Hindi Text Summarizer

[1]Saurav Sanap, [2]Ruturaj Sawant,[3]Shazeb Sayyed, [4]Shoaib Shaikh, [5]Devarshi Wadadkar,

Department of Computer Engineering, VIT, Pune.

Abstract: Text summarization involves reducing the text that is built from one or more texts so that particularly important information in the text is not lost. Text summarization compresses raw text to display only what is important and necessary for the user. When summarizing an article, decisions are often made quickly in order to capture the essence of the paper. It focuses on mining methods and applications in Python. The extraction method defines a set of sentences supported by the subject method. We also consider removing Hindi keywords and extracting nouns in sentences before selecting nouns. Disabling word removal removes empty keywords from input documents, and word search allows words to be grouped by matching numbering system terms. The system is based on a sentence scoring algorithm, supported by the emergence of subject-specific word numbering systems. The recommendations with the highest score will be added to the summary. The generated resumes are then processed to help remove irrelevant phrases from the pre-selected resume suggestions, making the suggestions look more like human-generated resumes.

Keywords: generalization of Hindi text, extractive access, tokenization, Textrank, Networkx, etc.

Introduction

The overview can serve as a quick guide to the information you need after reading the summary the user decides whether to read the entire document summaries save time daily life summaries include headlines biographies digests story summaries highlights event summaries abstracts technical paper summaries bulletin boards weather forecasts and movie trailers a lot of research has been done in english less in indian our goal is to develop an automatic text summary for hindi texts it is one of several languages spoken in different parts of the country although hindi is native to northern india it is spoken studied taught and understood throughout india hindi is written in the devanagari script hindi sentence syntax follows a subject-object-verb structure there are two types of reference systems a extracted totals and b abstract totals the extract aggregation feature selects important sentences from source documents based on their statistical linguistic or hybrid characteristics and creates a summary in short form summary text summarization is based on the application of natural language understanding people create resumes abstractly an abstract summarization method is to understand the initial text and reproduce it in fewer words hindi text summarization system is an extractive summarization system used to summarize hindi texts by retaining relevant sentences supported by the texts statistical and linguistic features hindi text summarization service includes her three main phases 1 preprocessing 2 processing and 3 making summaries preprocessing is an organized representation of the original hindi text during processing each sentence is assigned a weight using a feature weight formula during the summarization phase a biologically-inspired algorithm adjusts the weights to find the optimal solution the top sentences are then selected in the correct order for the final summary according to the compression ratio this paper focuses on the preprocessing phase of a hindi text summarization system the preprocessing steps include hindi text segmentation tokenization identifying word boundaries hindi stop word removal and word root extraction hindi stemmer is used to find the root of inflection stemming can be used to improve offer search efficiency the input to the preprocessing step is hindi text and the output is structured text you can also remove duplicate statements before processing steps.

I. **Literature Review**

Most of the previous work on extraction generalization mainly used two important steps.

I. Sentences are ranked based on a score calculated by combining several characteristics, such as term frequency (TF), location information, and key phrases.

II. Choose some of the best offers to build your resume. The first work on automatic text summarization was done (Luhn, 1958). He created summaries using the number of occurrences of a word and the frequency of a phrase in a document as features. Assume that the most frequent words represent the main topic of the document. Although several novel feature-based generalization methods have been developed in subsequent studies, the work presented by the authors is still used as a basis for extraction-based generalization. P.B. Baxendale (Baxendale, 1958) proposed a new function to represent the position of a sentence or the position of a sentence in an input document. Sentences located at the beginning or end of a document are analyzed to be more important than other sentences in the document. H.P. Edmundson (1969) [2] proposed a new structure for text summarization. They proposed two new features. Keywords that appear in the most visible words in a document, such as first, last, briefly, last, and so on. Second, if the sentence has a title, it is a simple feature title or title that gives extra weight to the sentence. Later (Kupiec, Pedersen, & Chen, 1995) proposed a machine-learning approach to text generalization. They described a new generalization method using a Naive Bayes classifier. The classification function classifies each sentence according to whether it is worth extracting or not. (Conroy & O`leary, 2001) [3] proposed two sentence extraction methods, QR (QR Matrix Decomposition) and HMM (Hidden Markov Models). In the QR method, the importance of each sentence was measured and the most important sentences were added to the summary. When this was done, the relative importance of the remaining sentences changed because some of them were redundant. They repeated this process until they captured enough important ideas. Another method was HMM, a sequential automatic text summarization model that estimates how likely each sentence should be in a resume. There are only three features used in HMM: position within a sentence, total number of terms within a sentence, and term similarity within a sentence within this document. In the end, an evaluation was performed by comparing the HMM-generated summary with the human-generated summary. (Erkan & Radev, 2004) [4] proposed a graph-based probabilistic method to calculate the relative importance of text units. They used the

LexRank approach to calculate sentence importance based on the concept of eigenvector centrality in a graphical representation of a sentence.

III. Overview

Here is an overview of the extraction-based generalization method. There are various extraction methods to help you find relevant sentences to add to your resume. These methods can be divided into statistical, linguistic, and hybrid approaches.

a) Statistical methods

The generalization of text supported by this approach is based on the distribution of specific features and avoids understanding the entire document. It uses classification and information retrieval methods. Classification methods classify sentences that will be part of a summary based on knowledge training. Information retrieval techniques use the position, sentence length, or occurrence of words in a document. This method extracts sentences that appear in the source text without considering the meaning of the words.

