COMPARISON OF EXTRACTIVE TEXT SUMMARIZATION METHODS

[1]SATISH KUMBHAR, [2]ANSH BORDIA, [3]SANKET KAPSE, [4]FALGOUN PAATIL

[1,2,3,4] College of Engineering, Pune, Wellesley Road, Shivajinagar, Pune, Maharashtra 411005
E-mail: [1] ssk.comp@coep.ac.in, [2] bordiaa14.it@coep.ac.in

Abstract - Extractive Text Summarization is a NLP problem of forming a summary by using the most important sentences of an article and has been addressed in many different ways. In our research, we develop a model that uses metrics for giving importance to sentences, and apply it to Machine Learning and Deep Learning Models. Using the results obtained we find the optimal approach, along with other approaches that work particularly well, to do extractive text summarization. In addition, we analyze why these models perform better than the rest. Finally, we also find the importance of each metric in forming a summary based on several evaluation measures.

Keywords - Natural Language Processing, Extractive Text Summarization, Deep Learning.

I. INTRODUCTION

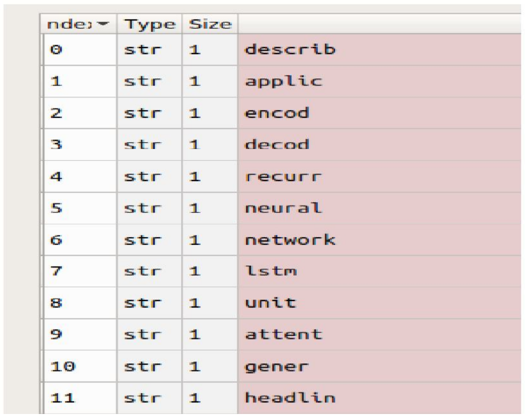
With the exponential increase in the availability of data and the ever increasing pace of life, the need to summarize has become imperative. Added to that, the task of summarization is needed in many and diverse fields including finance, search engines, news, research papers, etc. As a result, there has been a lot of work done in the field of extractive summarization. Past work include assigning importance to sentences by assigning weights to metrics such as word frequency, sentence placement, biased words, etc. Further, work was done using approaches of Fuzzy Logic, LSA, Cluster based method, Deep Learning Neural Networks, and Machine Learning Models including Naive Bayes, Decision Tree etc. 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 we 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 our research, we hope to aid future endeavors in the process of summarization and, subsequently, building a more accurate model. To train our model, we manually compiled a dataset from different sources and domains. 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 an ANN.

II. METHOD

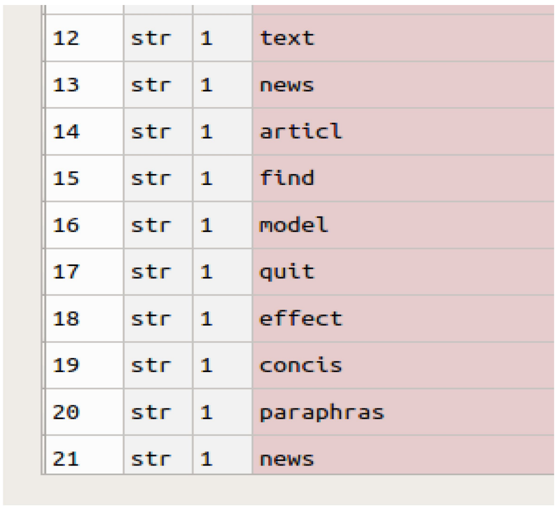
The method we employed included 3 phases: Preprocessing, Formation of Matrix of Features, and Application of different Models

Before this, we 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.

Preprocessing
We imported the dataset comprising of sentences and whether they were selected or not. The text was subjected to multiple preprocessing tasks. We 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 our method. Next, we removed all stop words, like, ‘a’, ‘it’, ‘so’, ‘they’, etc. as these are not needed in the process of text summarization. Finally, we 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



nde:	Type	Size	
0	str	1	describ
1	str	1	applic
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



