Article

A Framework for Generating Extractive Summary from Multiple Malayalam Documents

K. Manju [1,*], S. David Peter [2] and Sumam Mary Idicula [1]

[1] Department of Computer Science, Cochin University of Science and Technology (CUSAT), Kochi 682022, India; sumam@cusat.ac.in
[2] School of Engineering, Cochin University of Science and Technology (CUSAT), Kochi 682022, India; davidpeter@cusat.ac.in
* Correspondence: manju@mec.ac.in; Tel.: +91-9447-380-826

Abstract: Automatic extractive text summarization retrieves a subset of data that represents most notable sentences in the entire document. In the era of digital explosion, which is mostly unstructured textual data, there is a demand for users to understand the huge amount of text in a short time; this demands the need for an automatic text summarizer. From summaries, the users get the idea of the entire content of the document and can decide whether to read the entire document or not. This work mainly focuses on generating a summary from multiple news documents. In this case, the summary helps to reduce the redundant news from the different newspapers. A multi-document summary is more challenging than a single-document summary since it has to solve the problem of overlapping information among sentences from different documents. Extractive text summarization yields the sensitive part of the document by neglecting the irrelevant and redundant sentences. In this paper, we propose a framework for extracting a summary from multiple documents in the Malayalam Language. Also, since the multi-document summarization data set is sparse, methods based on deep learning are difficult to apply. The proposed work discusses the performance of existing standard algorithms in multi-document summarization of the Malayalam Language. We propose a sentence extraction algorithm that selects the top ranked sentences with maximum diversity. The system is found to perform well in terms of precision, recall, and F-measure on multiple input documents.

Keywords: Malayalam language; extractive mutidocument summarization; NLP; sentence encoding; TextRank; maximum marginal relevance

Citation: Manju, K.; David Peter, S.; Mary Idicula, S. A Framework for Generating Extractive Summary from Multiple Malayalam Documents. *Information* **2021**, *12*, 41. https://doi.org/10.3390/info12010041

Received: 24 December 2020
Accepted: 15 January 2021
Published: 18 January 2021

1. Introduction

Nowadays, the amount of data on the web is growing exponentially on any topic. The volume of data circulating in the digital space, generally the unstructured textual data, demands building automated text summarization tools to get insights from them quickly. Document summaries provide users the briefing of the most notable information contained in the document. Automatic document summarization is one of the most challenging and exciting issues in Natural Language Processing (NLP).

The automatic text summarization system has attracted substantial interest in providing relevant information in less time [1]. Text summarization is a process used to generate a simplified version of the original document. There are several types of summarization methods. Considering the whole or particular part of a text, summarization is categorized into generic and query relevant summarization. A generic summary presents an overall sense of the document's content, while a query-focused summary shows the document's content related to the user query [2,3].

Based on the number of source documents, there is single document summarization (SDS) and multi-document summarization (MDS). A single document produces a summary that is obtained from one source document where the content is sourced around a single topic [4]. At the same time, the multi-document summarization is taken from various sources or documents that discuss the same topic [5–7]. However, the task of summarizing multiple documents is more complicated than the job of summarizing a single document. The challenges associated with summarizing multiple documents are redundancy and cohesion. Initially, most of the research was conducted on single-document summarization. Multi-document summarization studies started by extending the single document techniques to more than one textual documents.

Based on the approach followed for generating the summary, text summarization can be extractive or abstractive. An extractive summary is generated by concatenating meaningful sentences extracted from the document to be summarized. On the other hand, an abstractive summary conveys the primary information from the documents [8]. Abstractive summarization requires extensive natural language processing. Therefore, it is more complicated than extractive summarization. Extractive summarization, because of its higher achievability, has attained a standard in summarizing documents [9].

This work focuses on extraction based multi-document summarization. Malayalam is taken as the language of study. Malayalam is an important language in India, which is the regional language of Kerala spoken by 37 million people around the world. Research on generating Malayalam text summaries is still in its infancy compared to the research accomplished in English or other languages. This is due to the issues and challenges related to the language's complexity and the lack of standard automatic Malayalam NLP tools. Even though certain works related to single document summarization have been carried out, there is no existing multi-document text summarization system for Malayalam [10–15].

Most multi-document summarization research follows extractive approaches by selecting sentences that best describe the source documents' main idea and combine them to generate the summary. Mainly the sentence selection process can be classified into three groups; statistical-based approaches, topic modeling, and graph-based approaches [9]. The statistical-based approaches deal with statistical features such as sentence length, position, and keywords to extract significant sentences from the source text [16]. Topic modeling helps to find hidden semantic structures inside documents to select summary sentences [17]. The graph-based methods construct a graph with sentences as the nodes and the relationship among sentences as the edges. Sentences are scored using the Page Rank algorithm [18]. Recent document summarization research focuses on deep learning methods as these provide better results than the traditional one. This study requires an extensive labeled data set for training. However, such a dataset for multi-document summarization is not available. Hence most research in multi-document summarization is still based on techniques that select salient sentences from source documents.

This study presents a generic extractive multi-document summarization model to extract a summary from multiple Malayalam documents. Despite the encouraging output of deep learning approaches, these deep learning models do not work for Malayalam language due to the lack of labeled data sets required for massive training. Since many previous studies have shown that graph-based methods perform well, we used the TextRank algorithm to rank and select sentences for the summary [18]. TextRank had shown promising results in most of the summarization tasks and is language independent [9]. As nouns directly increase the importance of a sentence, we have used the noun count to modify the page rank algorithm used with TextRank [19]. A multi-document summarization problem is the information overlap among salient sentences, which leads to redundancy in summary [20]. To deal with this, we combine MMR with TextRank. Experimental results show that the system is significantly efficient comparing to the existing multi-document summarization models for different Indian languages [21].

The rest of the paper is structured as follows: Section 2 gives insight into the state of the art extractive summarization techniques. Section 3 describes the system architecture. Section 4 deals with description of dataset. Section 5 discusses performance evaluation. Finally, Section 6 concludes the work.

2. Related Works

The creation of a single summary from Multiple documents has gained interest since the 1990s, mostly in news articles. Popular Internet news services, for example, Google news or Alta vista news, present clusters of related articles allowing readers to find all stories on a given topic easily. However, these services do not produce summaries. SUMMONS is the first multi-document summarizing system developed at Columbia University [22]. SUMMONS generates the summaries from multiple documents by merging relevant information about each identified event. Several attempts have been made in developing text summarization systems using different approaches. The common approaches used in literature are the statistical approach, topic-based approach, graph-based approach, and approaches based on machine learning [9]. The statistical-based approach extracts most salient sentences based on the shallow features of text such as the sentence's resemblance to the title, sentence position, presence of numerical data in the sentence, presence of proper nouns (named entities), TF-IDF (Term Frequency Inverse document frequency). Each of the above features assigns some weight to the words. Based on these weights, the scores are assigned to the sentences, and then highly scored sentences are chosen to generate the summary. This technique is language-independent; it can summarize a text in any language. It does not require additional linguistic knowledge.

Topic-based approaches infer topics by observing the distribution of words across documents. LDA is the first topic model that can solve multi-document summarization, and the researchers continuously improve this algorithm. Wu et al. proposed a topic modeling-based approach to extractive automatic summarization [17]. They extracted the candidate sentences related to topic words from a preprocessed novel document. They came up with a critical evaluation function to pick a subset of sentences from the candidate sentences to get a unique summary. This technique requires additional linguistic knowledge.

