AUTOMATIC TEXT SUMMARIZATION

Suchita, Mayuri, kamini, Prof. Suvarna.P.Kale

1. *Suchita N Girase, Information Technology, Sandip Polytechnic, Maharashtra, India*
2. *Mayuri R Sable, Information Technology, Sandip Polytechnic, Maharashtra, India*
3. *Kamini M achari, Information Technology, Sandip Polytechnic, Maharashtra, India*
4. *Prof. Suvarna.P.Kale, , Information Technology, Sandip Polytechnic, Maharashtra, India*

ABSTRACT

Extractive Text Summarization is a Natural Language Processing problem of forming a summary by using the most important sentences of an article and has been addressed in many different ways. In this research, a model is developed that uses metrics for giving importance to sentences, and applies it to Machine Learning and Deep Learning Models. The main objective of the model is to learn to classify which sentences belong and which do not in a summary. Using the results obtained the optimal approach, along with other approaches that work particularly well, to do extractive text summarization, are found. In addition, the analysis of why these models perform better than the rest is done. Finally, the importance of each metric in forming a summary is discovered based on several evaluation measures. Text summarization has a lot of research devoted to it, which has resulted in development of various techniques to do the same. The process of text summarization is broadly divided into two types – extractive and abstractive. To develop a model that does extractive text summarization along with finding and evaluating which parameters and machine learning and deep learning algorithms optimize the working of the model.

Keyword *Machine learning , Deep learning , Natural Language Processing(NLP) etc*

1. CHAPTER: 1

INTRODUCTION

Automatic text summarization is a technique that condenses the longer original text into a shorter format, giving a summary of the original topic. The summary is formed on the most important points and the main idea of the original text. As a result, the reader is presented with a compact and focused view of the original text.

1.1 Purpose and application

Due to the exponential increase in the availability of data in the recent years – amounting to almost a million terabytes - the need to summarize is imperative. With the ever increasing pace of life, the public no longer has the time to read long newspaper and magazine articles; a tool to develop an effective summary is much needed. Added to that, manpower of businesses is being continuously overwhelmed by the magnitude of data available; the need for a tool to remove the redundancies and find the main point is, again, much needed. Broadly, the application of summarization is valuable in domains like finance and retail, search engines, business analysis, news because it helps save time and improves productivity.

1.2 Problem statement

To develop a model that does extractive text summarization along with finding and evaluating which parameters and machine learning and deep learning algorithms optimize the working of the model.

2. CHAPTER NO: 2

Literature View

The process of extractive text summarization has seen several approaches in the past. The earliest techniques have relied on calculating the weight of a sentence using various factors. Each of the factors are in turn given their own separate weight to highlight their importance. But not much research has been focused on finding the importance of each factor. Further techniques have used fuzzy Logic, clustering, machine learning, deep learning and so on.

2.1 Text processing techniques used in extraction

Factor	**Description**
Term Frequency	This is the most widely and common factor used in summarization. This relies on counting of the number of times a word (except stop words) is repeated. Consequently, it helps assign weight to sentences accordingly. The TF IDF method is the greatest user of this factor.
Sentence Placement	This factor takes into account the the position of sentences in the summary. Usually, the starting and ending sentences contain important information regarding the content and context of the text, and this factor leverages this property.
Keyword or Key Phrase	The keyword or the cue phrase feature looks for words or phrases and classifies them into the positive or negative categories. The positive category words and phrases include, “in conclusion” , “therefore”, “this describes”, among others.
Proper Noun	This factor is important in documents in which proper nouns, like, person names, company names, country names etc. form a vital part of the summary.
Sentence Length	Depending on the length of the desired summary, this factor works by filtering out sentences that are above and below the set threshold.
Title of Document and Similarity Measure	In this, the document is checked to find words and phrases that match or are similar to the heading. It uses linguistic features to be able to do this. The words and phrases found have a greater chance of occurring in the summary.
Biased words	For this, a glossary of frequent words (domain specific) is made. Then, the text is searched and any

	such words are assigned a greater chance of affecting the summary.
Font Based	The font based factor is based upon the idea of finding words and phrases in bold, italics, underlined, or in a different font colour and size compared to the bulk of the document. These words have a greater chance of influencing the summary.
Sentence to sentence cohesion	For this factor, the sentences are evaluated against each other using the likeness factor. The likeness factor is calculated using NLP and linguistic techniques.
Non-essential words	This factor forms a negative filter. Words like, “Furthermore”, “Moreover” rarely include the main point and context of the passage and only provide additional information. Hence, it helps filter out the sentences accompanying these words from the summary.