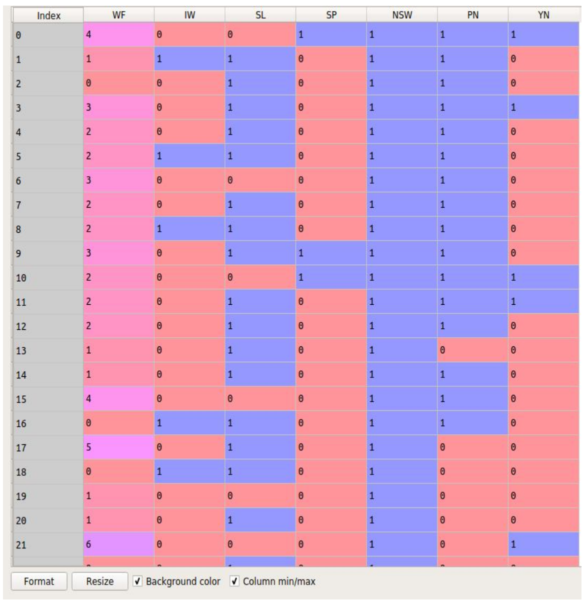
12	str	1	text
13	str	1	news
14	str	1	articl
15	str	1	find
16	str	1	model
17	str	1	quit
18	str	1	effect
19	str	1	concis
20	str	1	paraphras
21	str	1	news

Figure 1: Delay and Area evaluation of an XOR gate.

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 we fill 1, else 0. Lastly, if the sentence is the first or last sentence in the article then we fill 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



Index	WF	IW	SL	SP	NSW	PN	YN
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

Format
Resize
✓ Background color
✓ Column min/max

Figure 2

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'

Methods:

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

$$P(s \in S \mid F_1, F_2, ...F_k) = \frac{P(F_1, F_2, ...F_k \mid s \in S) P(s \in S)}{P(F_1, F_2, ...F_k)}$$

$$P(s \in S \mid F_1, F_2, ...F_k) = \frac{\prod k_j = 1 P(F_j \mid s \in S) P(s \in S)}{\prod k_j = 1 P(F_j)}$$

$P(s \in S)$ is a constant and $P(F_j \mid s \in S)$ and $P(F_j)$ can be estimated directly from the training set by counting occurrences. Sentences with a probability greater than 0.6 were chosen as part of a summary. The threshold of 0.6 was chosen after many trial and errors on training and test set.

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.

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. We used the random forest classifier from the ‘Keras’ library in python and set the number of trees as 10, again owing to trial and error to get the best possible accuracy for our dataset.

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) 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 we used the RBF kernel function.

k(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 our two categories

6) Artificial Neural Networks:

We used a simple, single layer ANN for our testing purposes. The Binary cross entropy function was used as the loss function as our 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.

RESULTS

We used the following metrics to assess the different methods

1. Accuracy = TP/TP + TN
2. Precision = TP/ TP + FP
3. Recall = TP/TP + FN
4. F measure = 2*Precision*Recall/ Precision + Recall
5. Following is the result table obtained for the different approaches

Table 1

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

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.

Table 2

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

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.

ERRORS

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

CONCLUSION

In our research we formed extractive text summaries using k-NN, Naive Bayes, Kernel SVM, Decision Tree, Random Forest, and ANN. Along with this we 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.
We hope that our results benefit the future development of summarizers in terms of the selection of model, metrics, and thus increasing accuracy. Although the models that we have 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.

REFERENCES

[1] Julian Kuplec, Jan Pedersen and Francine Chen. "A Trainable Document Summarizer".

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

[3] Daniel Chai. "A Trainable Web Page Document Summarizer". CS 224N Final

[4] M Indu and Kavitha KV., "Review on text summarization evaluation methods". Research Advances in Integrated Navigation Systems(RAINS), International Conference. 6-7 May 2016.

[5] Deepali K.Gaikwad and C.Namrata Mahender.2016."A Review Paper on Text Summarization" .International Journal of Advanced Research in Computer and Communication Engineering.(Mar.2016).Vol.5, Issue 3.

[6] A.R. Kulkarni, S.S. Apte, "A Domain-Specific Automatic Text Summarization Using Fuzzy Logic", International Journal of Computer Engineering and Technology (IJCET), vol. 4, no. 4, 2013, ISSN 0976-6367.

[7] 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.

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

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

[10] Vishal Gupta and Gurpreet Singh Lehal “A Survey of Text Summarization E xtractive Techniques”, Journal of Emerging Technologies In Web Intelligence, VOL. 2, NO. 3, AUGUST 2010

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