The proposed summarization model outperforms existing Tamil text summarization models.

Keywords:

Extractive Summarization, Hybrid Clustering, Effective, Summary, Tamil Text Summarization

1. INTRODUCTION

Documents such as news, education, technology-based, unveil the truth or facts of a certain domain in a detailed manner. The word "Document" which originated from the Latin term "Documentum" stands for "teaching" or "lesson". In general, a document entitles uncondensed knowledge about a specific domain. This contains both the core knowledge as well as its supplementary information in a detailed manner. Each document elaborates unique information about a specific area along with its origin, relevancy with other domains and future scope. The major challenge in understanding a document is to quickly perceive its core content. The eruption of data generation due to the growth of Internet and Web applications resulted in manual summarization being quite impossible.

This detonation of documents can be solved by applying Natural Language Processing technique called Automatic Text Summarization [1] which helps to rapidly conceive massive text content by creating an explicit and logical summary. Thus, for any adjudicative and knowledge acquisition, summarization technique is decisive and sensible in this fast-expanding world. Despite an ancient issue and the primary attempts of extraction of features in text summarization being carried out from 1950s, it is still a required research field because of its diverse applicability. The initial feature extraction models extracted features such as the frequency of words and phrases to identify significant sentences.

Summary generation can be performed in two styles: extractive and abstractive summarization techniques. Extractive Summarization [2], [3] is a conventional approach, where the key phrases are identified to select significant sentences from the vast text document and are included in the summary. Abstractive Summarization [2], [4] is an advanced method, which detects the important phrases, transcribes information and delivers the core information of the huge text document into a precise summary. Extractive summarization technique tends to be a simple, flexible, time-saving and self-regulating mechanism for generating a summary. Extractive summarization technique retains the structure of the input sentences to remain the same in the summary.

Among various types of techniques in extractive summarization, machine learning techniques have been widely utilized to select informative sentences and generate extractive summary. This can be either handled as a supervised or unsupervised problem. Unsupervised extractive summarization is implemented by representing each input word/sentence in an intermediate form that enumerates the recognized features. This representation encapsulates the connotation of the input and plotting those conceptually homogenous inputs are placed nearer in the high-dimensional space. This will assist in determining how far each sentence influences and contributes to the core content of a document. Sentences that are densely located tend to contain significant content. The clustering technique may be applied to this representation to group the sentences of related semantics. From each group of relevant sentences, based on their similarity measure, sentences are decided whether to be included or excluded from the summary.

The key challenge in the unsupervised summarization model is the effective extraction of features that majorly contributes to the identification of significant and similar sentences. In summarization, Clustering algorithm intends to form clusters by grouping the features and neglecting irrelevant information. Clusters should preferably be compact, well-dissociated and semantically similar. Generally, sentences that hold relevant information tend to lie in the same cluster. Since a single document conveys information in a particular context, its features are placed closer in the n-dimensional space, leading to the formation of larger clusters. Sentence selection becomes complex when the clusters are largely distinct. Thus, feature extraction and cluster generation make substantial contributions in effective retrieval of sentences to generate a precise, informative and understandable summary.

The salient objective of the proposed hybrid clustering summarization model is to effectively group the features that represent each sentence in the document. These are potentially grouped to generate an informative and smooth summary. The significant contributions of the proposed model are listed below.

- An efficient intermediate representation for each sentence in the document is generated using an embedding technique.
- The conceptually irrelevant sentences are identified and ignored as noise and forms clusters with semantically similar sentences.
- Sentence selection is facilitated by fragmenting the larger clusters into sub-clusters of identical sizes.
- An efficient sentence selection approach that eliminates redundancy is proposed to fetch sentences from each cluster equally to generate a summary of desired size.

In the proposed hybrid clustering model, text embedding is performed using Multilingual Bidirectional Encoder Representations from Transformers (MBERT), which is an unsupervised learning architecture constructed over Transformer architecture. Initially clustering algorithm is used to acquire dense portions that hold similar sentences and to remove noise. Once the dense portions are recognized, clusters are formed. Further, sub-clusters are derived from dense clusters to reveal the fragments of information. The estimation of parameters in the clustering algorithm is practiced by incrementing them based on specific conditions for effective cluster formation. A novel sentence selection method is proposed to evenly fetch sentences from each cluster/sub-cluster, which is governed based on the size of the required summary. The performance of the summarization model is evaluated upon the twenty news documents from CNN Daily/Mail dataset that are translated into Tamil.

The rest of the paper is assembled as given below. Recent studies on various Embedding techniques and several clustering algorithms that contribute to text summarization are reviewed in section 2. Section 3 elucidates each step for performing unsupervised text summarization. Performance evaluation of the proposed clustering algorithm, experimental results and the summary generation is populated in section 4. Section 5 concludes the work including inherent directions towards further system improvement.

2. BACKGROUND AND RELATED WORK

To facilitate the summary generation, extractive summarization [6], [7] has two phases, pre-processing and processing. The pre-processing phase [7] cleanses the document to extract input sentences by using a few steps like stop word removal, stemming, etc. In the processing phase, the weights of each sentence [7] in the document are estimated. Weights are determined using numerous features that are organized as Statistical and Linguistic features [8]. The informative term, the position of a sentence, frequency of a term, parts of speech tag are some instances of statistical features that hold numerical value, which is consolidated to derive the weights of each sentence. Linguistic features associate the terms with their semantics based on their characteristics. Embedding renders a constructive representation of words or sentences by conserving theoretical coincidence of words along with construction of n-dimensional vectors. The two most common models for generating representations are Vector Space Model (VSM) [9] and Latent Semantic Analysis (LSA) [9]. The significance of each sentence to the whole document is computed and is represented as a vector using VSM. LSA populates each input sentence into a latent semantic space by creating a word-by-sentence matrix termed as Singular Value Decomposition (SVD). As an additional functionality to VSM, Markov Random Walk (MRW) [10], Maximum Marginal Relevance (MMR) [11], LexRank [12] methods are incorporated to repeatedly select sentences by concurrently considering the topic relevance and redundancy. Hence, they outperform the traditional methods in extractive speech summarization tasks.

Capturing the connotation of a single word [13] is facilitated by generating multidimensional transformation matrices that help in the meaningful representation of tagging part-of-speech. Superior word embeddings can be done by giving additional weight to certain words in a sentence which is an extension of the continuous bag-of-words (CBOW) model [14]. Comparably the continuous skip-gram model improvises an embedding by considering the current word and another word altogether in same sentence. To retain the syntactic and semantic content of the document, continuous skip-gram and continuous bag-of-words models are manipulated.

In comparison with traditional embedding methods namely bag-of-words representation [15] or latent semantic indexing [16], modern techniques like word2vec [17] and GloVe [18] generate prominent representation by reviewing significantly accessible neighbors for a distinct term in a vector space. Regarding linear structures, subtracting two vectors defines the semantic contradiction between two terms. Thus, embedding witnessed benefits in plenty of applications in natural language processing and information retrieval.

Existing methods extract certain features from the source text for processing. As effective extraction of features facilitates efficient processing, discarding specific features leads to some limitations that are detailed in Table 1. Even when these methods generate a conceptual representation to facilitate extractive summarization, a challenge occurs while trying to encode polysemy words. Expectation-Maximization (EM) algorithm [19] is combined with a continuous skip-gram model to handle this problem. To address this problem, neural network models [20] are utilized to calculate the weights of each sentence from the source document. The essence of a neural network is that it is pre-trained with massive data which helps to extract conceptual information. Embedding methodology is improved by the inclusion of multi-task learning and transfer learning which empowers generalized representation to be performed on a variety of datasets. Sequence to sequence models play a vital role in specific applications like machine translation [21], question answering, sentiment analysis, etc.