2.2 Existing methods and techniques

Text summarization has a lot of research devoted to it, which has resulted in development of various techniques to do the same. The process of text summarization is broadly divided into two types – extractive and abstractive.

Here, the review will be based on the extractive side of the process. Some of the most notable mentions include, Term Frequency-Inverse document frequency method, LSA based method, Cluster based method, Summarization using neural networks and fuzzy logic, machine learning approach, and graph based approach. While the first method is built upon the idea of term frequencies to determine prospective words and phrase to be included in the summary, fuzzy logic does the same by using parameters of similarity to title, key-words and phrases and sentence length on the given text . Machine learning approaches focus on classifying prospective summary sentences, using different classifiers ,like Naive Bayes classifier , while deep learning is used to train neural networks for finding out the same. Cluster Method goes a step further in producing summary for multi-documents on different themes . Finally a graph based method, after employing per-processing, pruning, stop word removal, gives a tree with sentences attached as nodes, and, through the weights of the edges, shows what sentences are more likely to be included in the summary. A brief overview of some of the previously mentioned techniques is provided below:

2.2.1 Term Frequency-Inverse document frequency method

This method applies to – both – single documents and multiple documents texts. It uses the following formulas:

1) Term Frequency: TF (t) = Number of times term t appears in the document Total number of terms in documents

2) Inverse Document Frequency: IDF (t) = Ln (Number of documents/ No of documents with term t)

The terms with the score above the set threshold are pruned to remove any unnecessary part and arranged in the order of importance. For multi-documents, key terms and sentences are extracted using TF-IDF and single document summarization is used on the multi-documents to form the final summary.

2.2.2 Cluster based method

Cluster based method is useful, especially, for producing summaries for multi-documents based on different themes. The likelihood of a sentence being included in a summary is based on the following formula and using k-means clustering algorithm.

$$S(i) = W1 * C(i) + W2 * P(i) + W3 * F(i)$$

where, C is the parameter that tells the similarity of the sentence to the theme of the document; P is the parameter that tells the location of the sentence in the document; and F is the similarity to the first sentence of the document. W1, W2 and W3 are the weights associated with each factor. This technique clusters like information in different clusters. Since the presence of multiple documents can have a lot of repeated or redundant information, this technique has the deal with challenge of reducing redundancy, even from sentences that are important. Some variations of this technique use TDF-IDF to rank the different documents according to their TD-IDF score, i.e. documents with a higher score are more likely to contribute to the summary.

2.3 Motivation for proposed model

As the different techniques came out, each had its advantages and its disadvantages. Some performed better on particular domains, for example automobile articles, while others had a better overall performance. Keeping this in mind it was decided to study the effectiveness of different machine learning and deep learning models along with finding out which metrics are more influential in forming a summary. By finding what produces the most coherent summary through the research, it is aimed to aid future endeavors in the process of summarization and, subsequently, building a more accurate model. To train the model, a manually compiled dataset from different sources and domains was used. The decision to use of varied domains was an attempt to avoid bias towards particular topics and give the model equal exposure to different domains. The dataset comprises of individual sentences from each article and whether or not it was included in the summary. Consequently the training set was used on k-NN, Naive Bayes, kernel SVM, decision tree, random forest and ANN.

3. CHAPTER : 3
SYSTEM REQUIREMENT SPECIFICATION

3.1 Functional requirements

3.1.1 User-Interface

User will give a text file needed to be summarized, and the summary will be displayed

in a new page

3.1.2 Training data

The model has to be trained on a set of documents along with their corresponding

summaries. A summary will include all the sentences that give the main idea of the text.

