Automatic Summarization of Malayalam Documents using Text Extraction Methods

Jisha P Jayan

Langauge Computing Department, Centre for Development of Imaging Technology, Kerala, India

Govindaru V

Langauge Computing Department, Centre for Development of Imaging Technology, Kerala, India

Corresponding author: Jisha P Jayan, Email: jishapjayan@gmail.com

Text summarization is a technique for reducing lengthy passages of text into smaller portions. The goal is to develop a logical and fluent summary that only includes the document's major ideas. It's a important task in Natural Language Processing (NLP) that undoubtedly has a significant impact on synthesization of lengthy documents. With the rise of the digital documentation and publication, devoting time to thoughtfully read an article, document, or book in order to determine its relevance is no longer an option, especially given time constraints. In machine learning and NLP, automatic text summarization is a generic problem. A comparison of text summarization applying the Term Frequency —Inverse Document Frequency, Latent Semantic Analysis and TextRank algorithms for the Malayalam language is presented in this research paper.

Keywords: Abstractive Summarization, Extractive Summarization, Term Frequency —Inverse Document Frequency, Latent Semantic Analysis, Text Rank.

2021. In Raju Pal & Praveen K. Shukla (eds.), *SCRS Conference Proceedings on Intelligent Systems*, 443–457. Computing & Intelligent Systems, SCRS, India. https://doi.org/10.52458/978-93-91842-08-6-42

1 Introduction

With the exponential growth of information available on web, it's more significant nowadays to gather related information and store it in a clear and precise manner. Automatic document/text summarizing becomes an essential and important activity in this context. Automatic Text Summarization is prevalent problem in machine learning and natural language processing in which a computer program shortens lengthy texts and provides summaries with out loosing its essence. Some of the primary issues faced in the text summarization process are text identification, interpretation, and summary production, as well as examination of the created summary. Identifying important phrases in the document and exploiting them to uncover relevant information to add in the summary are critical tasks in extraction-based summarizing.

Extractive Summarization and Abstractive Summarization are two alternative techniques for text summarization. The former approaches attempt to summarize articles by recognizing key sentences or phrases from the source or original text and putting together contents to create a shortened version. The summary is then created using the retrieved sentences. Unlike extraction, the latter technique depends on advanced natural language techniques to paraphrase and reduce parts of a document. Abstractive machine learning algorithms can construct new phrases and sentences in order to capture the meaning of the source content. When done effectively in deep learning issues, such abstraction can help overcome grammatical mistakes. Summarization can further classified into Single document and Multi document summarization, Generic and Query based summarization etc. The significant sentences relating to a given area/topic from various sources are retrieved to create a summary in multi document summarising, whereas the relevant phrases/concepts from one document are analyzed in single document summarization. In generic summarising, the entire concept or idea of the document is extracted, but in query based summarization, phrases/sentences associated to the terms in the query are selected to create a summary.

With so much data moving in the digital world, machine learning algorithms that can automatically condense lengthy texts and deliver accurate summaries that pass the intended ideas fluently are required terms. Furthermore, using text summarization reduces reading time, speeds up the research process, and expands the quantity of information that may fit in a given space. Text summarizing saves content editors time and effort by generating automatic summaries, which would otherwise be spent manually creating article summaries. It decreases the amount of work required by the user to find the relevant information. It allows the user to quickly scan a text for accurate, concise, and precise information. Automatic software, on the other hand, does not miss important subtleties that the human eye does. The automatic text summarization technique makes it simple for the user to acquire all of the important information in a document.

In the field of extractive summarization, a large number of research studies have been conducted in foreign languages, but only a few studies in the field of abstractive summary have been happened. It is quite difficult to create an abstractive summarization in Dravidian languages due to their agglutinative nature [1]. Malayalam, a Dravidian languages, is an Indian language spoken mostly in Kerala. Due to numerous reasons, a good summarizer is not available in Malayalam. Because of its agglutinative nature, Malayalam language processing is complex, and many words are found as compound words. The language's morphology is heavily inflectional, derivative, and compounding. In contrast to English, Malayalam letters do not have upper or lower case, which, if present, would aid in the identification of pronouns.