As a replacement for sequence-oriented computation, recurrent attention methodology works better on language processing tasks. The first transmission model to generate representation functions is purely based on self-attention which does not involve sequence oriented RNN in the Transformer model [22]. Transformer architecture has multiple self-attention tires stacked on both the encoder and decoder portions. Multi-head attention projects have the query, key and its value to obtain

high-dimensional output values. Bidirectional Encoder Representation from Transformers (BERT) [5] is built over Transformer architecture to provide higher performance in NLP applications. This tries to detect the randomly masked 10% to 15% of the words in the training data. Prediction of the next sentence for the input sentence is also verified using BERT. Since the training of such a model consumes more time and two variants of the BERT model have been released by Google, which contains 110 million and 340 million parameters. BERT model is trained with numerous English tokens and performs better on English language. Multiingual BERT (MBERT) is trained on 104 languages and thus has the capacity to process texts in various languages. Those 104 languages in which the model is trained includes both Tamil and English. Due to the high performance of the larger model, it is utilized in the summarization process for embedding.

Once the intermediate representation is generated for the input sentences, sentence ranking and ordering have to be done to generate the final summary. Clustering-based techniques are applied to a group of similar sentences that have common words and those that are semantically related. To determine the informative and similar sentences in a cluster, centroids are required. One of the popular and simple algorithms for clustering is the K-means clustering [23] which partitions the input into k clusters and the objects that have the nearest mean are allotted to its constituent cluster but K-mean clustering lacks in performance when the input size is large and in situations where the number of documents cannot be determined in prior. The hierarchy of clusters is developed using hierarchical-based clustering [24]. This can be executed by either a bottom-up approach termed as agglomerative clustering or a top-down approach termed as divisive clustering technique. In a bottom-up approach, initially, each object is considered as a single cluster and gradually follows by grouping objects to form a single cluster. The top-down approach alternatively takes all objects in a single cluster and keeps on splitting objects while passing down the hierarchy but encounters challenges while reversing the clustering. At times, the number of cluster determinations also becomes difficult. In the situation of clustering large databases which may lead to generating clusters of arbitrary shape, density-based clustering algorithms are employed. Density-Based Spatial Clustering of Applications with Noise [25] a typical density-based clustering algorithm that gathers objects that are densely placed to form clusters and objects that lie scattered in the low dense region is considered as noise. While applying this clustering on applications that hold a large amount of relevant information, objects will be collected more in the initial clusters and the remaining clusters may contain limited objects. This results in inefficient sentence selection from clusters to generate a summary. To generate clusters of almost equal size, clusters that have more objects are segregated into sub-clusters.

Text summarization is performed through various approaches. Extraction of key phrases is done using a graph-based technique [26]. This lags in understanding the semantic similarity between key phrases and their impact during sentence selection. This challenge is resolved by clustering the principle concept of the document and then extracting the key phrases from each cluster. An improvement in dictionary-based and rule-based approaches is accomplished by clustering the major contents in the radiology reports followed by keyword extraction subjected to sentence selection [27]. Semantic framework [29] integrates machine learning and graph-based approach to extract semantically similar sentences for generating a summary.

The proposed hybrid clustering algorithm utilizes a multi-head attention-based embedding algorithm that identifies the keywords and extracts semantic similarity between sentences for generating intermediate representation for Tamil language. The constraint to be considered in hybrid clustering is that the size of all the clusters is less than the specified threshold. This guarantees clusters that are not extremely large in turn ensures effective sentence selection by equally picking up sentences from each cluster. The selected sentences are further checked for redundancy to generate a precise, non-redundant and informative summary.

3. METHODOLOGY

An extractive summarizer aims to generate a summary for the input document which comprises a series of sentences. The summary includes certain sentences which extract only vital information from the input document. The summary generation by the proposed model emphasizes detailed analysis of input documents and identification of important information. An architecture diagram of the proposed model is shown in Fig.1.

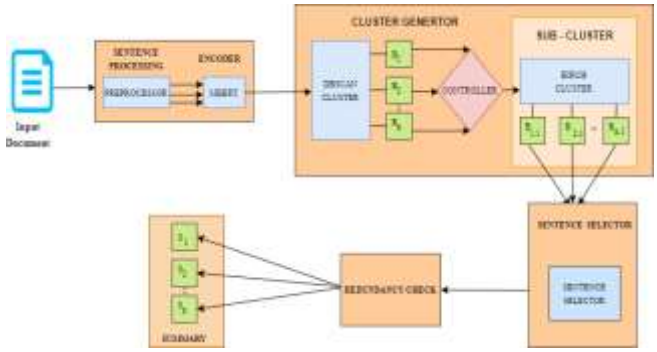

Fig.1. Architecture Diagram of Proposed Model

3.1 TEXT PREPROCESSING:

An important step in Natural Language Processing is pre-processing of text. This transfigures the input data into a more digestive structure to improve the performance of the machine learning algorithm.

Fig.2. Pre-processing steps in a document

Especially in unstructured data like text, it is vital to clean data. This phase is as important as building a model for processing the input data. The input document has repeated information, pictorial representation, tabulated details and some additional information to explain the core content of the document. Various pre-processing methods performed in the model are shown in Fig.2.

The following pre-processing rules are applied to remove certain and non-essential content from the input document.

- Length-based tuning: Too short sentences generally do not contain more information, whereas too long sentences are complex to read and understand, which lacks in conveying information explicitly. Thus, sentences with less than five tokens are considered too short sentences and are removed. Too long sentences having more than 25 tokens may have additional details that are not necessary for a summary. Too long sentences are verified for the presence of cue-words as they provide additional details to the content. The sample cue-words considered in this model are because, in addition to, like, such as, and so on. If such words exist, the tokens following the cue words are ignored.
- Positional Importance: Generally, the first sentence in the document contains the core information about the document. Thus, this sentence is certainly included in the summary.
- Supplementary content removal: Generally, documents include pictorial representations and tabulations to detail the theme of the document. Since these representations do not contribute more to summary generation, they are removed before processing. Contents within brackets are also removed.
- Expanding Contractions: The compressed form of words like couldn't, don't, i/p, etc are expanded into original words like could not, do not, input, etc.
- Sentence Tokenization. The input document is split into individual sentences. This is done to facilitate the process of generating an intermediate representation of sentences that extract various features available in the sentences.

3.2 DEEP REPRESENTATION OF INPUT

Conversion of text data into numerical format is mandated before being fed as input to a model for processing. For instance, word embedding is utilized to represent words for text analysis, in a real-valued vector format that conceals its meaning. The vectors of corresponding words that lie closer in space are considered to have a similar meaning. This can be achieved using language modelling or feature extraction techniques in which words from the vocabulary are mapped to vectors of real numbers. In addition, the semantic and syntactic meaning of the word is identified to generate its representation. Since word embedding fails to handle polysemy and homosemy, which simply means a word having multiple meanings.