Many Automatic Text Summarization (ATS) systems apply the graph-based methods for extractive text summarization such as LexRank [23], TextRank [18], and TextRankExt [24]. Graph-based approaches have achieved robust and promising results [25]. The common processing steps for a graph-based extractive method include: Representing the text elements (sentences or words) as nodes and the semantic similarity between each sentence pair as a weighted edge [26]. Using a ranking algorithm like PageRank to rank each node [27]. Selecting the top-ranked sentences for the summary. Hark et al. proposes a weighted and undirected graph model such that sentences represent the graph nodes, and edges between nodes represent the number of common words [28]. The graph-based methods captures redundancy, improves coherency, are language independent, and are robust to domain variations [29,30].

Some extractive MDS systems use deep-learning-based methods. Nallapati et al. propose a model called "SummaRuNNer" [31]. They applied an RNN-based model by handling extractive summarization as a sequence classification problem, and they did not use an attention mechanism. Hierarchical Structured Self Attentive Extractive Summarization Model (HSSAS), which uses attention mechanisms in both word and sentence layers to create sentences and document embeddings [32]. In both SummaRuNNer and HSSAS, each sentence is handled sequentially in the order same as the input document. Then it is binary classified either it can be included in the final summary or not. Some disadvantages of deep-learning-based systems include: (1) The requirement of human efforts to manually build massive training data. (2) the adaptation problems they may suffer when tested on a different corpus other than the trained domain [32].

Even though several summarization systems are reported for the English language, only a few works are there for Indian Languages [9,33]. Malayalam Document Summarization still has low performance, due to the complex nature of the language and the unavailability of fully functional NLP tools for language processing [34]. Very few works have been carried out in SDS, the majority being extractive type. The paper [10] discusses single document summarization using a heuristic approach. Here the features considered are frequency of words and number of characters in a word. The paper [12] proposes a statistical-based approach for extractive summarization and a semantic graph-based approach for single document abstractive summarization. In the paper, [13] the author proposes a Maximum Marginal Relevance based extractive system for a single document. The paper [15] uses certain statistical based sentence specific approach for summary generation. In paper [11], a semantic framework was used for single-document summarization. Table 1 gives the summary of the studies conducted in the Malayalam Language.

This work experiments with different sentence encoding schemes such as Term Frequency-Inverse Document Frequency (TF-IDF), Word embedding with pretrained model Fasttext, and Smooth Inverse Frequency (SIF) embedding. This research uses a modified Page Rank algorithm by taking the noun count as the graph node's initial rank representing the sentence. Capitalization can aid in recognizing Nouns in English language, which is not possible in the case of Malayalam. Therefore, to extract nouns from the text, a Morphological solution is required. This work also combines MMR to reduce redundancy in the candidate summary. Performance evaluation is done by comparing it with a statistical baseline model developed. All the works reported in the literature for Malayalam language have created their own data sets for evaluation.

Table 1. Malayalam Text summarization studies in the literature.

Authors, Publication Year	Extractive/ Abstractive	Single/Multi	Method Used	Evaluation
[10] KrishnaPrasad et al., 2016	Extractive	Single	Heuristic	Automatic, ROUGE Score ROUGE1: 0.57 ROUGE 2: 0.53
[11] Kishore et al., 2016	Abstractive	Single	Semantic Representation and Sentence Framing	Manual, F-Score 0.48 on News articles
[12] Kabeer et al., 2014	Extractive Abstractive	Single	Statistical scoring Semantic graph	Automatic, ROUGE Score(avg) ROUGE1: 0.533 (Extract) ROUGE1: 0.40 (Abstract)
[13] Ajmal et al., 2015	Extractive	Single	MMR	Metric not used
[14] Rahulraj et al., 2020	Extractive	Single	Semantic Role Labelling, Self organizing Maps	Manual, F-Score: 0.75 for a small data set.
[15] Manju et al., 2016	Extractive	Multidocument	Statistical Score	Manual, F-Score: 0.45

3. System Architecture

Extractive Summarization on Multiple Malayalam Documents

The overall architecture of the system is given in Figure 1. The following section, describes the important process flow of the architecture. Documents with related topic were given as input to the MDS system. The model works by summarizing each document and then summarizing the compound of the results.

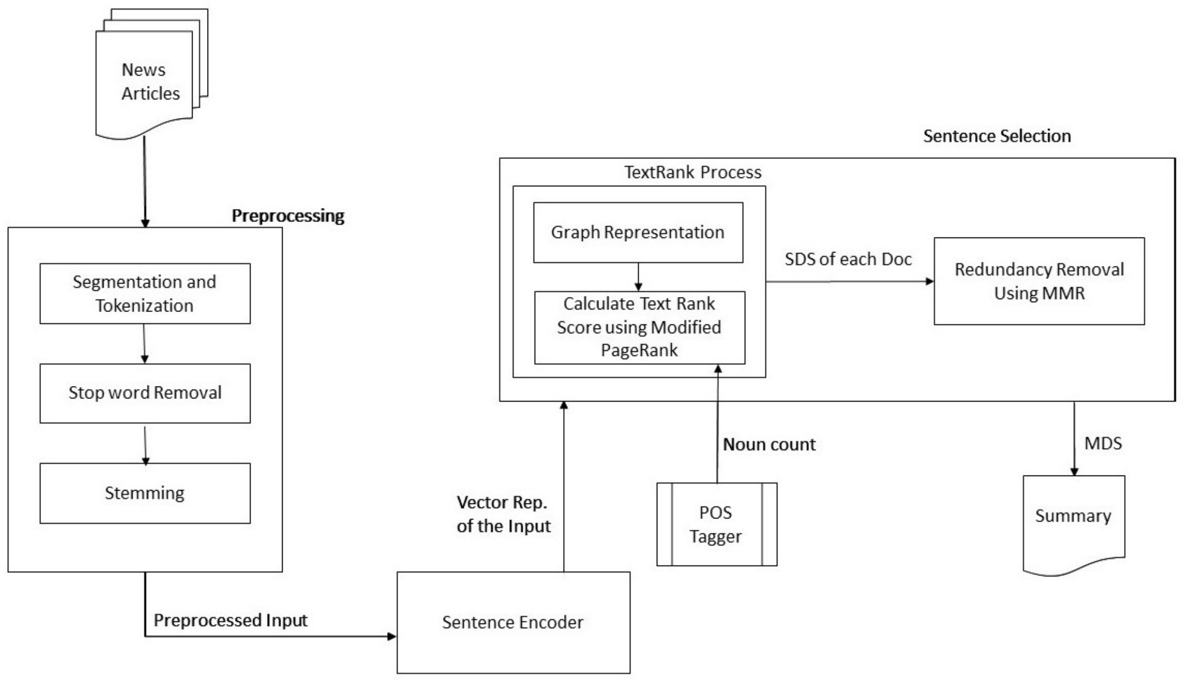


Figure 1. System Architecture of the Framework for Extractive multi-document summarization (MDS).

3.1. Preprocessing

Preprocessing consists of sentence segmentation, tokenization, stop word removal, and stemming.

