International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT2390298

Text Summarization Using Natural Language Processing

Jani Patel[1], Dr. Narendrasinh Chauhan[2], Dr. Krunal Patel[3]

[1]M. Tech. Student, Department of Artificial Intelligence, Charutar Vidya Mandal University, Anand, Gujarat, India

[2]Professor & Head, Department of Information Technology, Charutar Vidya Mandal University, Anand, Gujarat, India

[3]Associate Professor, Department of Information Technology, Charutar Vidya Mandal University, Anand, Gujarat, India

ARTICLEINFO

Article History:

Accepted: 15 April 2023
Published: 04 May 2023

Publication Issue
Volume 9, Issue 3
May-June-2023

Page Number
16-22

ABSTRACT

The availability of information today accessible in digital form has accelerated. Retrieving useful document from such large pool of information gets difficult. So, to summarize these text documents is very crucial. Text summarization is a process of minimizing the original source document to get essential information of that document. It eliminates the redundant, less important content and provides you with the vital information in a shorter version usually half a length of the original text. Creating a manual summary is a very time-consuming task. Automatic summarization helps in getting the gist of information present in a particular document in a very short period. In the comparison of all Indian regional languages, there is very less amount of work done for summarization of Hindi documents. This paper presents an effective way to summarize using a Text Rank algorithm. It focuses on summarizing single Hindi text document at a time based on natural language processing (NLP).

Keywords: Summarization, Online Information, Unstructured Information, Text Rank algorithm, F- Measure

I. INTRODUCTION

Internet exchanges a huge amount of data. In this new era, the Internet is being proliferated. So the problem of information overload has increased. Rather than reading the entire document which includes many examples, comparisons, supporting details, etc. for the readers, it is always convenient to read point to point specific gist of the document. Automatic text summarization is actually meant for this. This gives the reader a non-redundant presentation of the facts found in filtered details of the source text. Automatic text summarization is a process that extracts the most essential part of the original source document. It eliminates unnecessary, less important content and provides the essential information in a small version normally half a length of the original text. Summarization can be divided into two text categories:

Extractive Summarization vs. Abstractive Summarization.

Extractive Summarization : Extractive summarization systems form summaries by copying parts of the source text through some measure of importance and then combine those part/sentences together to render a summary. Importance of sentence is based on linguistic and statistical features.

Abstractive Summarization : Abstractive summarization systems generate new phrases, possibly rephrasing or using words that were not in the original text. Naturally abstractive approaches are harder. For perfect abstractive summary, the model has to first truly understand the document and then try to express that understanding in short possibly using new words and phrases. Much harder than extractive .

Majority of the research work done so far has been more emphasis on widely used English and other European languages. Due to the low volume of information available in the non-English language, the Indian Languages have been explored little less . However, the scenario is changing now and information in large quantities is available in different languages. The need for text summarization methods that handle Indian languages has seen growth. Hindi is written using Devanagari script and it has its own alphabet set. A Hindi root word consists of many morphological variants that are associated with inflection, making it difficult to extract a feature from Hindi texts.

In our study, we propose a system for automatic extractive summarization techniques for the creation of a summary of Hindi text documents. The system would currently produce a summary for single text documents at a time which are in Hindi. Hindi text summarization has various applications in those systems where text analysis and knowledge representation are required. This system is based on extractive summarization approach.

II.LITERATURE REVIEW

Here the relevant literature survey that uses various techniques for text summarization of Indian language documents is presented.

Arti Jain ,Anuja Arora , Jorge Morato , Divakar Yadav and Kumar Vimal Kumar [1]presented an idea to Automatic Text Summarization for Hindi Using Real Coded Genetic Algorithm . ATS works with Real Coded Genetic Algorithm which optimizes the feature weights using selection, Simulated Binary Crossover and Polynomial Mutation. RCGA selects the best chromosome which contains real valued weights of generated features, computes the distance between sentence scores and ranks the corpus sentences.

Dipali Telavane, Apurva Khude, Kartik Lakade, Mohini Chaudhari [2] presented an idea to Automatic Summarization of Hindi Text Documents Using Supervised Learning Method . Automatic extractive summarization system for summarizing the Hindi documents using Naïve-Bayes approach. Currently, the field of summarization is gaining importance as data on the internet is increasing at a very high rate. Instead of reading the entire document, reading the summary of a document is efficient.