Rather than being restricted to just words, sentence embedding would be used to extract more information by directly considering individual sentences. To achieve a higher performance in the representation of input sentences, MBERT [49] architecture is employed. MBERT [49] is a multilingual system that utilizes unlabeled text to train deep bidirectional representations. This is accomplished by using a condition that requires all layers to examine both right and left context at the same time. This model is pre-trained using text from Wikipedia and BookCorpus which is 16GB in size. Sentence representation on a huge corpus, model weights are utilized to infer input sentences and maps to a dense vector representation. This model is subjected to training for several days hence two variants of the BERT model are released by Google. The first model of BERT comprises 110M parameters (12 layers of a Transformer) and is represented as BERT-base model. Another BERT-large model has 340M parameters (24 layers of a Transformer). BERT-Large uncased model is applied as it works with higher performance due to the increase in the number of Transformer layers for producing $N \times E$ embedding vectors in the [CLS] layer of the BERT, where N represents the number of sentences and E represents dimension of embedding [5]. These embeddings may not be the optimum because the other layers produce $N \times W \times E$ embeddings, where W is the tokenized words. This is solved by considering the average or maximum of W to generate $N \times E$ embedding vectors. The embedding vectors for the tokenized sentences help in the effective detection of keywords and semantic association between keywords. These vectors are superior feature inputs that are placed in a high-dimensional plane. Multilingual BERT is applied to generate representations for the input text document.

3.2.1 Input/Output Representations:

A set of input sentences are converted into a numerical representation to perform several downstream tasks. For this purpose, WordPiece embedding [44] which comprises 30,000 token embeddings is applied. The initial token in the input sentence must be a special classification token ([CLS]). Sentence pairs (A,B) are separated using a special token ([SEP]). An embedding is appended with every token to indicate if it belongs to Sentence A or Sentence B. For each input token, intermediate representation is generated by concatenating its corresponding token embeddings, position embeddings and segment embeddings. The Fig.3 describes the procedure for the generation of deep representation of sentences using BERT.

3.2.2 Pre-training BERT:

BERT model is pre-trained with two unsupervised tasks namely Masked Language Model (MLM) and Next Sentence Prediction (NSP). Deep bidirectional models are more efficient than either left-to-right or right-to-left models. During MLM, to train a deep-bidirectional representation, a certain percentage of input tokens are hidden at random using a special token ([MASK]) and then predict those tokens. Hidden representations corresponding to the masked token is given as input to softmax layer for optimizing the prediction. This leads to an imbalance between pre-training and fine-tuning, as the [MASK] token is not present during fine-tuning. Instead of replacing the actual token with [MASK], the training data generator picks the position of the actual token to be hidden for prediction. As most NLP tasks perform based on the relationship between sentences, pre-training for a next sentence prediction is generated. For a sentence Pair (A,B) training is done in such a fashion that has Sentence B as the following sentence for Sentence A (labelled as IsNext) for a certain time and then picks random sentence from the corpus (labelled as NotNext). This MLM and NSP task contributes more to determining the relationship between tokens and sentences during the generation of representation.

During the fine-tuning of BERT model, a self-attention mechanism in Transformer supports diverse downstream tasks. For any task, input and output is fed into the BERT model and the entire parameters are fine-tuned. On the input side, sentence A and sentence B are similar in various aspects depending on the task. On the output side, the final [CLS] token provides the representation for the given input sentences.

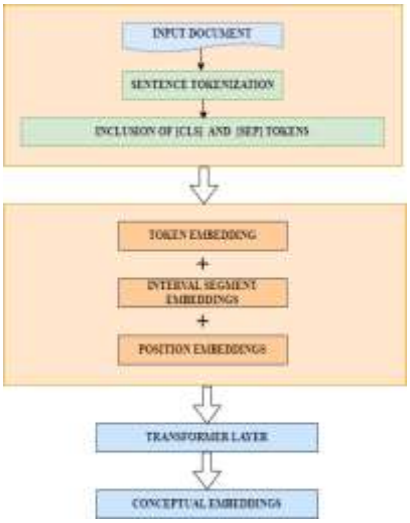

Fig.3. Deep Representation of sentences using BERT

3.3 HYBRID CLUSTERING

A document contains sentences that deliver the core content along with a few sentences which provide the details about the core information. Those sentences which have additional information to the core content can be ignored from the summary. Since this additional information may not have any semantic association with other sentences. The vectors which represent such sentences are scattered away from the other vectors (representation of other sentences). In such cases, noise reduction is necessary, seeming that the document may contain many such sentences with additional content. To facilitate the grouping of large data, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [25] algorithm is applied to retrieve the first level of clustering. DBSCAN is a density-based clustering algorithm that does not need the number of clusters to be specified in prior. The goal is to identify the specific groups in the vector space where there is a mild separation between high point density and its adjacent low point density. This clustering algorithm requires a function to estimate the distance between pair of vectors and certain constraints are applied to analyze its closeness. Among a huge volume of data that comprises noise and outliers, this clustering algorithm can form clusters of diverse shapes and sizes.

Two major parameters of DBSCAN algorithms are eps (ε) and minpts. Eps provides the radius around a given data point. Minpts represent the least count of data points that has to be grouped to signify a region as a dense portion. Dense Reachability and Dense Connectivity are the two concepts that give a better understanding of eps and minpts parameters. A data point is established to be reachable to another point if it rests within a specified distance from the other. Connectivity specifies the border point association. Two data-points '*p*' and '*q*' are density-connected regarding eps and minpts if there exists another data-point 'o' to which both the data-points 'p' and 'q' are density-reachable from 'o' concerning eps and minpts. Core points have a specific number of data-points with a certain distance from the point itself. Border point holds at least one core point at a certain distance. Noise or outliers are neither border points nor core points. These noises are those vectors that do not hold the core content of the input document. The DBSCAN algorithm initiates by randomly picking data-points one after the other. Framing a boundary with a radius of 'eps' if there exist at least 'minpts' data points within the boundary those points are gathered into the same cluster. The neighborhoods calculation is periodically repeated to detect the adjacent data-point and thereby expands the cluster.

Estimation of 'eps' and 'minpts' is essential before initiating the clustering using the DBSCAN algorithm. Wrong estimation of eps and minpts projects all the data points as noise. In proposed hybrid clustering, values for these parameters are estimated by a novel procedure:

- The eps and minpts values are initialized with a minimum value.
- Clustering data points concerning those values is performed.
- While eps and minpts value are incremented until it reaches the maximum limit, clusters are formed.
- Every time the clustering is performed for various eps and minpts values, the number of outliers and the number of clusters formed are stored.
- The parameter value for which there is a minimum number of outliers and a maximum number of clusters is determined and is taken as final eps and minpts value and the clustering is performed with that value.

The clusters are generated using DBSCAN and it is noted that the data points in the clusters are not equally distributed. The first few clusters hold a huge volume of data points and the number of data points in the remaining clusters gradually decreases hence the final cluster would contain a very minimal count of data points compared to that of the initial clusters. This leads to ineffective sentence selection from clusters for generating a summary. As a solution for this challenge, the clusters that contain data points more than a specific threshold are subjected to the next level of clustering.

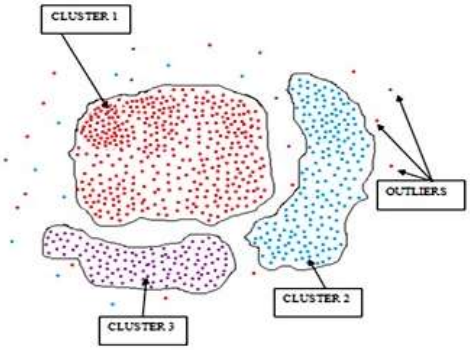

Fig.4. Clusters obtained from the DBSCAN algorithm.

A representation of clustering performed by the DBSCAN clustering algorithm is shown in Fig.4. Three clusters are formed with densely placed data points and the data points that are scattered far from dense data points are renounced from being included in the cluster. Since the clusters formed are larger,

sentence selection becomes challenging. This can be resolved by breaking the larger clusters into sub-clusters which would facilitate effective sentences selection.