3.2 Non functional requirements

3.2.1 Usability

Application must be simple and easy to use.

Application must be intuitive and simple in the way it displays all relevant data and

relationships.

3.2.2 Reliability

The application must inform the user in situations of a crash or error

Should be up for working as and when required.

3.3 Hardware requirements:

System runs on any regular PC/Laptop

If the user intends on training the model on a different dataset, then a machine with 4

GB or more

RAM and processor of 2.7 GHz or more is required.

3.4 Software requirement

Any modern web browser

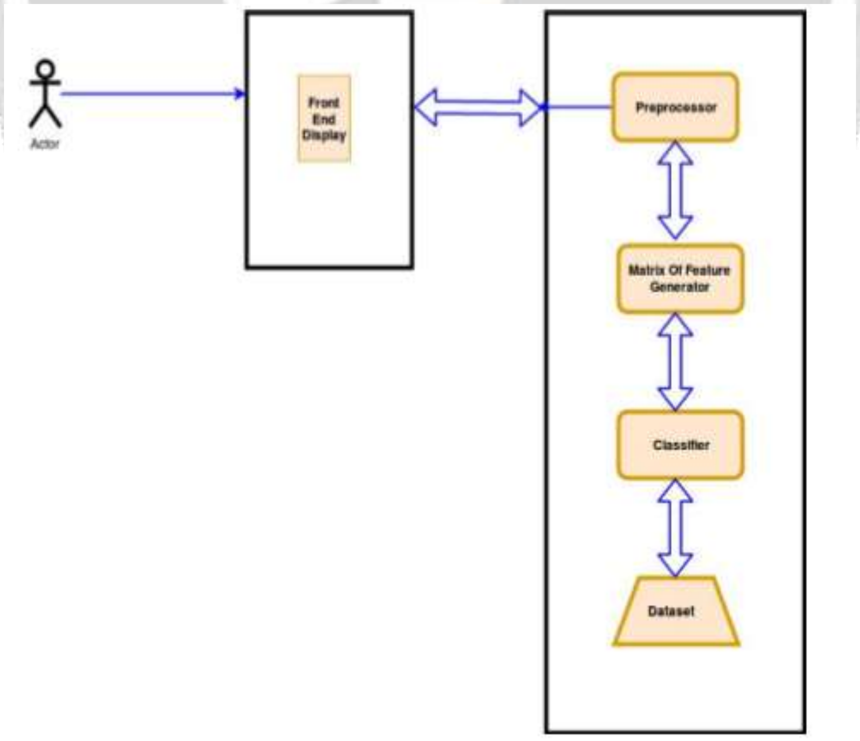
Language Used : Python environment

Libraries/Toolkits :NLTK , Sci-kit Learn , Keras , Pandas , Numpy

Webframe Used : Flask

4. CHAPTER NO: 4

4.1 SYSTEM ARCHITECHTURE



The model is pre-trained on the given dataset. When the user requests a summary to be generated by feeding in a document, the document is passed onto the server side where the model does preprocessing of the given text and applied the trained ANN classifier. It then sends the result back to the User. The browser block is the front end for the user. The user acceses the system wherein he/she is asked to provide the text for summarizing. After entering the text the text is feeded to the server side where the summarization is done. The server side consists of the model which takes in the text. The model first preprocesses the text to get it in the form that could be fed into the classifier. Using this preprocessed text a matrix of features is developed which is fed into the machine learning model. The classifer/model trains in itself on the dataset available meanwhile. When the matrix of features and classifier are both ready, the data is fed into the classifier. The classifer makes the prediction for each sentence. Each of the sentence classified as a summary sentence by the classifer is then propagated to the client side, where these sentences are displayed in the order of their appearance in the text.