Chetana Thaokar and Latesh Malik [3] presented an idea to summarize Hindi text using sentence extraction method. Hindi WordNet was used in the system to tag POS of words for checking Subject-Object-Verb (SOV) of the sentence. Later they used the genetic algorithm to optimize the process of summary generation and to minimize redundancy. At the end sentence ranking was used to rank the sentences according to the importance of them in their summary.

Dawinder Kaur, Rajbhupinder Kaur [4] developed a rulebased approach to generate the summary from a text document written in the Hindi language. For that, they have taken Hindi text paragraph and first perform

preprocessing on it. In the next step, they assign the weights the sentences according to the features present in sentences. Later they eliminate sentences having a lesser weight than that of minimum weight. In the end, they combine the other lines as a paragraph. They have tested on 60 documents from a different domain.

Bijal Dalwadi, Nikita Patel and Sanket Suthar [5] reviewed different text Summarization techniques available for Indian languages. They describe the process of extracting the important information and reducing the size of the original document and producing a summary by retaining important information on the original document.

Vipul Dalal and Dr. Latesh Malik [6] have created an automatic abstractive text summarization system for Hindi documents. They first performed pre-processing on a document to clean it. Later they calculate sentence features. They used the approach of bio-inspired computing. To create a summary semantic graph and particle swarm optimization algorithm was used.

Saiyed Saziyabegum and Priti S. Sajja [7] created a literature review on different approaches available for extractive summarization of any language document. They suggest various methods to do the extractive summarization of single documents. They also suggest methods to do the extractive summarization of Multilanguage document or summarization of multiple documents at once.

III.PROPOSED SYSTEM

The proposed system is automatic summarization of Hindi text documents using Text Rank algorithm. The input to the system is Hindi text documents and result i.e. output is an extractive summary of Hindi documents. Extractive summary is obtained by identifying the important sentences from the input text. The first step would be to concatenate all the text contained in the articles. Then split the text into individual sentences. Then do pre-processing on them. In the next step, find vector representation (word embeddings) for each and every sentence. Similarities between sentence vectors are then calculated and stored in a matrix. The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation.

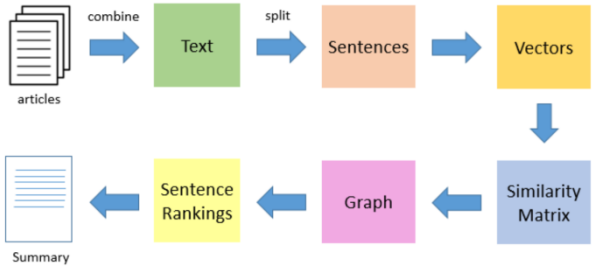


Fig. 1. Block diagram of extractive summarization system

3.1 READING FROM CSV FILE

First, read the data from CSV file which has hindi language datasets and then concatenate them and form a text, now split the text to individual sentences and do the further step on top of that. The columns are headline, summary and article.

3.2 PREPROCESSING

This is cleaning of data to begin with on datasets that contains information and some content collected from the websites. So we need to do the data preprocessing on this information. For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. One of the major forms of preprocessing is to filter out useless data. In natural language processing, useless words, are referred to as stop words. We apply preprocessing on the article which enables us to remove all the extra data from it which are of no use e.g.: 'में','के','है','से'.etc.

3.3 WORD EMBEDDING

Word embeddings are vector representation of words. These word embeddings will be used to create vectors

for sentences. This is language modeling techniques used in NLP. Where words from vocabulary is mapped to vectors or numbers. Word embeddings are in fact a class of techniques where individual words are represented as real-valued vectors in a predefined vector space. The algorithm use for word embedding is GloVe, The Global Vectors for Word Representation, or GloVe, algorithm is an extension to the word2vec method for efficiently learning word vectors, developed by Pennington, et al. at Stanford. The model is Text rank algorithm for obtaining vector representations for words. This is achieved by mapping words into a meaningful space where the distance between words is related to semantic similarity. Use this basically to find similarity between sentence vectors. Here now we create vectors for our sentences which are nothing but the article to which this is going to be implemented.