The main advantage of performing the second level of clustering is to equally split the sentences under each cluster. The first level of clustering using DBSCAN has eliminated the outliers and gathered adjacent dense portions as clusters. This results in the reduction of the dataset size which minimizes the memory requirement compared to that required to cluster the actual dataset. To intensively reduce memory utilization, the next level of clustering is performed using Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [28]. This algorithm initially condenses the information present in the dense region (formed using DBSCAN) as Clustering Feature (CF) entries. This entry is a very compressed description of the dataset, as each node in it constitutes a sub-cluster rather than an individual data point. A certain number of entries are present in each non-leaf node. Each entry in the leaf node holds a pointer that points to a child node and to the CF which comprises the aggregate of CFs in the child node (subclusters of subclusters). Leaf node comprises a certain number of entries which are CF (subcluster of data points). The overall entries in each leaf node should be smaller than the specified threshold. Intending to connect jointly the entire leaf nodes for coherent scanning, prev and next pointers are associated with each leaf node.

To insert an element in a CF entry, initially, the suitable leaf node is spotted. If the selected leaf node has not attained the maximum threshold, then check if there are elements closer to the element to be inserted and update the CF entry. If no elements are present, then insert this element into the new CF entry. If the maximum threshold is obtained in the selected leaf node, splitting a leaf node is done by fixing the two greater limit elements and distributing the other prevailing elements depending on the distance. On the insertion of an element in the existing CF entry, the information about the path to the leaf for every non-leaf entry is updated. While splitting of a leaf node is performed, the parent node is inserted with a new entry as a non-leaf node, which points to the leaf node that is newly formed. In a condition where the parents cross the threshold size, the parent node is also split and is verified till the root.

Clustering in BIRCH [30] is done by initializing the threshold value, followed by scanning the data and building the initial CF-tree. The threshold is the optimum count of data points that can remain in a CF tree's leaf node. If the lack of memory problem arises even before scanning the entire data, re-initialization of greater value for threshold is done and the scanning is performed again. A new and smaller CF tree is rebuilt by inserting the leaf entries. The eminence of this clustering algorithm is that the grouping of a smaller group of data points is done to form a smaller CF tree.

Thus, initially the clustering is done using the DBSCAN algorithm and the clusters are evaluated. The clusters holding sentences more than a threshold are split as sub-clusters by applying the BIRCH algorithm. If the size of the ruptured sub-cluster exceeds the specified threshold, then the sub-cluster is again split into smaller clusters. This step is iterated until the cluster size does not exceed the threshold. Thus, the clusters generated have a lesser number of sentences than the specified threshold and this guarantees efficient sentences selection from each cluster.

The splitting of larger clusters into sub-clusters is illustrated in Fig.5. Since the splitting of larger clusters into smaller sub-clusters results in evenly distributing the data points like groups, sentence selection from smaller clusters/sub-clusters is effective to generate the summary. Iterative splitting of clusters will continue until the sub-clusters size is less than the threshold for cluster size. Each sub-cluster is considered as an individual cluster during sentence selection.

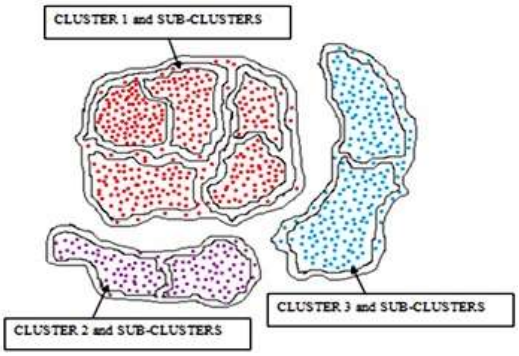

Fig.5. Clusters and sub-clusters are formed by Hybrid clustering algorithm

3.4 SENTENCE SELECTION FROM CLUSTERS:

The final step is the selection of sentences from each cluster to generate the summary. The two main constraints to be considered during this step are (i) the size of the summary generated should be larger than or equal to the required summary size, (ii) an equal number of sentences must be selected from each cluster.

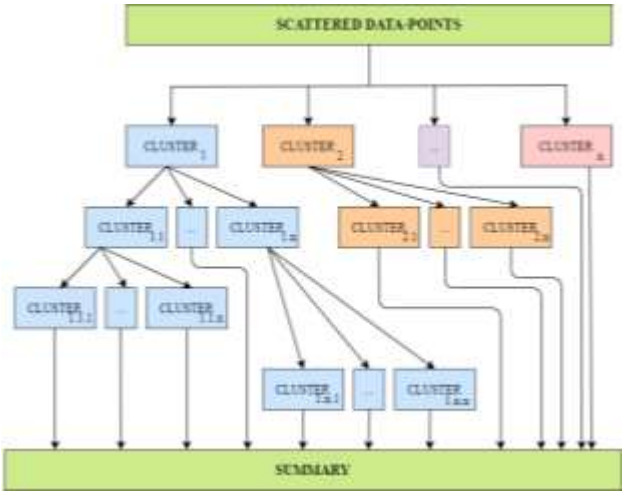

Fig.6. Selection of sentences in clusters/sub-clusters.

Clusters/Sub-clusters with cluster size less than the threshold are formed and the sentence selection is done from each cluster/sub-cluster to generate the summary. Fig.6 illustrates the selection of the sentence from clusters. Each cluster/sub-cluster is considered as an individual group from which sentences are selected. An iteratively equal number of sentences are picked from each cluster/sub-cluster till the size of generated summary matches the size of the required summary. The summary

sentences selected from clusters are arranged in the sequence of the input document and are produced as the summary.

In order to quantify the redundancy of sentences, the similarity between sentences has to be estimated. When a pair of sentences is highly similar, it indicates redundancy. The distance between sentences is measured using Manhattan distance [45]. Each sentence is represented as vectors and if the distance between the pair of vectors is high, then the sentences are semantically dissimilar. If the distance between vectors is low, then the sentences are similar. The distance between each pair of sentences in the summary is calculated. If the distance value is less than a threshold of 0.1, it shows that both the sentences convey almost similar content and thus any one of the sentences is chosen based on the position in the document. This chosen sentence is included in the summary.

The process flow of the Hybrid clustering algorithm is shown in Fig.7. The scattered data points are the vectors generated for each sentence using BERT embedding. These vectors are grouped using the DBSCAN algorithm which collects the densely lying vectors as clusters. The clusters whose size is greater than the desired threshold is further split into sub-clusters. The threshold for the size of the sub-cluster is defined in terms of a fixed percentage of total sentences. If the sub-cluster size exceeds the threshold, the sub-cluster will be iteratively split into internal sub-cluster until its size is less than the threshold. Each cluster/ sub-cluster is recognized as an individual group from which an equal number of sentences are extracted from each group until the summary size exceeds or is equal to the desired summary size.