Furthermore, the same word may appear in many phrases with different inflectional and morphological alterations, and the same meaning can be articulated in different sentences using synonyms. The absence of freely available and publicly accessible corpora is a significant barrier to the development of NLP tools for this language. Further, study in Malayalam is difficult due to a lack of comprehensive and efficient preprocessing tools. Although there have been a few research work in the area of extractive summarizing, an effective abstractive summarization system for Malayalam is still has to be developed.

The proposed study described in this article investigates automatic extractive text summarization strategies for Malayalam documents using Term Frequency —Inverse Document Frequency (TF-IDF), Latent Semantic Analysis (LSA) and Text Rank algorithm for Malayalam documents. The objective of adopting an extractive technique is to create a summary from the input document by picking the highest-ranking sentences.

2 Literature Surveys

There is a high need for document summarizing into short, understandable summaries because of the abundance of unstructured data and the requirement for condensed information. Various studies on abstractive and extractive summarization of text have been conducted in response to this requirement. In the area of abstractive summarization, just a few studies have been conducted in Indian languages. There are two types of techniques in these works: syntactic and semantic methods. A syntactic parser is used to examine the text in syntactic summarization, however it loses or misses the semantic representation of the input content. The input text, on the other hand, is represented semantically in the semantic method.

Baxendale [2] presents a simple approach for extracting sentences from a document's title, first and last sentences, or each paragraph. He claimed that the initial sentences of newspaper articles have a high possibility of being included in a summary. But in technical papers the last sentence or concluding parts are having high likelihood to include in summary. According to Lin and Hovy [3], the Baxendale position approach is not suited for phrase extraction in various domains. A sentence's discourse structure varies depending on the domain. The fundamental drawback of this approach was that it was domain-specific.

SweSum [4] was the first web-based automatic Swedish text summarizer. On the World Wide Web, it summarizes Swedish news articles in HTML-based text format. Texts in Danish, Norwegian, Spanish, English, French, Italian, Greek, Persian, and German are also supported by SweSum. The summary statements are generated using statistical, linguistic, and heuristic methods. Conroy and O'Leary [27] used the Hidden Markov model to extract sentences. According to the system, the likelihood of a sentence being included in a summary is determined by whether the preceding sentence is relevant to the next sentence. MEAD [26] is a system that calculates a sentence's score based on various parameters such as similarity to the centroid, sentence location, sentence length, and so on.

Balaji et al. [5] presented a semi-supervised bootstrapping approach for identifying essential abstractive summarization components. In the proposed methodology a completely linked semantic network of a document is offered as the input. To make a comprehensive semantic network, first generate semantic graphs for phrases, synonym concepts and co-referring entities are then used to connect them. A simplified and refined spreading activation algorithm determines the direction of node traversal, where the relevance of nodes and edges is assessed based on the node and its associated edges within the consideration. To generate a summary, the most important nodes and edges are chosen. For abstractive summarization that deals with the multi-documents, Khan et al. [6] came up with a semantic graph-based technique with an enhanced ranking algorithm. The graph nodes in the semantic graph represent predicate argument structures (PASs), which constitute the semantic structure of sentences and are automatically discovered via semantic role labelling. The edges of the network show similarity weight, which is calculated using PAS semantic similarity. The essential nodes and edges which could be used to describe the summary from this structure are identified using a graph ranking approach. For sentence extraction, the Farisum [23] system used the SweSum architecture. It's a Persian web-based summarizer. The Farisum used the same architecture as SweSum, except that the it did not employ a lexicon. Azmi and Al-Thanggam [24] suggested an extraction technique-based methodology for creating a summary in Arabic. It presented a summary method based on Rhetorical Structure Theory. Following that, the summary sentences are ranked, with the highest ranked sentences being chosen as the summary. For Hindi and Punjabi Text Summarization, Gupta [25] presented a hybrid approach. This method determines a sentence's feature score, and high-scoring sentences are gathered for a summary.

Kabeer and Idicula [7] applied both statistical and semantic graph-based methods to summarise Malayalam documents. The most significant sentences are retrieved using statistical metrics in the statistical sentence scoring approach. Sentences are changed into clauses using a semantic graph-based method. Subject, object, and verbs are retrieved from these phrases. A semantic graph for the entire text is created using these triples. Using the semantic graph reduction approach, a subgraph is created from this graph. This subgraph represents the generated summary for the sentences. The final summary sentences are formed from the subgraph. Similar to LexPageRank, Manju K et al [8] suggested a graph-based multi-document extractive summarization strategy for Malayalam, a Dravidian language. The documents are represented as a weighted undirected graph in the proposed model. The Page Rank algorithm is used to choose the most important sentences for the summary.