3.4 SIMILARITY MATRIX AND GRAPH

First, fetch vectors (each of size 100 elements) for the constituent words in a sentence and then take mean/average of those vectors to arrive at a consolidated vector for the sentence. Based on vector of sentences we prepare a similarity matrix, i.e. find similarities between the sentences, and use the cosine similarity approach for this challenge. Let's create an empty similarity matrix for this task and populate it with cosine similarities of the sentences. Let's first define a zero matrix of dimensions (n * n). Initialize this matrix with cosine similarity scores of the sentences. Use Cosine Similarity to compute the similarity between a pair of sentences and initialize the matrix with cosine similarity scores. Before proceeding further, let's convert the similarity matrix into a graph. By definition graph is a combination of nodes(vertices) and identified pair of nodes(edges, links).The nodes of this graph will represent the sentences and the edges will represent the similarity scores between the sentences. Assuming that your matrix is an numpy array, you can use the method Graph=networkx.from_numpy_matrix('numpy_adj_matrix.npy') to draw the graph. On this graph, I will apply the Page Rank algorithm to arrive at the sentence rankings.

3.5 PAGE RANKING AND TEXT RANK

Page Rank algorithm inspired Text Rank. Page Rank is used primarily for ranking web pages in online search results. In order to rank these webpage, we would have to compute a score called the Page Rank score. This score is the probability of a user visiting that page.

Suppose we have 4 web pages — w1, w2, w3, and w4. These pages contain links pointing to one another. Some pages might have no link – these are called dangling pages.

webpage	links
w1	[w4, w2]
w2	[w3, w1]
w3	[]
w4	[w1]

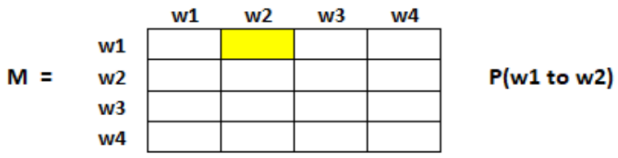
Web page w1 has links directing to w2 and w4
w2 has links for w3 and w1
w4 has links only for the web page w1
w3 has no links and hence it will be called a dangling page
In order to rank these pages, we would have to compute a score called the **PageRank score**. This score is the probability of a user visiting that page.

To capture the probabilities of users navigating from one page to another, we will create a square **matrix M**, having n rows and n columns, where **n** is the number of web pages.

M =

	w1	w2	w3	w4
w1				
w2				
w3				
w4				

Each element of this matrix denotes the probability of a user transitioning from one web page to another. For example, the highlighted cell below contains the probability of transition from w1 to w2.



The initialization of the probabilities is explained in the steps below:

1.Probability of going from page i to j, i.e., M[i][j], is initialized with 1/(number of unique links in web page wi)

2.If there is no link between the page i and j, then the probability will be initialized with **0**

3.If a user has landed on a dangling page, then it is assumed that he is equally likely to transition to any page. Hence, M[i][j] will be initialized with 1/(number of web pages)

Finally, it's time to extract the top N sentences based on their rankings for summary generation. For our, based on similarity we take out top 2 or 3 sentences and summarize them. The basic idea of Text Rank is to provide a score for each sentence in a text, then you can take the top-n sentences and sort them as they appear in the text to build an automatic summary.

First, the words are assigned parts of speech, so that only nouns and adjectives are considered. Then a graph of words is created. The words are the nodes/vertices. Each word is connected to other words that are close to it in the text. In the graph, this is represented by the connections on the graph.

The algorithm is then run on the graph. Each node is given a weight of 1. Then the algorithm goes through the list of nodes and collects the influence of each of its inbound connections. The influence is usually just the value of the connected vertex (initially 1, but it varies) and then summed up to determine the new score for the node. Then these scores are normalized, the highest score becomes 1, and the rest are scaled from 0 to 1 based on that value. Each time through the algorithm gets closer to the actual "value" for each node, and it repeats until the values stop changing.

In post-processing, the algorithm takes the top scored words that have been identified as important and outputs them as key/important words. They can also be combined if they are used together often.

IV.RESULTS

Precision = number of summary sentences extracted matching to human summary/total number of sentences in extracted summary
Precision is also called accuracy of the system, which indicates how good a system is to select sentence as a summary sentence.

Recall = number of summary sentences extracted matching to human summary/total number of sentences in the human summary
The recall represents completeness of a sentence. A recall value of 1 indicates that the system selects all the actual summary sentences in the document. It is also called the sensitivity of the system.

F1 score = 2(precision ∗ recall) / (precision + recall)

	Rouge 1			Rouge 2		
	Precision	Recall	F1 score	Precision	Recall	F1 score
Text Rank Algorithm	0.6854	0.9497	0.7962	0.5724	0.9154	0.7043

TF – IDF Score	0.6330	0.9367	0.7733	0.5570	0.9097	0.6975

V.CONCLUSION

In this paper, we have proposed the idea of creating an automatic extractive summarization system for summarizing the Hindi documents using Text Rank algorithm. Currently, the field of summarization is gaining importance as data on the internet is increasing at a very high rate. Instead of reading the entire document, reading the summary of a document is efficient. But creating a manual summary is a tedious task. Automatic summarization can provide summary at a very fast speed. One potential solution is to summarize a document using either extractive or abstractive methods. The text summarization by extractive is easier to build.