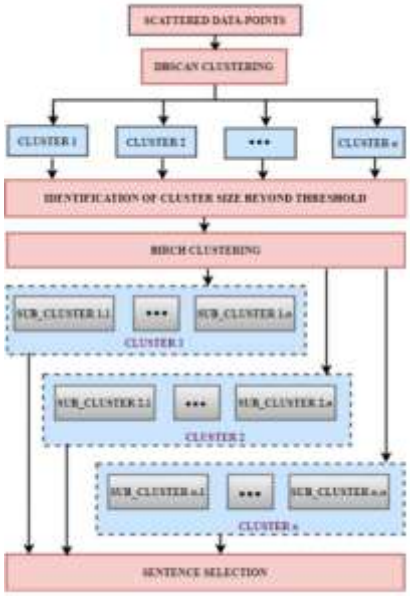

Fig.7. Hybrid Clustering Workflow

HYBRID CLUSTERING ALGORITHM

Input: Text Document (TD)
Output: Extractive Summary (ES)

Begin

/***Text pre-processing** (TD)*/
Length based tuning, positional importance, ignore content within bracket, ignore additional information,
expand contractions and sentence tokenization.
return {S_1,S_2,.....,S_n};
/* Where, S_n =Clean_Sentences; n= number of sentences in the document*/
/*Embedding using MBERT{S_1,S_2,.....,S_n}*/
BERT: {24layers of Transformers,
maps each sentence to a vector representation}.
return $(N \times E)$, {V_1,V_2,.....,V_n};
/* Where, E= embedding dimension; vector= V_n=Data-points*/
/***Estimation of eps and Minpts value*/**
/*DBSCAN (Data-points, eps, MinPts)*/
/*Where, eps= radius around the Data-points; Minpts= minimum number of Data-points within "eps" */
get optimal (eps & Minpts)
if (ClusterID= empty)
/* Where, ClusterID= Initial_cluster*/
for (eps=x; eps≤y; eps++)
/* Where, x =minimum limit and y= maximum limit*/
for (MinPts =m; MinPts ≤n; MinPts ++)
/* Where, m = minimum limit and n= maximum limit*/
/***DilateCluster** (Data-points, eps, MinPts)*/
Store the no of clusters generated and noise for all eps and Minpts value.
end for
end for
end if
if (no_of_Clusters=max and no_of_outliers=min)
Choose the specific eps and Minpts for clustering.
end if
/*Clustering using DBSCAN (Data-points, eps, MinPts)*/
/*Where Clusters= Clusters formed using DBSCAN algorithm, C=Clusters*/
/***Cluster_size_verification (**Clusters)*/
for (C=C_1; C≤C_n; C++)
if (size (Clusters) >threshold))
/*Where, threshold= user defined value*/
perform sub-clustering using BIRCH.
end if
/*BIRCH (Branching_factor, n_clusters, threshold_CF)
/*Where Branching_factor=maximum number of CF tree in each node,
n_clusters= number of final clusters,
threshold_CF= radius of the finally formed CF trees and the closest CF tree,
CF= Cluster Feature Tree= Tree to store all the Data-points*/
form CF_tree
insert data-point within Clusters into t_1
if (size (CF_tree)> threshold_CF))
threshold_CF++;
rebuild CF_tree t_2 from t_1
repeat until all the data points are included in the CF_tree
end if
Group CF_tree within desired range as clusters.
return Sub-clusters.
/***Sub-cluster_size_verification** (Sub-Clusters)*/
if (size(sub-clusters)>threshold))
perform Sub-clustering using BIRCH
return Sub-clusters.

```
        end if
      end for
      /*Select summary sentences from clusters*/
        for each (Cluster)
if (Cluster has Sub-clusters)
   extract one sentence from each Sub-cluster.
else
   extract one sentence from cluster.
            collect all the sentences as summary.
calculate (size (summary))
         if (size (summary)< size(expected_summary))
            /*Where summary= collection of sentences from each
cluster,
            expected_summary= user_expected_summary
  repeat sentence extraction from clusters.
else
  return summary.
end if
       end for
```

4. EXPERIMENTAL SETUP

The proposed hybrid clustering model is implemented in python, utilizing packages such as spacy, pandas, NumPy, scipy, sklearn, and Transformers. The performance of this model is evaluated by applying it on a benchmark dataset. The dataset on which the model is applied, and the summary output is described below.

4.1 DATASET

The proposed hybrid clustering model is evaluated on CNN Daily/Mail dataset [34], which includes a long paragraph summary dataset This dataset comprises of English news articles produced by reporters at CNN and Dailymail. On average, approximately 781 tokens, 40 sentences are available in each article. This dataset includes 2,87,226 training pairs of news articles and its relevant summary, 13,368 verification pairs of article and summary, 11,490 testing pairs of articles and summary. This benchmark dataset is exclusively designed for machine-reading comprehension and question answering. The recent version of the dataset [39] supports extractive and abstractive document summarization. This non-anonymized dataset comprises textual news articles and the representative summary with the highlights. The articles from the test samples in this dataset are fed as input to the hybrid clustering algorithm and summaries are generated using the proposed system, which is then compared with the translated reference summary to evaluate the quality of summary.

From the testing pair of the CNN/DailyMail dataset, 20 input document are translated into Tamil and is used for the evaluation of the proposed model on Tamil document summarization. Also, the Tamil input document along with its system generated summary is reviewed by several reviewers based on the coverage, coherence, redundancy and the quality of the summary.

4.2 HYBRID CLUSTERING

The proposed system combines two clustering algorithms namely Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) to generate clusters of equal sizes to facilitate effective sentence selection. Initially, DBSCAN groups the data that are densely coupled and ignores the noise. During observation, it is noticed that few clusters are extremely large, as a single document comprises of sentences conveying information on a specific domain. As deciding on the number of sentences to be retrieved from each distinct-sized cluster is challenging, the clusters are tried to be fragmented into identical sizes. BIRCH is applied to split the large-sized clusters created by DBSCAN. The hyperparameters of the proposed system comprise of the parameters of DBSCAN and BIRCH algorithms. These hyperparameters are tuned in such a fashion to generate equal-sized clusters and reduce the number of sentences being considered as noise. The Table.2 lists the parameters of hybrid clustering algorithms.

Table.2. Parameters of Hybrid clustering algorithm

Parameters	**Default values**	**Fine-tuned values**
Density-Based Spatial Clustering of Applications with Noise (DBSCAN)		
Eps	0.5	Increments from 2 till the condition is satisfied
min_samples	5	from 2 till 20 for every iteration
Metric	Euclidean	Euclidean
metric_params	None	None
Algorithm: {'auto', 'ball tree', 'kd_tree', 'brute'}	Auto	Auto
leaf_size	30	Not application for 'auto' algorithm
P	None	Equivalent to Euclidean distance
n_jobs	None	Based on the necessity for parallel processing
Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)		
Threshold	0.5	0.5
branching factor	50	Based on the required summary size
n_clusters	3	3
compute_labels	True	False
Copy	True	True

The 'eps' and 'min_samples' parameter values of DBSCAN are estimated based on a constraint of having maximum number of clusters to be created with a smaller number of noise data. The parameter value of 'branching_factor' in BIRCH is fixed based on required summary size. It is utilized to generate sub-clusters from the large-sized clusters (holding too much data) to facilitate effective sentence selection.

4.3 EMBEDDING EVALUATION

Embedding transforms words/tokens into their numerical representations. This transformation facilitates the processing of textual data. Word embeddings map each word in the input data to relevant real-valued numerical vectors. Each word is tokenized and is represented in a vector space. The goal of word embedding is to grasp the semantic meaning of each input token. On the other hand, representation of the entire sentence and its contextual information is performed using Sentence Embedding. Similar representations are assigned to sentences having similar meanings.