For Malayalam text summary, Kanitha and Shanavas [9] adopted a statistical graph theoretic technique. The nodes indicate the sentences, while the edges denote the relationships. A graph's cardinality indicates the importance of sentences. By choosing a threshold value, the important summary sentences are chosen based on their cardinality. In their paper, Kishore et al. [10] employed the Karaka tree as an appropriate semantic representation for describing the phrases in the document. Because it resembles the Malayalam grammatical specification, the Karaka tree, which is primarily focused on Panini's grammar framework (PGF), is an excellent representation for describing Malayalam sentences. Based on sentence aggregation criteria, the Karaka trees are merged. A phrase extractor module was also employed, which uses statistical methodologies to determine the document's essential ideas. As a result, the system combines the advantages of both extractive and abstractive techniques.

Kallimani et al. [11] deal mostly on statistical methodologies in their Kannada text summarising study. On the basis of key word extraction, Jayashree et al. proposed a text Summarizer for Kannada. The keywords for the summary were extracted using Inverse-Document Frequency methods using Term-Frequency. In their research, Banu M et al. [12] adopted a semantic graph reduction technique. Individual sentences' semantic triples of Subject, Object, and Predicate are extracted to construct a semantic graph for the entire content. The number of nodes in these semantic triples is reduced by a semantic normalization method, resulting in a sub-graph. This sub-graph serves as the foundation for building abstractive summaries.

A conceptual model for abstractive text summarization was proposed by Nikita and Sharvari [13]. An approach for generating an abstractive summary for the input document is described using a graph reduction technique. This study offers a system that recognizes a document as input and processes it by creating a rich semantic graph, which is then reduced to generate a summary. Sunitha.C et al. [14] used Paninian Grammar and Karaka theory to determine semantic roles in the text. These semantic roles can be used to identify the subject, object, and predicate, which will be used for text summarization.

A text-based clustering methodology was proposed by Basha and Kaliyamurthie [15]. The cosine similarity is used to compute the similarities between the words after the dataset has been preprocessed. The similarities between the components are analyzed, and vector data is generated. The clustering particle is calculated from the vector data. The approach proposed by Amoli [16] is a hybrid algorithm based on summarization. They normalised the text to eliminate words before calculating the TF-IDF score for each word. Following this calculation, separate clusters are formed, and the most important weight sentences are selected for summarization from each cluster. Khan et al. [17] designed a framework for multi-documents abstractive summarization; the technique selects summary contents from the semantic representation of the source documents rather than from the source document phrases. Using semantic role labelling, the contents of the actual documents are represented by predicate argument structures in this framework. The content for the summary is chosen by ranking the predicate argument structures that are based on optimal features and then utilising language generation to generate phrases from the predicate argument structures.

3 Algorithms used in this Research work.

3.1 TF-IDF Algorithm

The term frequency —inverse document frequency, or TF-IDF, is a numeric metric used to estimate the relevance of a word in a document based on how frequently it appears in that document and a set or collection of documents. This means that if a term appears frequently in a document, it must be significant, so the term needs to be assigned with a high score. If a word appears in too many other texts, it is most likely not a unique identifier, therefore it will take a lower score
The following is the formula for calculating tf and idf:

$$TF(w) = \frac{Number\ of\ times\ term\ w\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document} \tag{42.1}$$

$$IDF(w) = \log_e \frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ term\ w\ in\ it} \tag{42.2}$$

The tfidf for a word can be calculated as follows:

$$TF - IDF(w) = TF(w) * IDF(w) \tag{42.3}$$

3.2 TextRank Algorithm

TextRank is a graph-based method to summarize a document that uses a graphical representation to locate important terms. The TextRank algorithm is based on the PageRank [22] algorithm used by Google for search results [18]. The nodes of the graph are represented by the sentences in the document. The TextRank algorithm ranks a sentence as a whole. Important sentences are chosen based on the similarity index. Term Frequency - Inverse Document Frequency is used to calculate the word importance in any sentence from the source document.