5. CHAPTER 5:

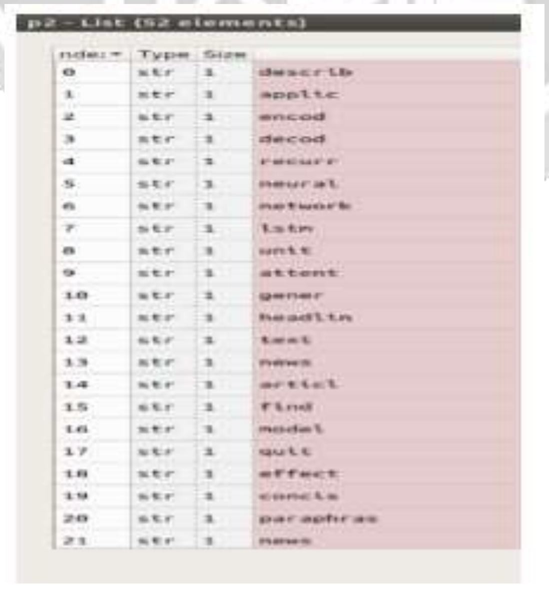
IMPLEMENTATION DETAILS

IMPLIMENTATION

The method employed included 3 phases: preprocessing, formation of matrix of features, and application of different models Before this, it was decided to include word frequency, sentence placement, presence of important words, like, 'therefore', 'thus', 'hence' etc., presence of non-essential words and phrases, like, 'moreover', 'furthermore', 'in addition', etc., sentence length, and presence of proper nouns, as metrics.

5.1 Preprocessing

The dataset comprising of sentences and whether they were selected or not was imported. The text was subjected to multiple preprocessing tasks. The preprocessor first broke the text into individual words, and removed all forms of punctuation, and white spaces, and turned all words into lower case. The transition of lower case was done to address a future process, described later, in the method. Next, it removed all stop words, like, „a‟, „it‟, „so‟, „they‟, etc. as these are not needed in the process of text summarization. Finally, it stemmed all the remaining words. The process of stemming and converting to lower case was done to reduce the number of unique words in order to conserve memory and reduce processing time. For example stemming coverts the words 'lovingly' and 'loving' which have the same meaning in terms of a summary. It converts them into the root „love‟. The following is an example of a sentence after word tokenization and preprocessing.

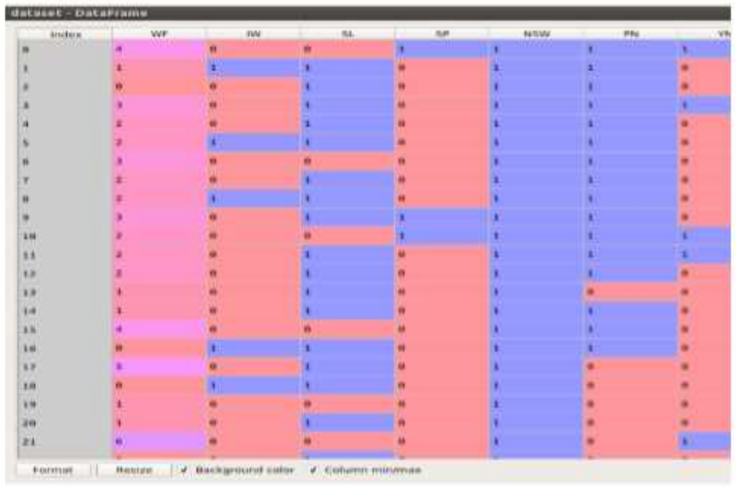


p2 - List (52 elements)

ndex	Type	Size	
0	str	1	describ
1	str	1	appltc
2	str	1	encod
3	str	1	decod
4	str	1	recurr
5	str	1	neural
6	str	1	network
7	str	1	lstm
8	str	1	unit
9	str	1	attent
10	str	1	gener
11	str	1	headlin
12	str	1	text
13	str	1	news
14	str	1	articl
15	str	1	find
16	str	1	model
17	str	1	quit
18	str	1	effect
19	str	1	concis
20	str	1	paraphras
21	str	1	news