- Segmentation of sentence: Each document D_i of the input set consists of a collection of sentences. Here, each D_i is segmented as $S_1, S_2, \ldots, S_n$ where each S_i denotes ith sentence in the document and n the number of sentences in the document. Language specific rules were used for sentence boundary detection and abbreviations.
- Tokenization: Terms in each sentence are tokenized to $t_1, t_2, \ldots, t_m$ where each t_j denotes the distinct terms occurring in S_i of D and m the number of words in S_i.
- Stop word removal: Stop words are generally the most common words in a language. They are filtered out using a Stop word list. Removing stop words simplifies the vectorization of the sentence.
- Stemming: It is a process of reducing inflected words to their word stems. Stemming is essential because the same document can have a word in different forms with grammatical variations. This module uses a rule-based approach that utilizes a set of suffix stripping rules. It follows the iterative suffix stripping algorithm to handle multiple levels of inflection. The stemmer used is similar to Indicstemmer [35].

3.2. Sentence Encoder

After pre-processing, the next step is to map each sentence into a fixed length vector of real numbers. A vector space representation of document D_i will be a vector representation of every term x in D_i. This numeric representation depicts the significant characteristic of the text. The proposed work uses different word embeddings like TF-IDF, Word2Vec and Smooth Inverse Frequency (SIF) to vectorize the sentence and the details are given below.

- TF-IDF representation
  The vectorization technique commonly used in information retrieval is Term Frequency Inverse Document Frequency (TF-IDF). The algorithm measures how often a term occurs (tf) in a specific document and multiplies this with the value accounting for how common the word is in the complete document collection (idf), as in Equation (1). Terms with the highest tf-idf scores are the terms in a document that are

distinctively frequent in a document. The sentence score is obtained by taking the average tf-idf of the terms in the sentence.

$$
\begin{aligned}
tf - idf(t,d,D) &= tf(t,d) * idf(t,D) \\
tf(t,d) &= \frac{f_d(t)}{\max\limits_{w \in d} fd(w)} \\
idf(t,D) &= \log\left(\frac{|D|}{d \in D : t \in d}\right)
\end{aligned} \tag{1}
$$

- Word2Vec representation.
  Words are mapped to the vectors by several methods such as Continuous Bag-of-Words model(CBOW), Continuous Skip-Gram model, etc. [36]. It extracts semantic and syntactic information about the word. Semantically similar words are mapped to nearby points in the vector space. In this work the vectorization of the terms in the document are performed using the pretrained word embedding model FastText for Malayalam, trained on Common Crawl and Wikipedia. FastText follows CBOW model with position weights in dimension 300, and character ngram of length 5. The final sentence level representation is the average of the vectors corresponding to words in the sentence.
- Smooth Inverse Frequency encoding.
  Similar to regular word embedding, sentence embedding embed a full sentence into a vector space. These sentence embedding inherit features from their underlying word embedding. Taking the average of the word embedding in a sentence tends to give too much weight to terms that are quite irrelevant. Arora et al. propose a different, surprisingly simple unsupervised approach for sentence embedding construction called the smooth inverse frequency(SIF) embedding, which they propose as a new baseline for sentence embedding [37]. Their approach is summarized in Algorithm 1 [37]. SIF takes the weighted average of the word embedding in the sentence. Every word embedding is weighted by $a/(a + p(w))$, where a is a parameter that is typically set to 0.001, and $p(w)$ is the estimated frequency of the word in a reference corpus.
  After computing all means, the first principal component is computed, and the projection of all sentence vectors on this first principal component is removed. The authors claim that this common component removal reduces the amount of syntactic information contained by the sentence embedding, thereby allowing the semantic information to be more dominant in the vector's direction.
  The SIF model first computes the weighted average of the word embedding vectors as the initial embedding vector of each sentence in the document. FastText the pretrained model for Malayalam, is used to get the word embedding. These initial sentence embedding vectors are modified with the Principal Component Analysis to get the final embedding vector of each sentence.

Algorithm 1: 1 SIF embedding algorithm by Arora et al. (2017) [37].

```
Input: Word embeddings $v_w : w \in V$, a set of sentences (bag-of-words) $S$,
        parameter $a$, and estimated probabilities $p(w) : w \in V$ of the words.
Output: : Sentence embeddings $v_s : s \in S$.
for all sentences $s \in S$ do
    $v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} \frac{a}{a+p(w)} v_w$;
end
Compute the first principal component $u$ of $v_s : s \in S$;
for all sentences $s \in S$ do
    $v_s \leftarrow v_s - uu^T v_s$
end
```

3.3. Sentence Selection

The general objective for sentence selection is to select maximally informative sentences, without information overlap between the sentences, to maximize the coverage of the original document with the summary. At sentence selection, the summarization system has to identify the appropriate collection of significant sentences that form the final summary, considering the factors redundancy and cohesion. The traditional method for selecting sentences is to pick the top-rated sentences directly. Following baseline methods are used for sentence selection. The sentence selection phase in the proposed model includes TextRank and Maximum Marginal Relevance. TextRank process is used to generate a summary from each document. To consolidate the summary results, concatenate the individual summaries and perform MMR to create the candidate summary.

- TextRank
  TextRank is a graph-based ranking algorithm used for extractive Single Document Summarization which can be extended for MDS. Graph-based methods are frequently used in text summarization as they provide handy solutions. It intends to rank each vertex in a graph by importance with regard to the connecting vertices. The ranks are updated by recursive computing until the vertex rank globally converge, as described in [18]. The philosophy is same as for the PageRank algorithm introduced by Brin and Page (1998) [27]. The more incoming links a page has, the more important it is. The more important the page, the more important the pages' outgoing links are.
  The difference when working with text instead of web-pages is how the links are formed. In TextRank the pages are represented as sentences, and the links are determined by the sentence-level similarity by a chosen metric [18]. For example, the TF-IDF representation can be used to represent sentences, and cosine similarity can be used to calculate sentence closeness. The graph obtained is an undirected graph. Traditionally PageRank which is applied on directed graph, can be applied to undirected graphs where the out-degree of a vertex is equal to the in-degree of the vertex. By iteratively scoring sentence importance by recommending edges in the graph, the most important sentences attain higher scores at convergence. The vertices with the highest probability represent the most important sentences, and are selected for the summary.
  PageRank algorithm starts by initializing the rank of each node by 1/N, where N is the number of nodes in the graph which will be same as the number of sentences in the document. The proposed work uses noun count in each sentence to modify the original PageRank algorithm. The initial rank of each node is the noun count in the sentence instead of 1/N. In the proposed model the graph is built based on a document, which has links between sentences in the document that can be indicated with weights. The formula is modified so that it takes into account edge weights when computing the score associated with a vertex in the graph. $Pr(V_i)$ is modified as in Equation (2), where w_{ij} is the weight of the edge connecting the sentences V_i and V_j which is the cosine similarity between these two sentences.

$$Pr(V_i) = (1-d) + d * \sum_{V_j \epsilon In(V_i)} \left(\frac{w_{ij} * Pr(V_j)}{|Out(V_j)| - 1} \right) \tag{2}$$

  where d is the damping factor with a value 0.85.
  To determine the initial PageRank of each sentence, noun count has to be obtained for each sentence. This is extracted using the POS Tagger developed by CDAC which tags the nouns in each sentence [38].
  Consider a document with five sentences, and the noun count in each sentence is as listed in Table 2. The cosine similarity between sentences or edge weights are recorded in Table 3. Table 4 records the new rank for the sentences after one iteration on applying the Algorithm 2 which uses a modified PageRank algorithm.