3.3 LSA Algorithm

The algebraic-statistical method of detecting the meaning of words and the similarity of sentences based on information about the context in which the words are used is known as Latent Semantic Analysis (LSA). It keeps track of information about which words are often used in each given sentence, while preserving information of common words across sentences. The more frequent words between sentences, the more semantically linked those sentences are. For detecting semantically comparable words and sentences, the LSA approach employs Singular Value Decomposition (SVD). SVD is a method for modelling word and sentence relationships. Input matrix construction, singular value decomposition, and sentence selection are the three basic steps in all LSA-based summarization systems.

4 Summarization of Malayalam Documents

Malayalam has a rigid and wide grammar structure. Computationally understanding the language structure, determining the meaning of sentence, extracting relationships and implementing the grammar is a tedious task. Now a day′s, the internet/web has a large number of Malayalam documents. Finding the useful data from various web pages, on the other hand, is heavy task. Reading each and every pages and exploring the useful data is time consuming. An efficient summarizer can handle these tasks efficiently. The architecture for developing the text summarization is depicted in Figure: 1.

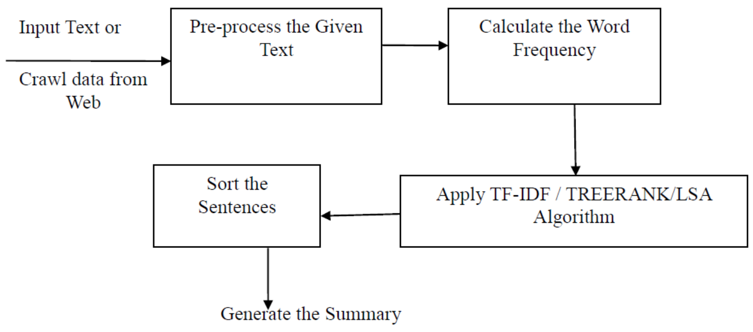
Input Text or
Crawl data from Web
Pre-process the Given Text
Calculate the Word Frequency
Apply TF-IDF / TREERANK/LSA Algorithm
Sort the Sentences
Generate the Summary

Figure 1: Architecture of Malayalam Text Summarization

In all proposed methods, the Malayalam document in the text format is taken as input. The input text is subjected to different types of pre-processing techniques, which is the most important step in obtaining a consistent and accurate analytical result. The pre-processing techniques applied to the given input includes, removing the special characters, digits and stop words from the input text. Then tokenization performs the splitting of a text document into a sequence of words and the word list is generated.

4.1 Using TF-IDF Algorithm

After the preprocessing steps, the frequency of words is calculated. A dictionary is created where all the words and its frequency is stored as key value pair. Then the sentence score is calculated and a dictionary is created where each sentence index with corresponding score values are stored as key value. Individual sentences from the tokenized sentences are chosen and the sentence score is computed to identify the most significant sentences. The calculation is carried out by passing each sentence, frequency of word and the list of pre-processed sentences. The word frequency, sentence and entire document sentences are used to calculate the score for each sentence. Once the scores have been calculated, the top sentences are included in the summary depending on the retention rate defined by the user.

4.2 Using TextRank Algorithm

For the input document, after the pre-processing steps, extract all the sentences from the text document. Then each and every sentence is represented into vector format. After this, the similarities between sentence vectors are computed and stored in the matrix format. For the calculation of sentence rank, this similarity matrix is turned into a graph, with sentences as vertices and their associated similarity scores as edges. Using the top N sentences depending on their rankings calculated, the summary for the given input document is generated.

4.3 Using LSA Algorithm

The term-frequency vector is produced using the TF matrix, and the term-frequency vector for each selected word is generated from the wordlist obtained during data preparation. The IDF is then calculated using the document-frequency vector and total amount of sentences. Multiplying the TF matrix with the IDF vector yields the TF-IDF matrix. The TF-IDF matrix's rows and columns are words and sentences, respectively. After the computation of TF-IDF matrix, SVD is used to factor the matrix and extract the essential sentences from the right singular matrix. To reduce the dimension of the term-by-document matrix, SVD is applied. The decomposition of the matrix generates three matrices: U, S, and V-Transpose [19]. V-Transpose is the right singular vector matrix, S is the diagonal matrix of non-negative singular values ranked in descending order, and U is the left singular vector matrix. The output summary is generated when the input matrix has been created and the singular value decomposition of the matrix has been completed.