2. Linguistic Method

In this, the method must remember and know deeply the linguistic knowledge, so that the pc is going to be ready to analyze the sentences and then decide which sentence to be selected. It identifies term relationships within the document through part-of-speech tagging, grammar analysis, thesaurus usage, and extracting meaningful sentences. Parameters are often cue words, Title features, or nouns and verbs within the sentences. Statistical approaches could also be efficient in computation but Linguistic approaches check out term semantics, which can yield better summary results. In practice, in the context of linguistic approaches, we have also adopted simple statistical computation (term-frequency- inverse-document frequency (TF-IDF) weighting scheme) to filter terms. However, there are relatively few studies in the literature discussing linguistic approaches that use term weighting systems derived from appropriate mathematical (probabilistic) models to give more meaning to weight definitions. We also suggested some use cases in future work.

3. Hybrid method

Better use of the previous two methods for a meaningful and concise summary.

IV. Methodology

a) Components:

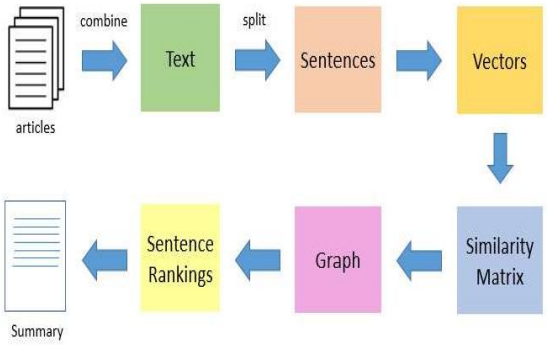
In our summarizer, there are 3 main codes. One of them is for tokenization, which preprocesses documents to apply our algorithms. Second, to implement text ranking in preprocessed documents. And the third is to check the accuracy of the results for your resume. The main libraries imported into the code are:

1. Networkx: A Python library for training graphs and networks.
2. Numpy: A Python library for working with arrays. There are also functions for working with linear algebra, Fourier transforms, and matrices.
3. Math: Provides access to common math functions and constants in Python that you can use in your code for more complex math calculations.
4. Sklearn: Supports Python numerical and scientific libraries such as NumPy and SciPy.

b) Algorithm:

We use a text ranking algorithm for generalization. Textrank is an extract and unsupervised text summarization method. Let's look at the flow of the Textrank algorithm as follows.

• The first step is to merge all the text in the text document.
• Then split the sentences into documents.
• Then find vectors for all sentences.



• Matrix compilation using sentence similarity
• The similarity matrix calculates the rank of the sentence by transforming it into a graph with the sentence as the vertex and the similarity score as the edge.
• Select a certain number of proposals based on evaluation at the end.

c) Code flow:

Our code is divided into three parts.

1. Input tokenization (preprocessing):

This part of the code reads the user's input into a text file, preprocesses it appropriately, and uses it to generate a summary. This is the first and most important task to complete. First, clean up the text by removing punctuation marks. Tokenization includes removing words that contain spaces, splitting hyphenated words, removing suffixes to determine the root of words, creating a

dictionary of stemmed words, and stopping words. Use these tokens to create sentences, write them to a file, and save them to the system to generate summaries.

2. Summary:

Here, first, get the tokenized file. It then applies a text ranking function to the imported files. The text ranking uses a CountVectoriser from the tokenized file to generate the Bow_matrix. The bow_matrix transform tfidf is then transposed to create a similarity graph. We then apply a page rank to this similarity plot. So, we generate a score for all proposals, then sort them in descending order and print 30% of the total number of proposals accordingly.

3. Key points:

This part of the code is used to test the accuracy of the output against a human summary. First, check the length of the machine output and the reference output. A length of 0 skips this file. It then lists the sentences in both files and then checks for correctness with the sentences common in both files. Then check the full results of F1.

V. Results and Conclusions

Our proposed system can be easily generalized to any number of input files. Also, this system allows you to choose the size of your resume. I've kept it at 30% for now, but you can easily change it to whatever you like. It also shows that generalized results can be similar to human summed results. During testing, we were able to successfully achieve over 90% accuracy with respect to human results.

VI. Future Scope

Expanding the system to more languages is part of future work the generated summaries are available in multiple languages and the system can work in multiple languages to generate the summaries the system can be integrated with language translation systems to generate summaries in the language of the users choice regardless of the original language of the document this may include crawling the web for summaries of the many documents available on the internet. Also creating a GUI application where the user can load a text document and the application will try to summarize it.

VII. References

1. Baxendale, P. B. (1958). Machine-made index for technical literature— an experiment. IBM Journal of Research and Development, 2(4), 354–361
2. Edmundson, H. P. (1969). New methods in automatic extracting. Journal of the ACM (JACM), 16(2), 264–285. Erkan, G., & Radev, D. R. (2004).
3. Conroy, J. M., & 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 (pp. 406-407)
4. Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research, 22, 457–479.
5. 5.https://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html
6. https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/
7. https://docs.python.org/3/library/