Calculation of the modified PageRank is as follows:

$$Pr(S_1) = (1-d) + d * \left(\frac{Pr(S_2)*w_{12}+Pr(S_3)*w_{13}+Pr(S_4)*w_{14}+Pr_(S_5)*w_{15}}{4} \right) \quad (3)$$

- Maximum Marginal Relevance
  Maximum Marginal Relevance is a classic unsupervised algorithm used for sentence selection. MMR is used to maximize relevance and novelty in automatic summarization [39]. MMR was actually proposed to solve the information retrieval problem by measuring the relevance between the user query Q and sentences in the document. This measure is calculated by the formula

$$MMR \overset{def}{=} Arg \max_{S_i \in R \setminus A} \left[\lambda \left(Sim_1(S_i, Q) - (1-\lambda) \max_{S_j \in A} (Sim_2(S_i, S_j)) \right) \right] \quad (4)$$

  where, Q is the query
  R is the set of documents related to the Query
  A is the subset of documents in R already selected
  $R \setminus A$ is the set of unselected documents in R
  λ is a constant in the range [0–1], for diversification of results.
  $Sim_1$is the similarity between the considering sentence and Q.
  Sim_2 is the similarity between the considering sentence and the existing sentences in the summary.
  For applying MMR in document summarization, the following changes are made to Equation (4):
  R is the set of sentences obtained from the previous process, i.e., TextRank. S_i is an element of unselected sentences in the document R, and S_j is an element of the existing summary list, and Q is formed with the terms that best describe the input set R with high TF-IDF values. Sim_1 and Sim_2 are the similarities between the sentences calculated using the cosine similarity between their corresponding TF-IDF vectors.

$$Cosine - similarity(S_i, S_j) = \frac{\sum_{k=1}^{n}(TF - IDF(w_{ik}) * TF - IDF(w_{jk}))}{\sqrt{\sum_{k=1}^{n} TF - IDF(w_{ik})^2} * \sqrt{\sum_{k=1}^{n} TF - IDF(w_{jk})^2}} \quad (5)$$

Table 2. Modified Text Rank: Initial node score.

Sentence #	S_1	S_2	S_3	S_4	S_5
Initial Rank (Noun Count)	5	3	2	1	2

Table 3. Modified Text Rank: edge weights.

	S_1	S_2	S_3	S_4	S_5
S_1		3	5	1	3
S_2	3		2	4	1
S_3	5	2		1	2
S_4	1	4	1		3
S_5	3	1	2	3	

Table 4. Modified Text Rank: node score after iteration 1.

Sentence #	S_1	S_2	S_3	S_4	S_5
Initial Rank (Noun Count)	5.78	5.56	7.9	5.56	5.56

Algorithm 2: Textrank algorithm with modified Page Rank.

```
Input: Document $D$, Summary Length $l$
Output: l Sentences with highest score
$G \leftarrow Buildgraph(D)$;
Initialize $score$ with the noun count of each node;
$Converge \leftarrow False$
while $Converge \neq True$ do
    $Converge \leftarrow True$;
    $Oldscore \leftarrow score$;
    for $s \in length(D)$ do
        score[s]← Updatescore(G,sentence    s,d = 0.85,score)
        if $|score[s] - oldscore[s]| > \varepsilon$ then
            Converge ← False
        end
    end
end
Return $l$ sentences with highest score;
```

3.4. Summary Extraction

This is the final and last step to summarization. The summarizer system selects the top k most important sentences to produce a summary.

4. Description of Dataset

There is no standard dataset available for evaluating the summarization system in this initial stage of Malayalam document summarization. Hence we created a data set with 100 document sets, and each set having three documents. Characteristics of the data set is presented in Table 5. While creating the corpus, we have taken news articles from three prominent Malayalam e-newspapers, namely Mathrubhumi, Manorama, and Madhyamam. Our data set included almost 300 news articles from March 2019 to October 2019. We selected similar items belonging to the same topic from these three sources and saved them as text files with web scraping tools. The articles belonged to sports, politics, health, and entertainment, ranging from 10 to 70 sentences. A group of PG students in the Computer Science department created the model summary for each set. A Malayalam Language expert validated this. For each group, a reference summary was created and was used for validation. During the evaluation, the reference summary was compared with the system-generated summary.

Table 5. Characteristic of the dataset.

Dataset Parameters	
Number of sets of documents	100
Number of documents in each set	3
Average number of sentences per document	21.7
Maximum number of sentences per document	70
Minimum number of sentences per document	10
Summary length (%)	40

5. Performance Evaluation

5.1. Evaluation Metric

Summarization evaluation system can be divided into two categories, namely extrinsic and intrinsic evaluation as given in Jones and Galliers [40]. Intrinsic evaluation measures the quality of the summary by comparing them to ideal summaries. Extrinsic evaluation measures how well the summaries help in performing a particular task. Extrinsic evaluation is also called a task-based evaluation. Here, we are following intrinsic evaluation, which can be done in two ways, evaluation by humans and methods for automatic evaluation. In human evaluation, human judges compare the system-generated summary with the model summary. The problem with this technique is that there is no single or ideal summary, as one can generate different summaries for the same document [41]. Therefore, manual evaluation is time-consuming and difficult. Recall Oriented Understudy for Gisting Evaluation(ROUGE) metrics are the de facto standard for automatic summarization evaluation. The ROUGE metrics are based on comparing n-grams between the system-generated summary and the reference summaries.

Experiments are done using the ROUGE [42] tool, which evaluates the performance in terms of three metrics, precision, recall and F-score. ROUGE-N is computed as in Equation (6). The scores generated by the metrics range from 0 to 1. The higher the score, the more matching content is available with the reference summary and the proposed system-generated summary.

$$ROUGE - N = \frac{\sum\limits_{S \in \{ReferenceSummaries\}} \sum\limits_{gram_n \in S} Count_{match}(gram_n)}{\sum\limits_{S \in \{ReferenceSummaries\}} \sum\limits_{gram_n \in S} Count(gram_n)} \tag{6}$$

where n stands for the length of the n-gram, $gram_n$ is the maximum number of n-grams co-occurring in a candidate summary, and $Count_{match}(gram_n)$ is the set of reference summaries.

In the current study, we have used ROUGE-1 and ROUGE-2 to evaluate the proposed summarization system. The precision, recall and F-score are adopted for evaluation, which is accomplished with the reference summary. Suppose S is the summary, and TP denotes the retrieved valid sentences, TN denotes retrieved invalid sentences, FP denotes non-retrieved valid sentences, and FN denotes the non-retrieved invalid sentences; then recall, precision and F-score are determined as follows.

- Recall: Recall is the ratio of total retrieved valid sentences to the total of retrieved and non-retrieved valid sentences in summary.

$$Recall = \frac{|TP|}{|TP| + |FP|} \tag{7}$$

- Precision: Precision is the ratio of total retrieved valid sentences to the total of retrieved valid and invalid sentences in summary.

$$Precision = \frac{|TP|}{|TP| + |TN|} \tag{8}$$

- F-score: F-score is the harmonic mean of precision and recall.

$$F - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{9}$$

5.2. Experiments and Results

All experimental processes were performed using a computer with an Intel Core i5-8250 CPU 1.80 GHz and 8 GB RAM using Python.