5.2 Matrix of features

The matrix of features consists of a row for each sentence and 6 columns, each corresponding to one metric. For each article, the five most frequent words were computed and their use will be mentioned later. The matrix of features is filled with respect to each row, i.e. each sentence. Each preprocessed sentence is taken in and its records filled in. The word frequency metric corresponds to the number of frequent words in a particular sentence. If an important word is present, then the record is filled as 1, else 0 and similarly for non-essential words. If sentence length is less than 15 then record is filled as 1, else 0. If a proper noun is present then the model fills 1, else 0. Lastly, if the sentence is the first or last sentence in the article then the model fills 1, else 0; it is common to find the first or last sentences playing a vital part in forming a summary. The following figure is an example of a sample matrix of feature representation.



dataset - DataFrame

Index	WF	IW	SL	SP	NSW	PN	Y
0	4	0	0	1	1	1	1
1	1	1	1	0	1	1	0
2	0	0	1	0	1	1	0
3	3	0	1	0	1	1	1
4	2	0	1	0	1	1	0
5	2	1	1	0	1	1	0
6	3	0	0	0	1	1	0
7	2	0	1	0	1	1	0
8	2	1	1	0	1	1	0
9	3	0	1	1	1	1	0
10	2	0	0	1	1	1	1
11	2	0	1	0	1	1	1
12	2	0	1	0	1	1	0
13	1	0	1	0	1	0	0
14	1	0	1	0	1	1	0
15	4	0	0	0	1	1	0
16	0	1	1	0	1	1	0
17	5	0	1	0	1	0	0
18	0	1	1	0	1	0	0
19	1	0	0	0	1	0	0
20	1	0	1	0	1	0	0
21	6	0	0	0	1	0	1

5.3 Application of different models

Each model used below was fed the matrix of features and predicts whether the sentence will be a summary or not, i.e. 'Yes' or 'No'

5.4 Methods:

5.4.1 Naive Bayes classifier:

Naive Bayes assumes that the six features are statistically independent of each other and generates a probability of it being included in the summary. Sentences with a probability greater than 0.6 were chosen as part of the summary. The threshold of 0.6 was chosen after many trial and errors on training and test set.

5.4.2 Decision tree algorithm:

The ID3 algorithm of decision tree, with 'Information Gain' function as the feature selection parameter was used to classify the sentences.

5.4.3 Random forest:

Random forest creates multiple decision trees and merges them together in order to get a more accurate prediction compared to decision trees. The random forest classifier from the “Keras” library in python was implemented and the number of trees set to 10, again owing to trial and error to get the best possible accuracy for the dataset.

5.4.4 K-Nearest neighbors:

A k-NN predicts the class of a tuple based on the class of the k nearest neighbors. K was chosen as 5, which is usually a common value used, and the Euclidean distance measure was used. The majority nearest tuple decided the class of the new sentence.

5.4.5 Kernel Support Vector Machine:

The fact that Kernel SVM's computations are not significantly slowed by non-linearly separable data and it performs much better under these circumstances as it does not have to scale features to a higher dimension, made us choose this for testing. For this approach the RBF kernel function was used. f(x, y) = exp(- ||x - y|| / sigma) It finds how similar a sentence is to an ideal summary sentence and as a result classifies it into one of the two categories

5.4.6 Artificial Neural Network:

A simple, single layer ANN for testing purposes was implemented. The binary cross entropy function was used as the loss function as the output was binary. The input and the hidden layer used the rectifier activation function while the sigmoid function was used on the output layer. The sigmoid function gave us the probability of a sentence being part of a summary. Once again 0.6 was used as the threshold owing to the reason mentioned earlier.

6. CHAPTER 6:
RESULT

6.1 Metric and algorithm wise results

For the evaluation the following metrics to assess the different methods were used: Accuracy = TP/TP + TN Precision = TP/ TP + FP Recall = TP/TP + FN F1 measure = 2*Precision*Recall/ Precision + Recall

Following is the result table obtained for the different approaches:

Algorithm	**Accuracy**	**Precision**	**Recall**	**F-measure**
KNN	0.667	0.750	0.689	0.718
Naive Bayes	0.762	0.858	0.877	0.868
Random Forest	0.776	0.835	0.875	0.854
Decision Tree	0.764	0.832	0.856	0.843
Kernel SVM	0.761	0.832	0.856	0.843
ANN	0.803	0.880	0.864	0.872