- ***Word2Vec:*** This is an efficient and simple word embedding technique utilized for addressing advanced NLP problems. It is repeatedly applied on a huge text corpus to understand associations between words. This technique uses cosine similarity metric to measure the relevancy between words. The cosine angle is 1, which means that the words are overlapping and the angle being 90 shows that the words are semantically dissimilar. Similar representations are assigned to contextually similar tokens. Word2Vec is prominent in apprehending contextual association between words.
- ***GloVe (Global Vectors for Word Representation):*** This is an extension of Word2Vec which gathers global semantic information in text corpus by estimating the global word-word co-occurrence matrix. This encapsulates the benefits of both the matrix factorization method and the local context window method. This holds a simpler error function that minimizes the computational cost of model training. This method is efficient in apprehending semantic information.
- ***FastText:*** This technique is an extension of Word2Vec embedding. Rather than just taking individual words into a network for processing, FastText segments each word into several sub-words. The embedding for the input token will be the concatenation of these sub-words. Once the training is done, the word embeddings for the entire n-grams present in the training dataset are retrieved. As there is a high possibility of recurrence of n-grams in other words, this method effectively handles rare words.
- ***InferSent:*** This is a supervised sentence embedding technique, which is trained on Natural Language Inference (NLI) data. A pair of sentences is taken as input and are encoded to generate actual sentence embeddings. The association between these embeddings is derived using concatenation, element-wise product and absolute element-wise difference. The resultant vector is in turn given as input to a classifier for classification into one among the three categories. GloVe vectors are incorporated as pre-trained word embeddings in InferSent.

4.4 SUMMARY EVALUATION

To evaluate the quality of summary generated by hybrid clustering model, the number of terms that overlap, namely skip-grams, longest common sub-sequences among the generated summary and the reference summary is determined. Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [31] is the metric used to evaluate the quality of the proposed system. ROUGE-1 metric for unigram, ROUGE-2 metric for bigram and ROUGE-L metric for Longest common subsequence are the three variants of ROUGE metrics used to state the quality of the system. With the unigram, bigram and longest common sequence comparison among the generated and the reference summary, the precision, recall and f-score are determined. Precision defines how far the sentences selected by the system to be included in generated summary are available in the reference summary. This illustrates the extent to which the generated summary is relevant to the reference summary. Recall defines how far the sentences available in reference summary are being selected by the system to be included in the generated summary. F-score is the balanced means of precision and recall. F-Score of ROUGE-1, ROUGE-2 and ROUGE-L is employed for evaluation. DBSCAN, BIRCH and the proposed hybrid clustering algorithm are executed with the testing pair in CNN/Dailymail dataset and the summary is generated in various sizes. The metrics that are considered for evaluating the summary manually by various reviewers are.

- Coherence: Contextual Connection between adjacent sentences to facilitate readability.
- Coverage: All the information/content in input are included in the summary.
- Redundancy: Repetition of sentences from the input passage into the summary.
- Summary Quality: The overall quality of the Summary in the scale of 1 to 5.

4.5 EXPERIMENTAL RES ULTS

- ***Analysis Of English Summary Evaluation:*** The generated summary for the English document is compared with its respective reference summary and the performance is evaluated using the ROUGE f-score metrics and the values determined during evaluation are produced in Table.3.

Table.3. Performance metrics evaluation of Hybrid Clustering model on English document.

Required Summary size	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
DBSCAN Algorithm			
30%	0.238	0.098	0.238
50%	0.247	0.100	0.245
60%	0.273	0.113	0.254
75%	0.303	0.123	0.271
BIRCH Algorithm			
30%	0.275	0.117	0.256
50%	0.323	0.139	0.297
60%	0.330	0.141	0.302
75%	0.341	0.149	0.313
Proposed Hybrid Clustering Algorithm			
30%	0.402	0.207	0.343
50%	**0.429**	**0.213**	**0.371**
60%	0.501	0.217	0.376
75%	0.535	0.240	0.379

As the expected summary size increases, most of the informative sentences are included in the summary. Generally, a summary is expected to be half the size of the input document, the respective evaluation score of DBSCAN, BIRCH and proposed hybrid clustering algorithm are compared and it is evident that the proposed algorithm outperformed the other two clustering techniques.

- ***Analysis Of Tamil Summary Evaluation:*** The generated summary for the document translated into Tamil is compared with its respective reference summary and the performance is evaluated using the ROUGE f-score metrics and the values determined during evaluation are produced in Table.4.

Table.4. Performance metrics evaluation of Hybrid Clustering model on Tamil document.

Required Summary Size	ROUGE-1	ROUGE-2	ROUGE-L
DBSCAN Algorithm			
30%	0.207	0.089	0.216
50%	0.236	0.098	0.234
60%	0.268	0.109	0.245
75%	0.294	0.119	0.259
BIRCH Algorithm			
30%	0.258	0.106	0.244
50%	0.319	0.127	0.286
60%	0.325	0.123	0.289
75%	0.333	0.147	0.301
Proposed Hybrid Clustering Algorithm			
30%	0.398	0.201	0.327
50%	**0.417**	**0.209**	**0.366**
60%	0.498	0.209	0.369
75%	0.512	0.224	0.370

From the above Table.it is evident that, as the size of the generated summary increases, the sentences included in the summary also increases. Basically, a summary is expected to be half the size of the input document, the respective evaluation score of DBSCAN, BIRCH and proposed hybrid clustering algorithm are compared and it is evident that the proposed algorithm outperformed the other two clustering techniques.

- **Analysis Of Embedding Evaluation on English document:** The proposed hybrid clustering algorithm is implemented with various word embeddings and sentence embeddings on English document and the results are recorded in Table.5. The comparative study of various embedding techniques on Tamil document as the embedding techniques represent the English text into vectors.

Table.5. Performance evaluation of various embedding techniques with Hybrid Clustering model on English document.

Various Embedding Techniques + Proposed Hybrid Clustering Algorithm	ROUGE-1	ROUGE-2	ROUGE-L
Word2Vec [35]	0.371	0.186	0.201
GloVe [46]	0.383	0.193	0.217
FastText [47]	0.398	0.199	0.219
InferSent [48]	0.407	0.206	0.352
MBERT [49]	**0.429**	**0.213**	**0.371**

The proposed hybrid clustering algorithm performed after generating intermediate representations using MBERT is compared with various embedding techniques. From the experiment it is evident that the representations generated using MBERT are efficient when compared to other embedding techniques. With the results obtained, the summary generated achieved a high ROUGE score.

This comparative study involves various embedding algorithms performed with TextRank and PACSUM techniques. TextRank [40] represents each sentence from the source document as nodes present in an undirected graph. The similarity between each sentence is given as the weight of the edges connecting the nodes. PageRank [41] technique is used to determine the centrality of a node which helps in deciding whether to include the sentence in the summary or not. Advancement in the TextRank algorithm is made by incorporating BERT to calculate sentence similarity and to build a directed graph in which the edges are decided by the relative sentence position.

Position-Augmented Centrality-based Summarization (PACSUM) [42] is a type of sentence selection technique that works based on sentence position. The various embedding techniques are incorporated with PACSUM [43] and the quality of the summary is evaluated. LEAD-3 [32] baseline model extracts three sentences from the source document to generate the summary. From literature analysis, it is noted that these systems are applied on CNN/DailyMail dataset and its performance is evaluated using ROUGE metrics is tabulated in Table.5. While comparing the evaluation score of the proposed system with an existing system, it is evident that the proposed system outperforms in generating an effective summary.

- **Analysis Of Embedding Evaluation on Tamil document:** The proposed hybrid clustering algorithm is implemented with various word embeddings and sentence embeddings on Tamil document and the results are recorded in Table.6. The comparative study of various embedding techniques on Tamil document as the embedding techniques represent the English text into vectors.

Table.6. Performance evaluation of various embedding techniques with Hybrid Clustering model on Tamil document.