The model performed the MDS in two steps. For each document in the document set, a single document summary was generated using the modified Textrank algorithm. Next, a summary of summaries(candidate summary) was produced using the MMR algorithm. One by one, each document in the document set, stored as a .txt file, was given to the model. During the preprocessing phase, the stopwords were removed, and on performing stemming, each word was in its root form. The POS Tagger gave the tagged output of the preprocessed document. From this, noun counts in each sentence were taken. The sentence encoder did the vectorization of each sentence. We tried different word embedding methods like TF-IDF, Word2Vec, and SIF. For applying the Textrank process, the document was modeled as an undirected graph. Each sentence was the vertex of the graph. An edge was set between the sentences, and the similarity score was the edge weight. PageRank algorithm was run on the graph with a modification in the initial score of each vertex. The score of each vertex was initialized to the noun count in the sentence. The node obtained a new weight dependent on the node's initial rank, the connected nodes, and the edge weights connecting the nodes on every iteration. During every iteration, the most significant node weight increased faster than the other nodes, so when the summary was extracted, the sentences referring to these nodes came first.

Figure 2 shows the performance metrics of the TextRank model with modified PageRank algorithm experimenting with sentence encoding schemes TF-IDF, Fasttext, and SIF. It is evident from the graph that the values returned from SIF embedding outperformed other text embedding representations. When TF-IDF was used to vectorize the sentence, it did not consider the semantic meaning. Therefore, we integrated word embedding into our system for that purpose. Facebook's pretrained model FastText was used for this. For representing a sentence as a vector, we took the mean of all the word embeddings present in the vocabulary. Whereas in SIF embedding, the semantic information of the sentence will be more dominant in the vector representation. SIF embedding is much superior to the averaging of word vectors.

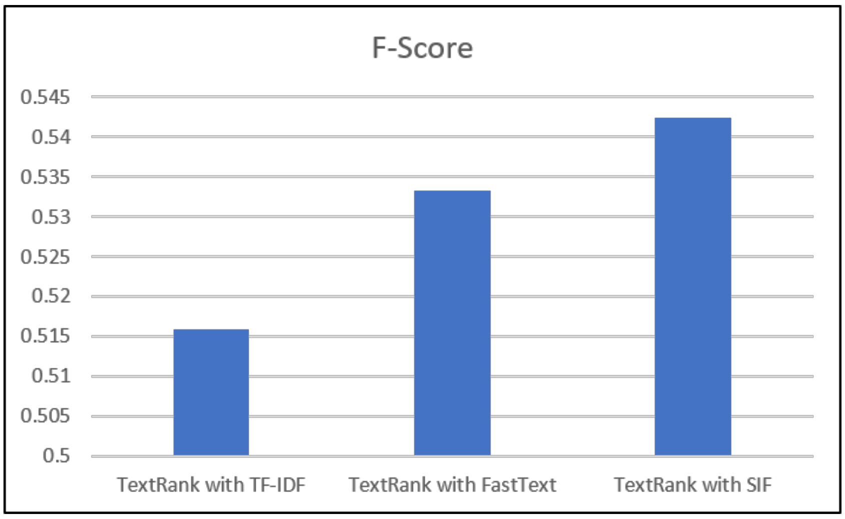
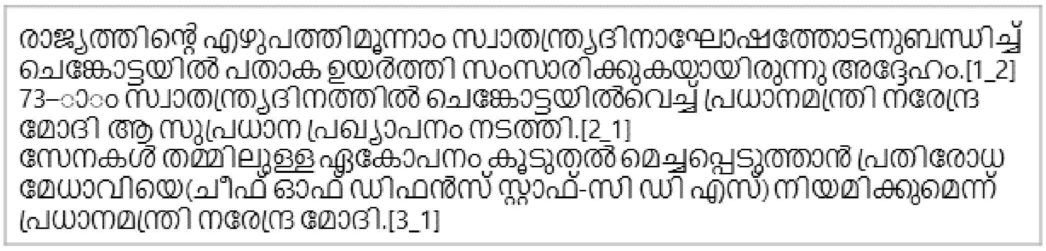


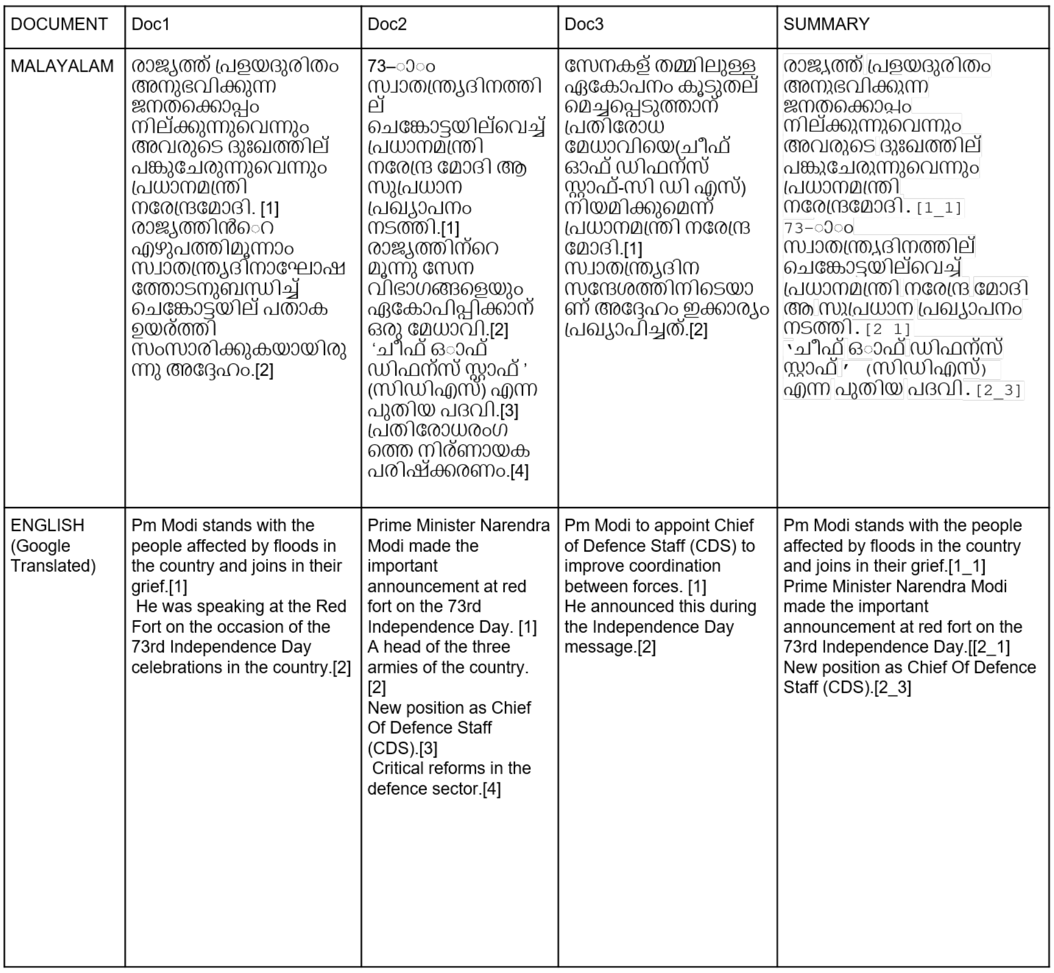
Figure 2. Comparison of the different sentence encoding schemes Term Frequency-Inverse Document Frequency (TF-IDF), Fasttext, and Smooth Inverse Frequency (SIF) when used with TextRank model.