Table 7.1

The ANN led to the highest accuracy while the machine learning model of random forest followed with the second highest accuracy. Although all the algorithms hover around the same accuracy, k-NN seems far off. The reason for KSVM performing better than SVM is attributed to the fact that KSVM is better suited to handle non-linear data which is exactly what the dataset used contains. Random forest performing better than the regular decision tree is due to the fact that it takes into account the performance of multiple trees and averages out the result which makes the accuracy better off. The ANN unlike other models learns from the each epoch and uses backpropogation to adjust its weights to get the least possible error. Hence, it performs the best.

The following table shows the results for each feature:

Metric/Feature	**Accuracy**
Word Frequency	0.762
Important Words	0.761
Sentence Length	0.664
Sentence Placement	0.757
Non-essential words	0.751
Proper Nouns	0.703

Table 7.2

Word frequency and important words were the most prominent metrics. Given that phrases and words like „thus", „therefore", „to summarize" form important part of summaries, and sentences with most frequent non-stop words are also likely to form a summary this comes as no surprise. One anomaly was the sentence placement value equal to 0.242 in the KNN algorithm. These anomalies are attributed to the accuracy of the models and the fact the there are differences between new real world articles and that from those used in the dataset.

6.2 Errors

For approaches involving the Naïve Bayes and ANN, where the probabilities of each sentence is calculated and those above the threshold are chosen, the approach shows a limitation. In rare circumstances, the approach has classified no sentences above the threshold or as „Yes". To handle this, the two sentences with the maximum probability as the summary were chosed.

7. CONCLUSIONS :

In this project the model formed extractive text summaries using k-NN, Naive Bayes, kernel SVM, decision tree, random forest, and ANN. Along with this it was found which metrics are most influential in forming a summary. The best performing models turned out to be random forest while the best metrics turned out to be important words and word frequency. The results would benefit the future development of summarizers in terms of the selection of model, metrics, and thus increasing accuracy. Although the models that were tested cover a respectable size of implementations, the list is not exhaustive. There exist many more models which could be researched upon in the future.

8. ACKNOWLEDGEMENT

With deep sense of gratitude we would like to thanks all the people who have lit our path with their kind guidance. We are very grateful to these intellectuals who did their best to help during our project work.

It is our proud privilege to express deep sense of gratitude to, **Prof. PRASHANT PATIL, Principal** of Sandip Polytechnic, Nashik, for his comments and kind permission to complete this project. We remain indebted to **Prof. V. J. CHAUDHARI, H.O.D INFORMATION TECHNOLOGY Department** for their timely suggestion and valuable guidance.

The special gratitude goes my guide **Prof. S.P.KALE** and staff members, technical staff members of INFORMATION TECHNOLOGY Department for their expensive, excellent and precious guidance in completion of this work. We thank to all the colleagues for their appreciable help for our working project.

With various industry owners or lab technicians to help, it has been our endeavor to throughout our work to cover the entire project work.

We are also thankful to our parents who providing their wishful support for our project completion successfully.

And lastly we thanks to our all friends and the people who are directly or indirectly related to our project work.

9. REFERENCES

[1] Klaus Zechner, "A Literature Survey on Information Extraction and Text Summarization", Computational Linguistics Program, Carnegie Mellon University, April 14, 1997.

[2]. Mani, M. T. Maybury, "Advances in Automatic Text Summarization" in , The MIT Press, 1999.

[3] Khosrow Kaikhah. "Automatic Text Summarization with Neural Networks". Intelligent Systems, 2004. Proceedings. 2004 2nd International IEEE Conference, Volume: 1. July 2004.

[4] P.-Y. Zhang, L. Cun-He, "Automatic text summarization based on sentences clustering and extraction", Computer Science and Information Technology 2009. ICCSIT 2009. 2nd IEEE International Conference, 2009.

[5] Josef Steinberger and Karel Jeˇzek. "Evaluation Measures For Text Summarization", Computing and Informatic"s, Vol. 28, 2009, 1001–1026, V 2009-Mar-2.