5 Results and Discussion

For the testing of the algorithms discussed, five different datasets are collected by scraping the contents from Malayalam Wikipedia. Each document contains around 100 sentences. These documents that are collected are saved in UTF-8 format as separate text files. One summary is generally viewed a reference summary for each document and is prepared manually. The evaluation of these algorithms is entirely focused on this reference summary.

The evaluation criteria chosen for the automated evaluation of the generated summary are recall, precision, and F-measure. Rough is used to calculate accuracy, recall, and F-score in order to compare the outcomes of the proposed framework. Recall Oriented Understudy for Girting Evaluation (ROUGH) [20] is a collection of metrics for assessing /evaluating automatic text summarization and machine translations. The metrics, in general, compare an automatically generated summary to a reference summary or a set or group of reference summaries. Between the ideal summary and the extracted summary, the ROUGE evaluation method relies on n-gram co-occurrence, the longest common subsequence, and the weighted longest common subsequence [21]. ROUGE-N is a n-gram based ROUGE score that compares n-grams in the ideal and reference summaries. Rouge-1 measures the overlap of unigram between the system summary and reference summary which is given in Table: 1.

Table 1: Comparison of performance measures

Models	Evaluation metrics	Data 1	Data 2	Data 3	Data 4	Data 5
3*TF-IDF	Precision	55.55	73.23	77.29	56.52	69.45
	Recall	50.31	67.83	71.69	48.96	60.34
	F-Score	55.73	70.426	74.38	52.46	68.58
3*LSA	Precision	64.65	79.09	70.56	59.31	71.56
	Recall	55.83	72.98	75.21	53.65	68.45
	F-Score	59.92	75.91	72.81	56.34	69.97
3*TextRank	Precision	64.33	79.68	59.45	49.56	64.16
	Recall	58.43	70.69	63.27	52.83	61.98
	F-Score	61.24	74.92	61.3	51.14	63.02

Table: 2 compares precision, recall, and the F-measure using TF-IDF, LSA, and TextRank, and Figure: 2 depicts a graphic representation of the same comparison analysis.

Table 2: Comparison of average performance measures

	TF-IDF	LSA	TextRank
Precision	66.41	69.03	63.44
Recall	59.83	65.22	61.44
F-Score	62.32	66.99	62.32

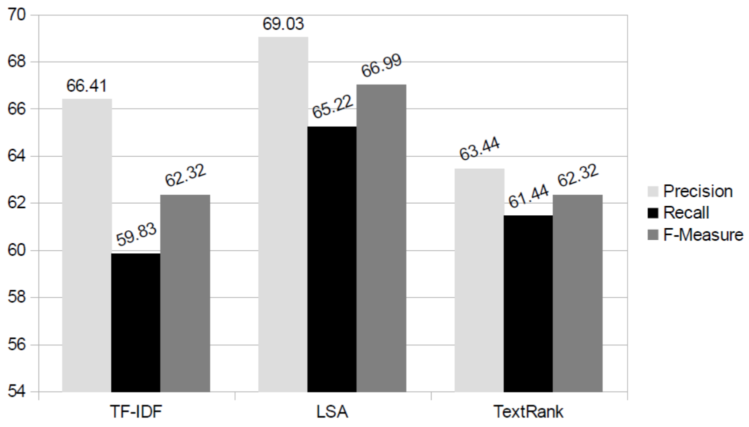

Figure 2: Average Performance Comparison

6 Conclusion

The emergence of text-based resources has proved problematic in finding information that meets the user's demands. Text summarizing techniques are proposed and evaluated in order to overcome this challenge. Our summarization research began with the extraction of basic features and expanded to include lexical chains, statistical approaches, graph-based approaches, and algebraic solutions. In this research paper, three different algorithms namely TF-IDF, LSA and

TextRank are compared for text summarization with the Malayalam language. The methods discussed in this study are domain independent. Standard evaluation metrics which include precision, recall, and f-score are used to measure the performance of the algorithms. These scores were generated using the ROUGE evaluation tool. ROUGE includes 5 evaluation metrics and only ROUGE-1 is considered here. The other ROUGE measures can be taken to generate the scores as an extension to this study. Further, this work can be extended by using LSA with other algorithms and TextRank with wordembedding′s or cosine similarity. The results obtained are very much promising and can be extended to abstractive summarization of Malayalam documents.