VI.REFERENCES

[1] Jain, A.; Arora, A.; Morato, J.; Yadav, D.; Kumar, K.V. Automatic Text Summarization for Hindi Using Real Coded Genetic Algorithm. Appl. Sci. 2022, 12, 6584. https://doi.org/ 10.3390/app12136584

[2] Dipali Telavane, Apurva Khude, Kartik Lakade, Mohini Chaudhari, Automatic Summarization of Hindi Text Documents Using Supervised Learning Method. International Journal for Research in Engineering Application & Management (IJREAM) ISSN : 2454-9150 Vol-04, Issue-10, Jan 2019.

[3] C. Thaokar and L. Malik, "Test model for summarizing Hindi Text using extraction method", Proceedings of 2013 IEEE Conference on Information and Communication Technologies, 2013.

[4] D. Kaur and R. Kaur, "Automatic Summarization of Text Documents Written in Hindi Language", International Journal of Computer Science and Mobile Computing, vol. 3, no. 10, pp. 320-323, 2014.

[5] Bijal Dalwadi, et.al. "A Review: Text Categorization for Indian Language", 2349-4476, International Journal of Engineering Technology Management and Applied Sciences, March 2015.

[6] V. Dalal and D. Malik, "Automatic Summarization for Hindi Text Documents using Bio-inspired Computing", IJARCCE, vol. 6, no. 4, pp. 682-688, 2017. Available: 10.17148/ijarcce.2017.64130.

[7] S. Saziyabegum and P. S., "Literature Review on Extractive Text Summarization Approaches", International Journal of Computer Applications, vol. 156, no. 12, pp. 28-36, 2016. Available: 10.5120/ijca2016912574.

[8] M. Ali, A. Al-Dahoud and B. Hawashin, "Enhanced Feature-Based Automatic Text Summarization System Using Supervised Technique", INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY, vol. 15, no. 5, pp. 6757-6767, 2016. Available: 10.24297/ijct.v15i5.1630.

[9] Jain, A.; Yadav, D.; Tayal, D.K. NER for Hindi Language Using Association Rules. In Proceedings of the International Confer- ence on Data Mining and Intelligent Computing (ICDMIC), Delhi, India, 5–6 September 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1–5.

[10] S. Zaware, D. Patadiya, A. Gaikwad, S. Gulhane and A. Thakare, "Text Summarization using TF-IDF and Textrank algorithm," *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, 2021, pp. 1399-1407, doi: 10.1109/ICOEI51242.2021.9453071.

[11] P. R. Dedhia, H. P. Pachgade, A. P. Malani, N. Raul and M. Naik, "Study on Abstractive Text Summarization Techniques," 2020 International Conference on Emerging Trends in Information

Technology and Engineering (ic-ETITE), 2020, pp. 1-8, doi:10.1109/ic-ETITE47903.2020.087.

[12] Awasthi, K. Gupta, P. S. Bhogal, S. S. Anand and P. K. Soni, "Natural Language Processing (NLP) based Text Summarization - A Survey," *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, Coimbatore, India, 2021, pp. 1310-1317, doi: 10.1109/ICICT50816.2021.9358703.

[13] Aniket Suryavanshi, Bhavika Gujare, Allan Mascarenhas and Bhanu Tekwani. Hindi Multi-document Text Summarization using Text Rank Algorithm. *International Journal of Computer Applications* 174(29):27-29, April 2021.

[14] Mehta, Harsh & Bharti, Drsantosh & Doshi, Nishant. (2022). Automatic Text summarization in Gujarati language. 1-6. 10.1109/iSSSC56467.2022.10051338.

[15] Pokharkar, Akshay and Dhumal, Pratik and Singh, Aman and Hadawale, Hrithik, Text Summarizer Using NLP (May 1, 2022).

Cite this article as :

Jani Patel, Dr. Narendrasinh Chauhan, Dr. Krunal Patel, "Text Summarization Using Natural Language Processing", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 9, Issue 3, pp.16-22, May-June-2023. Available at doi : https://doi.org/10.32628/CSEIT2390298
Journal URL : https://ijsrcseit.com/CSEIT2390298