It was observed that the proposed modified TextRank algorithm showed improved performance on SDS and MDS tasks. The SIF embedding, which used the pretrained Word2Vec model FastText for Malayalam, seems to perform well and made the model less dependent on preprocessing. With SDS, the modified TextRank with SIF embedding gives a satisfactory result. We can retain the sequence of the summary as in the original document by maintaining the sentence position. MDS, on the other hand, has the issue of redundancy in summary. This is due to the similarity among the combined texts. In this research, we combine TextRank with MMR to reduce the sentence similarity in the summarized result. For example, Figure 3 shows the snippet of the summarised text obtained on using the Modified TextRank algorithm throughout for generating Multi-document summary. Here we can see that the sentences 1_2 and 2_1 are similar. To remove redundancy, this work uses MMR in the last stage to summarize summaries, which reduces the similarity and increases the divergence among sentences. The last column in Figure 4 shows the candidate summary obtained on applying TextRank for the individual summary generation of each document in the document collection and then MMR to summarize summaries. It shows that applying MMR to create the candidate summary successfully reduced the similar sentences.



രാജ്യത്തിന്റെ എഴുപത്തിമൂന്നാം സ്വാതന്ത്ര്യദിനാഘോഷത്തോടനുബന്ധിച്ച് ചെങ്കോട്ടയിൽ പതാക ഉയർത്തി സംസാരിക്കുകയായിരുന്നു അദ്ദേഹം.[1_2]
73–ാം സ്വാതന്ത്ര്യദിനത്തിൽ ചെങ്കോട്ടയിൽവെച്ച് പ്രധാനമന്ത്രി നരേന്ദ്ര മോദി ആ സുപ്രധാന പ്രഖ്യാപനം നടത്തി.[2_1]
സേനകൾ തമ്മിലുള്ള ഏകോപനം കൂടുതൽ മെച്ചപ്പെടുത്താൻ പ്രതിരോധ മേധാവിയെ(ചീഫ് ഓഫ് ഡിഫൻസ് സ്റ്റാഫ്-സി ഡി എസ്) നിയമിക്കുമെന്ന് പ്രധാനമന്ത്രി നരേന്ദ്ര മോദി.[3_1]

Figure 3. Sample output of the multi-document summarization (MDS) system on taking Doc1, Doc2 and Doc3 of Figure 4 as inputs and using modified TextRank to get individual summary as well as the candidate summary. 1_2 refers to sentence 2 in Document 1.



DOCUMENT	Doc1	Doc2	Doc3	SUMMARY
MALAYALAM	രാജ്യത്ത് പ്രളയദുരിതം അനുഭവിക്കുന്ന ജനതക്കൊപ്പം നില്ക്കുന്നുവെന്നും അവരുടെ ദുഃഖത്തില് പങ്കുചേരുന്നുവെന്നും പ്രധാനമന്ത്രി നരേന്ദ്രമോദി. [1] രാജ്യത്തിൻെറ എഴുപത്തിമൂന്നാം സ്വാതന്ത്ര്യദിനാഘോഷത്തോടനുബന്ധിച്ച് ചെങ്കോട്ടയില് പതാക ഉയര്ത്തി സംസാരിക്കുകയായിരുന്നു അദ്ദേഹം.[2]	73–ാം സ്വാതന്ത്ര്യദിനത്തില് ചെങ്കോട്ടയില്വെച്ച് പ്രധാനമന്ത്രി നരേന്ദ്ര മോദി ആ സുപ്രധാന പ്രഖ്യാപനം നടത്തി.[1] രാജ്യത്തിന്റെ മൂന്നു സേന വിഭാഗങ്ങളെയും ഏകോപിപ്പിക്കാന് ഒരു മേധാവി.[2] ‘ചീഫ് ഓഫ് ഡിഫന്സ് സ്റ്റാഫ് ’ (സിഡിഎസ്) എന്ന പുതിയ പദവി.[3] പ്രതിരോധരംഗത്തെ നിര്ണായക പരിഷ്ക്കരണം.[4]	സേനകള് തമ്മിലുള്ള ഏകോപനം കൂടുതല് മെച്ചപ്പെടുത്താന് പ്രതിരോധ മേധാവിയെ(ചീഫ് ഓഫ് ഡിഫന്സ് സ്റ്റാഫ്-സി ഡി എസ്) നിയമിക്കുമെന്ന് പ്രധാനമന്ത്രി നരേന്ദ്ര മോദി.[1] സ്വാതന്ത്ര്യദിന സന്ദേശത്തിനിടെയാണ് അദ്ദേഹം ഇക്കാര്യം പ്രഖ്യാപിച്ചത്.[2]	രാജ്യത്ത് പ്രളയദുരിതം അനുഭവിക്കുന്ന ജനതക്കൊപ്പം നില്ക്കുന്നുവെന്നും അവരുടെ ദുഃഖത്തില് പങ്കുചേരുന്നുവെന്നും പ്രധാനമന്ത്രി നരേന്ദ്രമോദി. [1_1] 73–ാം സ്വാതന്ത്ര്യദിനത്തില് ചെങ്കോട്ടയില്വെച്ച് പ്രധാനമന്ത്രി നരേന്ദ്ര മോദി ആ സുപ്രധാന പ്രഖ്യാപനം നടത്തി. [2 1] ‘ചീഫ് ഓഫ് ഡിഫന്സ് സ്റ്റാഫ് ’ (സിഡിഎസ്) എന്ന പുതിയ പദവി. [2_3]
ENGLISH (Google Translated)	Pm Modi stands with the people affected by floods in the country and joins in their grief.[1] He was speaking at the Red Fort on the occasion of the 73rd Independence Day celebrations in the country.[2]	Prime Minister Narendra Modi made the important announcement at red fort on the 73rd Independence Day. [1] A head of the three armies of the country. [2] New position as Chief Of Defence Staff (CDS).[3] Critical reforms in the defence sector.[4]	Pm Modi to appoint Chief of Defence Staff (CDS) to improve coordination between forces. [1] He announced this during the Independence Day message.[2]	Pm Modi stands with the people affected by floods in the country and joins in their grief.[1_1] Prime Minister Narendra Modi made the important announcement at red fort on the 73rd Independence Day.[[2_1] New position as Chief Of Defence Staff (CDS).[2_3]

Figure 4. Sample Output of the Framework for Extractive MDS. As it is difficult to show the entire document, as a sample we have taken documents with only a few sentences. The index of a sentence, [2_1] denotes sentence 1 in Doc2.

To evaluate the performance of the proposed MDS framework, we did a comparison with MDS using Statistical TF-IDF and MDS using Modified TextRank algorithm with variation in sentence encoding. The absence of Malayalam MDS works demanded the development of these models. Table 6 compares the performance of Statistical Scoring with TF-IDF, Modified TextRank with TF-IDF, Modified TextRank with Fasttext, Modified TextRank with SIF and Modified TextRank with SIF + MMR.