References

[1] Sunitha, C., Jaya, A. and Ganesh, A. (2016). A study on abstractive summarization techniques in Indian languages. *Procedia Computer Science*, 87:25-31.

[2] Baxendale, P.B. (1958). Machine-made index for technical literature—an experiment. *IBM Journal of research and development*, 2(4):354-361.

[3] Hovy, E. and Lin, C.Y. (1999). Automated text summarization in SUMMARIST. *Advances in automatic text summarization*, 14:81-94.

[4] Hahn, U. and Mani, I. (2000). The challenges of automatic summarization. *Computer*, 33(11):29-36.

[5] Balaji, J., Geetha, T.V. and Parthasarathi, R. (2016). Abstractive summarization: A hybrid approach for the compression of semantic graphs. *International Journal on Semantic Web and Information Systems*, 12(2):76-99.

[6] Khan, A. et al. (2018). Abstractive text summarization based on improved semantic graph approach. *International Journal of Parallel Programming*, 46(5):992-1016.

[7] Kabeer, R. and Idicula, S.M. (2014). Text summarization for Malayalam documents—An experience. In *International Conference on Data Science & Engineering*, 145-150.

[8] Manju, K., Peter, D.S. and Idicula, S.M. (2016). Graph based extractive multi-document summarizer for Malayalam-an experiment. In *Proceedings of the World Congress on Engineering*, 1.

[9] Kanitha, D.K., Mubarak, D.M.N. and Shanavas, S.A. (2018). Malayalam text summarization using graph based method. *International Journal of Computer Science and information Technologies*, 9(2):40-44.

[10] Kishore, K., Gopal, G.N. and Neethu, P.H. (2016). Document Summarization in Malayalam with sentence framing. In *International Conference on Information Science*, 194-200.

[11] Kallimani, J.S. and Srinivasa, K.G. (2011); Information extraction by an abstractive text summarization for an Indian regional language. In *7th International Conference on Natural Language Processing and Knowledge Engineering*, 319-322.

[12] Banu, M. et al. (2007). Tamil document summarization using semantic graph method. In *International Conference on Computational Intelligence and Multimedia Applications*, 2:128-134.

[13] Munot, N. and Govilkar, S.S. (2015). Conceptual framework for abstractive text summarization. *International Journal on Natural Language Computing*, 4:39-50.

[14] Sunitha, C., Jaya, A. and Ganesh, A. (2018). Semantic Role Labeling Of Malayalam Web Documents In Cricket Domain. *Journal of Theoretical & Applied Information Technology*, 96(8):2232-2241.

[15] Basha, M.J. and Kaliyamurthie, K.P. (2017). An improved similarity matching based clustering framework for short and sentence level text. *International Journal of Electrical and Computer Engineering*, 7(1):551-558.

[16] Amoli, P.V. and Sh, O.S. (2015). Scientific documents clustering based on text summarization. *International Journal of Electrical and Computer Engineering*, 5(4):782-787.

[17] Khan, A., Salim, N. and Kumar, Y.J. (2015). A framework for multi-document abstractive summarization based on semantic role labelling. *Applied Soft Computing*, 30:737-747.

[18] Petasis, G. and Karkaletsis, V. (2016). Identifying argument components through textrank. In *Proceedings of the Third Workshop on Argument Mining*, 94-102.

[19] Singular Value Decomposition (SVD) tutorial.
https://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm

[20] Lin, C.Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, 74-81.

[21] Lin, C.Y. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language*

Technology Conference of the North American Chapter of the Association for Computational Linguistics, 71-78.

[22] Page, L. et al. (1998). The pagerank algorithm: Bringing order to the web. In *Proceedings of the International Conference on the World Wide Web.*

[23] Kanitha, D. K. et al. (2018). Issues in Malayalam Text Summarization. *Gender and Language in the Indian Linguistic Area*, 3(1):201-204.

[24] Ibrahim, A. and Elghazaly, T. (2012). Arabic text summarization using rhetorical structure theory. In *8th International Conference on Informatics and Systems*, 34-38.

[25] Gupta, V. and Lehal, G.S. (2013). Automatic text summarization system for Punjabi language. *Journal of Emerging Technologies in Web Intelligence*, 5(3):257-271.

[26] Radev, D. R. et al. (2004). *MEAD-a platform for multidocument multilingual text summarization.*

[27] Conroy, J.M. and O'leary, D.P. (2001). Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 406-407.