Various Embedding Techniques + Proposed Hybrid Clustering Algorithm	ROUGE-1	ROUGE-2	ROUGE-L
FastText [47]	0.287	0.176	0.198
InferSent [48]	0.198	0.194	0.352
MBERT [49]	**0.417**	**0.209**	**0.366**

By comparing the combination of MBERT and proposed hybrid clustering algorithm with various embedding techniques, it is evident that the proposed method is efficient when compared to other embedding techniques. It Is also evident that the proposed model suits well for Tamil language also. With the results obtained, the summary generated achieved a high ROUGE score.

Table.5. Performance metrics for traditional unsupervised summarization methods

Traditional Unsupervised Summarization methods	ROUGE-1	ROUGE-2	ROUGE-L
LEAD-3 [32]	0.405	0.177	0.367
TF-IDF + TextRank [41]	0.332	0.118	0.296
Skip-thought + TF-IDF + TextRank [41]	0.314	0.102	0.282
BERT + TextRank	0.308	0.096	0.274
TF-IDF + PACSUM [42]	0.392	0.163	0.353
Skip-thought + PACSUM [42]	0.386	0.161	0.349
BERT + PACSUM [43]	0.407	0.178	0.369
BERT + DBSCAN + BIRCH	**0.429**	**0.213**	**0.371**

While analyzing the metrics, it is clear that the quality of summary generated by the proposed hybrid methods outperforms the existing unsupervised methods.

Comparison of the proposed model with the results of a few recent state-of-the-art models for Neural Network-based Automatic summarization is performed. NN-SE [33] is an effective Neural Network model which includes a hierarchical document encoder and decoder with an attention layer for extracting summary sentences. SummaRuNNer [34] generates a summary by using the

Recurrent Neural Network (RNN) model. HSSAS [35] incorporates hierarchically structured self-attention to generate embeddings for sentences and documents. BANDITSUM [36] handles summarization as a Contextual Band (CB) problem, which detects the sentences from the document to be included in the summary to increase the ROUGE score. These methods are trained and evaluated using CNN Dailymail dataset and the results of evaluation are illustrated in Table.6.

Table.6. Performance metrics for Neural Network based summarization methods

Neural Network based Summarization methods	ROUGE-1	ROUGE-2	ROUGE-L
NN-SE [37]	0.35	0.13	0.32
SummaRuNNer [38]	0.39	0.16	0.35
HSSAS [39]	0.42	0.17	0.37
BANDITSUM [40]	0.41	0.18	0.37
BERT + DBSCAN + BIRCH	**0.429**	**0.213**	**0.371**

From the above comparisons, it is evident that the proposed hybrid clustering model is efficient as it gives a better result in terms of ROUGE. The clustering efficiency of the proposed model is evaluated and is proven to be efficient to generate clusters that tightly couple similar points and repel the dissimilar points. As the cluster formation is done effectively, the extraction of sentences from clusters is effective and the summary generated is worthy.

4.6 READER'S OPINION ON GENERATED TAMIL SUMMARY

The summary generated by the proposed clustering algorithm for 20 translated news documents is shared with several reviewers and their opinions are collected. The metrics used by the reviewers to evaluate the summary is listed below.

- ***Coherance***: Contextual Connection between adjacent sentences to facilitate readability.
- ***Coverage***: All the information/content in input are included in the summary.
- ***Redundancy***: Repetition of sentences from the input passage into the summary.
- ***Quality of the Summary***: The overall quality of the Summary is on the scale of 1 to 5.

From the scores given by the reviewers, it is inferred that the summary generated is coherent and includes all the information present in the input document. It is also mentioned that the information is not redundant. The quality of the summary generated for all 20 documents is rated on the scale of 1 to 5 and is mentioned in Table.7. The consolidated comments and average quality score given by each reviewer are mentioned in Table.7.

Table.7. Consolidated Score provided by the reviewers.

Reviewers	Coherence	Coverage	Redundancy	Average Summary quality
Reviewer 1	Yes	Yes	No	4.736842105
Reviewer 2	Yes	Yes	No	4.684210526
Reviewer 3	Yes	Yes	No	4.684210526
Reviewer 4	Yes	Yes	No	4.842105263
Reviewer 5	Yes	Yes	No	4.526315789
Reviewer 6	Yes	Yes	No	4.526315789
Reviewer 7	Yes	Yes	No	4.684210526
Reviewer 8	Yes	Yes	No	4.578947368
Reviewer 9	Yes	Yes	No	4.578947368
Reviewer 10	Yes	Yes	No	4.631578947

From the above evaluation, it is evident that the summary generated by the proposed clustering algorithm is coherent and it covers all the significant information present in the input

document. From the ROUGE score calculated for the English and the Tamil summary, it proves that the summary holds the important terms present in the input document. From the reader's review it is observed that the summary is readable and conveys all the information present in the input document. This also proves that the summary does not hold any redundant information. From the quality score provided by the user it shows that the summary is readable and conveys the information in a precise manner to the user.

5. CONCLUSION AND FUTURE WORK

In this paper, a novel hybrid clustering model is proposed to perform unsupervised extractive text summarization model for both English and Tamil language. The input document is pre-processed to remove the irrelevant content and to extract only sentences required for the summarization process. The sentences are represented as feature vectors by considering their semantic similarity by using the Transformer model that is being trained in multiple languages. These vectors are subjected to clustering to group/split the similar and dissimilar vectors. The hybrid clustering algorithm eliminates the noises and frames the maximum number of clusters to group densely placed vectors. The size of the clusters is split as sub-clusters within a larger cluster until the size of the cluster remains less than the threshold. An equal number of sentences are extracted from each cluster/sub-cluster until the size of the summary is equal to the expected summary size. The efficiency of the proposed clustering algorithm and the generated summary are evaluated and compared with various state-of-art unsupervised models and Neural Network-based models. The summary generated is also evaluated based on the reader's opinion and certain metrics.

Our future directions include exploring efficient embedding technology that can extract semantic information along with relevancy in similar sentences and grouping them by applying novel techniques, to perceive the succeeding closer vectors and include them in the summary.

REFERENCES

[1] F. Kyoomarsi, H. Khosravi, E. Eslami, P.K. Dehkordy and A. Tajoddin, "Optimizing Text Summarization based on Fuzzy Logic", *Proceedings of International Conference on Computer and Information Science*, pp. 347-352, 2008.

[2] V. Gupta and G.S. Lehal, "A Survey of Text Summarization Extractive Techniques", *Journal of Emerging Technologies in Web Intelligence*, Vol. 2, No. 3, pp. 258-268, 2010.

[3] C.S. Saranyamol and L. Sindhu, "A Survey on Automatic Text Summarization", *International Journal of Computer Science and Information Technologies*, Vol. 5, No. 6, pp. 7889-7893, 2014.

[4] V. Gupta and G.S. Lehal, "A Survey of Common Stemming Techniques and Existing Stemmers for Indian Languages", *Journal of Emerging Technologies in Web Intelligence*, Vol. 5, No. 2, pp. 157-161, 2013.

[5] J. Devlin, M.W. Chang, K. Lee and K. Toutanova, "Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding, *Computer Science*, pp. 1-6, 2018.

[6] V. Gupta and G.S. Lehal, "A Survey of Text Mining Techniques and Applications", *Journal of Emerging Technologies in Web Intelligence*, Vol. 1, No. 1, pp. 60-76, 2009.

[7] N. Bhatia and A. Jaiswal, "Trends in Extractive and Abstractive Techniques in Text Summarization", *International Journal of Computer Applications*, Vol. 117, No. 6, 2015.