Statistical Scoring with TF-IDF is an implementation of the work [15] while considering additional feature values. The objective of statistical scoring for sentence selection is to give each sentence an importance score, which acts as a good measure. The probability of a sentence to be present in summary is proportional to its score. A set of features represents each sentence, and the score is a function of the weighted sum of the individual feature values. The features used are TF-IDF, Sentence length, Sentence position, Noun Counts, Named entity counts, Numerical data count. The score of each sentence is expressed as in Equation (10)

$$Score(S_i) = \sum_{j=0}^{n} f_j(S_i) * w_j \tag{10}$$

From Table 6 it is evident that both for ROUGE-1 and ROUGE-2 scores, Modified TextRank with SIF embedding + MMR gave the best results. Figure 5 shows the graphical comparison of the research results. These results indicate that TextRank algorithm alone is not as good as the combination of TextRank and MMR. The summary quality is also improved on applying MMR at the last stage of the framework. We also compared the current research results reported in the Malayalam language mentioned in Table 1 with the models developed for MDS in Table 6. It is found that the Document summarization framework based on Modified TextRank and MMR produces superior results to the previous studies.

Table 6. Comparison of the Research Results.

Model	ROUGE1	ROUGE2
Statistical with TF-IDF		
Recall	0.480132	0.438446
Precision	0.441331	0.417328
F-score	0.459915	0.427626
Modified TextRank with TF-IDF		
Recall	0.536102	0.506133
Precision	0.497013	0.472813
F-score	0.515818	0.488906
Modified TextRank with Fasttext		
Recall	0.553261	0.500123
Precision	0.514732	0.491721
F-score	0.533302	0.495886
Modified TextRank with SIF		
Recall	0.568312	0.512782
Precision	0.518761	0.481371
F-score	0.542407	0.49658
Modified TextRank with SIF+MMR		
Recall	0.588137	0.543281
Precision	0.56211	0.507652
F-score	**0.574829**	**0.524863**

The bold values show the highest scores among the numbers in the table.

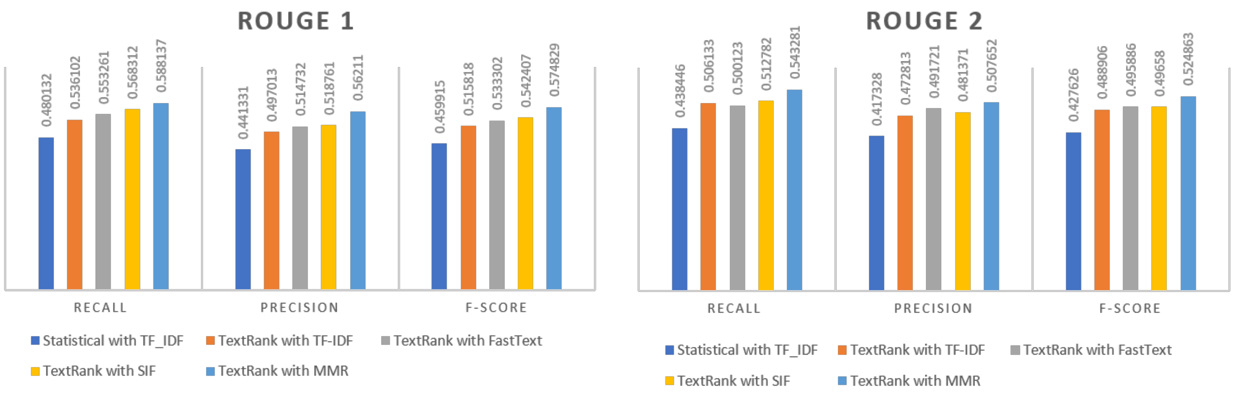


Figure 5. Graphical Comparison of the Research Results.

To measure the performance of the proposed framework, we conducted a comparison against some of the works done in Indian Languages. In [21] the authors implemented the four techniques used in Indian Languages and experimented on 100 news articles for each language. The techniques used were graph-based technique for Hindi text summarization [43], a hybrid model for Punjabi text summarization [44], Text rank-based technique for Marathi language [45], and semantic graph-based Tamil summarizer [46]. Table 7 shows that our proposed approach provides a better result than the work conducted in different Indian Languages. Even though the experiments with other languages were on Single Documents, this provides a baseline for comparison. It is evident from the table that our framework outperformed all the existing works conducted in Indian languages. It is proved that our method is efficient for the extractive summarization of multiple Malayalam documents.

Table 7. Comparison with other researches in Indian languages.

Methods	Language	Precision	Recall	F-Score
Kumar et al. (2015) [43]	Hindi	0.44	0.32	0.37
Gupta and Kaur (2016) [44]	Punjabi	0.45	0.21	0.29
Rathod (2018) [45]	Marathi	0.43	0.27	0.33
Banu et al. (2007) [46]	Tamil	0.42	0.31	0.35
TextRank with SIF + MMR (Proposed Model)	Malayalam	**0.59**	**0.56**	**0.57**

The bold values show the highest scores among the numbers in the table.

6. Conclusions

The objective of this research was to design, implement and evaluate a sentence scoring algorithm for multi document extractive summarization for Malayalam. In order to achieve this, the research utilizes the TextRank algorithm as a baseline. The TextRank algorithm was experimented with the different text embedding models, of which Fasttext and SIF embedding showed an improvement over the TF-IDF. Moreover these representations had the advantage of eliminating preprocessing steps like stemming and stop word removal. This work tried to extend the TextRank algorithm on single document summarization to multiple documents. It was found that the summary obtained after concatenating the summary of multiple documents had redundancy. Therefore, MMR algorithm was used to remove the redundancy in the summary. ROUGE-1 and ROUGE-2 were used for Summary Evaluation. Lack of standard Malayalam NLP tools was also an issue in this research. The experimental results showed that modified TextRank with SIF embedding+MMR could provide significant improvement in the quality of the generated summary. However, the issue of sentence reordering was there in the generated summary which should be probed further. Nonetheless, this issue is actually an open research problem in multi document summarization of all languages.