[8] T. Vodolazova, E. Lloret, R. Muñoz and M. Palomar, "The Role of Statistical and Semantic Features in Single-Document Extractive Summarization", *Artificial Intelligence Research*, Vol. 2, No. 3 pp. 1-10, 2013.

[9] Y. Gong and X. Liu, "Generic Text Summarization using Relevance Measure and Latent Semantic Analysis", *Proceedings of International Conference on Research and Development in Information Retrieval*, pp. 19-25, 2001.

[10] X. Wan and J. Yang, "Multi-Document Summarization using Cluster-based Link Analysis", *Proceedings of International Conference on Research and Development in Information Retrieval*, pp. 299-306, 2008.

[11] J. Carbonell and J. Goldstein, "The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries", *Proceedings of International Conference on Research and Development in Information Retrieval*, pp. 335-336, 1998.

[12] G. Erkan and D.R. Radev, "Lexrank: Graph-based Lexical Centrality as Salience in Text Summarization", *Journal of Artificial Intelligence Research*, Vol. 22, pp. 457-479, 2004.

[13] W. Ling, C. Dyer, A.W. Black and I. Trancoso, "Two/Too Simple Adaptations of Word2vec for Syntax Problems", Proceedings of International Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1299-1304, 2015.

[14] W. Ling, Y. Tsvetkov, S. Amir, R. Fermandez, C. Dyer, A.W. Black and C.C. Lin, "Not All Contexts are Created Equal: Better Word Representations with Variable Attention", *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, pp. 1367-1372, 2015.

[15] C.D. Manning and P. Raghavan, "Introduction to Information Retrieval", *Cambridge University Press*, 2008.

[16] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer and R. Harshman, "Indexing by Latent Semantic Analysis", *Journal of the American Society for Information Science*, Vol. 41, No. 6, pp. 391-407, 1990.

[17] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality", *Advances in Neural Information Processing Systems*, pp. 3111-3119, 2013.

[18] J. Pennington, R. Socher and C.D. Manning, "Glove: Global Vectors for Word Representation", *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, pp. 1532-1543, 2014.

[19] F. Tian, H. Dai, J. Bian, B. Gao, R. Zhang, E. Chen and T.Y. Liu, "A Probabilistic Model for Learning Multi-Prototype Word Embeddings", *Proceedings of International Conference on Computational Linguistics: Technical Papers*, pp. 151-160, 2014.

[20] D. Bahdanau, K. Cho and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate", *Computer Science*, pp. 1-6, 2014.

[21] I. Sutskever, O. Vinyals and Q.V. Le, "Sequence to Sequence Learning with Neural Networks", *Advances in Neural Information Processing Systems*, pp. 3104-3112, 2014.

[22] N. Kalchbrenner, L. Espeholt, K. Simonyan, A.V.D. Oord, A. Graves and K. Kavukcuoglu, "Neural Machine Translation in Linear Time, *Machine Learning*, pp. 1-7, 2016.

[23] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations", *Proceedings of Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, No. 14, pp. 281-297, 1967.

[24] Y. Zhao, G. Karypis and U. Fayyad, "Hierarchical Clustering Algorithms for Document Datasets", *Data Mining and Knowledge Discovery*, Vol. 10, No. 2, pp. 141-168, 2005.

[25] M. Ester, H.P. Kriegel, J. Sander and X. Xu, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *kdd*, Vol. 96, No. 34, pp. 226-231, 1996.

[26] A. Gupta, L. Fabian and J.A. Brill, "Phosphatidylinositol 4, 5-Bisphosphate Regulates Cilium Transition Zone Maturation in Drosophila Melanogaster", *Journal of Cell Science*, Vol. 131, No. 16, 2018.

[27] M. Moradi and N. Ghadiri, "Different Approaches for Identifying Important Concepts in Probabilistic Biomedical Text Summarization", *Artificial Intelligence in Medicine*, Vol. 84, pp. 101-116, 2018.

[28] T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases", *Sigmod Record*, Vol. 25, No. 2, pp. 103-114, 1996.

[29] C.Y. Lin, "ROUGE: Recall-Oriented Understudy for Gisting Evaluation", *Proceedings of the Workshop on Text Summarization*, pp. 1-7, 2003.

[30] K. Woodsend and M. Lapata, "Automatic Generation of Story Highlights", *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pp. 565-574, 2010.

[31] J. Cheng and M. Lapata, "Neural Summarization by Extracting Sentences and Words", *Computation and Language*, pp. 1-6, 2016.

[32] R. Nallapati, F. Zhai and B. Zhou, "Summarunner: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents", *Proceedings of International Conference on Artificial Intelligence*, pp. 1-6, 2017.

[33] K. Al-Sabahi, Z. Zuping and M. Nadher, "A Hierarchical Structured Self-Attentive Model for Extractive Document Summarization", *IEEE Access*, Vol. 6, pp. 24205-24212, 2018.

[34] Y. Dong, Y. Shen, E. Crawford, H. van Hoof and J.C.K. Cheung, "Banditsum: Extractive Summarization as a Contextual Bandit", *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, pp. 3739-3748, 2018.

[35] A. See, P.J. Liu and C.D. Manning, "Get to the Point: Summarization with Pointer-Generator Networks", *Computation and Language*, pp. 1-6, 2017.

[36] Y. Liu and M. Lapata, "Text Summarization with Pretrained Encoders", *Proceedings of International Conference on Machine Learning*, pp. 1-6, 2019.

[37] K.M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman and P. Blunsom, "Teaching Machines to Read and Comprehend", *Advances in Neural Information Processing Systems*, Vol. 28, pp. 1693-1701, 2015.

[38] R. Mihalcea and P. Tarau, "Textrank: Bringing Order into Text", Proceedings of International Conference on Empirical Methods in Natural Language Processing, pp. 404-411, 2004.

[39] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", *Computer Networks and ISDN Systems*, Vol. 30, pp. 107-117, 1998.

[40] H. Zheng and M. Lapata, "Sentence Centrality Revisited for Unsupervised Summarization", *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pp. 6236-6247, 2019.

[41] B. Desgraupes, "Clustering Indices", Available at https://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf, Accessed in 2013.

[42] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey and J. Dean, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation", *Proceedings of International Conference on Artificial Intelligence*, pp. 1-7, 2016.

[43] A. Huang, "Similarity Measures for Text Document Clustering", *Proceedings of International Conference on Computer Science Research*, Vol. 4, pp. 9-56, 2008.

[44] D.C. Beddows, M. Dall'Osto and R.M. Harrison, "Cluster Analysis of Rural, Urban and Curbside Atmospheric Particle Size Data", *Environmental Science and Technology*, Vol. 43, No. 13, pp. 4694-4700, 2009.

[45] Y. Goldberg and O. Levy, "Word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method", *Proceedings of International Conference on Machine Learning*, pp. 1-6, 2014.

[46] J. Pennington, R. Socher and C.D. Manning, "Glove: Global Vectors for Word Representation", *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, pp. 1532-1543, 2014.

[47] B. Athiwaratkun, A.G. Wilson and A. Anandkumar, "Probabilistic Fasttext for Multi-Sense Word Embeddings", *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pp. 1-11, 2018.

[48] H.A.M. Hassan, G. Sansonetti, F. Gasparetti, A. Micarelli and J. Beel, "Bert, Elmo, Use and Infersent Sentence Encoders: The Panacea for Research-Paper Recommendation?", *Proceedings of International Conference on Machine Learning*, pp. 6-10, 2019.

[49] T. Pires, E. Schlinger and D. Garrette, "How Multilingual is Multilingual BERT?", *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pp. 4996-5001, 2019.