Author Contributions: Formal analysis, K.M.; Investigation, K.M.; Methodology, K.M.; Project administration, K.M. and S.M.I.; Software, K.M.; Supervision, S.D.P. and S.M.I.; Validation, S.D.P.; Writing—original draft, K.M. and S.M.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Luhn, H.P. The Automatic Creation of Literature Abstracts. *IBM J. Res. Dev.* **1958**, *2*, 159–165. [CrossRef]
2. Gong, Y.; Liu, X. *Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis*; Association for Computing Machinery: New York, NY, USA, 2001. [CrossRef]
3. Ouyang, Y.; Li, W.; Li, S.; Lu, Q. Applying regression models to query-focused multi-document summarization. *Inf. Process. Manag.* **2011**, *47*, 227–237. [CrossRef]
4. Radev, D.; Blair-Goldensohn, S.; Zhang, Z. Experiments in Single and Multi-Document Summarization Using MEAD. In Proceedings of the First Document Understanding Conference, New Orleans, LA, USA, 13–14 September 2001.
5. Qiang, J.P.; Chen, P.; Ding, W.; Xie, F.; Wu, X. Multi-document summarization using closed patterns. *Knowl.-Based Syst.* **2016**, *99*, 28–38. [CrossRef]
6. John, A.; Premjith, P.; Wilscy, M. Extractive multi-document summarization using population-based multicriteria optimization. *Expert Syst. Appl.* **2017**, *86*, 385–397. [CrossRef]
7. Widjanarko, A.; Kusumaningrum, R.; Surarso, B. Multi document summarization for the Indonesian language based on latent dirichlet allocation and significance sentence. In Proceedings of the 2018 International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, Indonesia, 6–7 March 2018; pp. 520–524.
8. Fang, C.; Mu, D.; Deng, Z.; Wu, Z. Word-sentence co-ranking for automatic extractive text summarization. *Expert Syst. Appl.* **2017**, *72*, 189–195. [CrossRef]
9. Gambhir, M.; Gupta, V. Recent automatic text summarization techniques: A survey. *Artif. Intell. Rev.* **2017**, *47*, 1–66. [CrossRef]
10. Krishnaprasad, P.; Sooryanarayanan, A.; Ramanujan, A. Malayalam text summarization: An extractive approach. In Proceedings of the 2016 International Conference on Next Generation Intelligent Systems (ICNGIS), Kottayam, India, 1–3 September 2016; pp. 1–4.
11. Kishore, K.; Gopal, G.N.; Neethu, P. Document Summarization in Malayalam with sentence framing. In Proceedings of the 2016 International Conference on Information Science (ICIS), Kochi, India, 12–13 August 2016; pp. 194–200.
12. Kabeer, R.; Idicula, S.M. Text summarization for Malayalam documents—An experience. In Proceedings of the 2014 International Conference on Data Science & Engineering (ICDSE), Kochi, India, 26–28 August 2014; pp. 145–150.
13. Ajmal, E.; Rosna, P. Summarization of Malayalam Document Using Relevance of Sentences. *Int. J. Latest Res. Eng. Technol.* **2015**, *1*, 8–13.
14. Raj, M.R.; Haroon, R.P.; Sobhana, N. A novel extractive text summarization system with self-organizing map clustering and entity recognition. *Sādhanā* **2020**, *45*, 32.
15. Manju, K.; David, P.S.; Idicula Sumam, M. An extractive multi-document summarization system for Malayalam news documents. In Proceedings of the 1st EAI International Conference on Computer Science and Engineering, Penang, Malaysia, 11–12 November 2016; p. 218.
16. Ko, Y.; Seo, J. An effective sentence-extraction technique using contextual information and statistical approaches for text summarization. *Pattern Recognit. Lett.* **2008**, *29*, 1366–1371. [CrossRef]
17. Wu, Z.; Lei, L.; Li, G.; Huang, H.; Zheng, C.; Chen, E.; Xu, G. A Topic Modeling Based Approach to Novel Document Automatic Summarization. *Expert Syst. Appl.* **2017**, *84*, 12–23. [CrossRef]
18. Mihalcea, R.; Tarau, P. Textrank: Bringing order into text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 404–411.
19. Elbarougy, R.; Behery, G.; El Khatib, A. Extractive Arabic Text Summarization Using Modified PageRank Algorithm. *Egypt. Inform. J.* **2020**, *21*, 73–81. [CrossRef]
20. Goldstein, J.; Carbonell, J. *Summarization: Using MMR for Diversity-Based Reranking and Evaluating Summaries*; Technical Report; Language Technologies Institute at Carnegie Mellon University: Pittsburgh, PA, USA, 1998.
21. Verma, P.; Verma, A. Accountability of NLP Tools in Text Summarization for Indian Languages. *J. Sci. Res.* **2020**, *64*, 258–263. [CrossRef]
22. Radev, D.R.; McKeown, K.R. Generating Natural Language Summaries from Multiple On-Line Sources. *Comput. Linguist.* **1998**, *24*, 469–500.
23. Erkan, G.; Radev, D. Lexpagerank: Prestige in multi-document text summarization. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 365–371.

24. Barrera, A.; Verma, R. *Combining Syntax and Semantics for Automatic Extractive Single-Document Summarization*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 366–377. [CrossRef]
25. Uçkan, T.; Karcı, A. Extractive multi-document text summarization based on graph independent sets. *Egypt. Inform. J.* **2020**, *21*, 145–157. [CrossRef]
26. De la Peña Sarracén, G.L.; Rosso, P. *Automatic Text Summarization Based on Betweenness Centrality*; Association for Computing Machinery: New York, NY, USA, 2018. [CrossRef]
27. Page, L.; Brin, S.; Motwani, R.; Winograd, T. *The PageRank Citation Ranking: Bringing Order to the Web*; Technical Report; Stanford InfoLab: Stanford, CA, USA, 1999.
28. Hark, C.; Karcı, A. Karcı summarization: A simple and effective approach for automatic text summarization using Karcı entropy. *Inf. Process. Manag.* **2020**, *57*, 102187. [CrossRef]
29. Nasar, Z.; Jaffry, S.W.; Malik, M.K. Textual keyword extraction and summarization: State-of-the-art. *Inf. Process. Manag.* **2019**, *56*, 102088. [CrossRef]
30. Moratanch, N.; Chitrakala, S. A survey on extractive text summarization. In Proceedings of the 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), Chennai, India, 10–11 January 2017; pp. 1–6. [CrossRef]
31. Nallapati, R.; Zhai, F.; Zhou, B. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17, San Francisco, CA, USA, 4–9 February 2017; pp. 3075–3081.
32. Al-Sabahi, K.; Zuping, Z.; Nadher, M. A Hierarchical Structured Self-Attentive Model for Extractive Document Summarization (HSSAS). *IEEE Access* **2018**, *6*, 24205–24212. [CrossRef]
33. Dhanya, P.M.; Jathavedan, M. Article: Comparative Study of Text Summarization in Indian Languages. *Int. J. Comput. Appl.* **2013**, *75*, 17–21.
34. Sunitha, C.; Jaya, A.; Ganesh, A. A study on abstractive summarization techniques in indian languages. *Procedia Comput. Sci.* **2016**, *87*, 25–31. [CrossRef]
35. Thottungal, S. Indic Stemmer. 2019. Available online: https://silpa.readthedocs.io/projects/indicstemmer (accessed on 12 March 2019).
36. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
37. Arora, S.; Liang, Y.; Ma, T. A simple but tough-to-beat baseline for sentence embeddings. In Proceedings of the ICLR 2017, Toulon, France, 24–26 April 2017.
38. Natural Language Processing at KBCS, CDAC Mumbai. Available online: http://kbcs.in/tools.php (accessed on 18 May 2019).
39. Verma, P.; Om, H. A novel approach for text summarization using optimal combination of sentence scoring methods. *Sādhanā* **2019**, *44*, 110. [CrossRef]
40. Jones, K.S.; Galliers, J.R. *Evaluating Natural Language Processing Systems: An Analysis and Review*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 1995; Volume 1083.
41. Ibrahim Altmami, N.; El Bachir Menai, M. Automatic summarization of scientific articles: A survey. *J. King Saud Univ. Comput. Inf. Sci.* **2020**. [CrossRef]
42. Lin, C.Y. Rouge: A package for automatic evaluation of summaries. In Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain, 25–26 July 2004; pp. 74–81.
43. Kumar, K.V.; Yadav, D. An improvised extractive approach to hindi text summarization. In *Information Systems Design and Intelligent Applications*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 291–300.
44. Gupta, V.; Kaur, N. A novel hybrid text summarization system for Punjabi text. *Cogn. Comput.* **2016**, *8*, 261–277. [CrossRef]
45. Rathod, Y.V. Extractive Text Summarization of Marathi News Articles. *Int. Res. J. Eng. Technol.* **2018**, *5*, 1204–1210.
46. Banu, M.; Karthika, C.; Sudarmani, P.; Geetha, T. Tamil document summarization using semantic graph method. In Proceedings of the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007), Sivakasi, India, 13–15 December 2007; Volume 2, pp. 128–134.