Text Highlighting – A Machine Learning Approach

Varun Ranganathan[1], Ashwini Raja[2], Avanthikaa Ravichandran[3], Nethra Viswanathan[4]

[1,2,3,4] *Student, Department of Computer Science, SSN College of Engineering, Tamil Nadu, India*

Abstract - *Text Highlighting involves selecting certain important sentences from a document that help provide a good overview of the entire document. In this paper we explore five different machine learning models built to perform text highlighting. We also propose a new method of generating an extractive summarization dataset from human generated summaries for a set of documents, inspired by the work of Nallapatti et. Al [1].*

Key Words: **Text Highlighting, Text Summarization, Extractive Summarization, Machine Learning, Support Vector Machine, Neural Network, Convolutional Neural Network**

1. INTRODUCTION

Automatic Text Summarization is the process of reducing document text to highlight the essence of it and retain only the major points of the original document. There are two different approaches to create these summaries – extractive text summarization and abstractive text summarization. Extractive summaries are created by choosing sentences from the text that are necessary to capture the meaning of it while ignoring those sentences that can be removed without losing the meaning of the given text. It does not generate any new sentences. Abstractive summaries, on the other hand, seek to understand the meaning of the sentences and uses Natural Language Processing techniques to generate new sentences that capture this meaning in a shorter text.

The project highlighted in this paper uses only extractive techniques since it focuses on text highlighting rather than summary generation. Text highlighting chooses those sentences verbatim from the text that provide an overview of the entire passage.

Additionally, extractive summarization techniques have shown better results than most abstractive summarization techniques since this technique does not modify the intent of the sentence, especially when the usage of a particular figure of speech is not common across languages or the way a specific language is spoken in different regions. Most abstractive systems use extractive techniques as well to improve the accuracy of the output.

Work has been carried out in the field of automatic text summarization from as early as the 1950s. Early works mainly used features such as word and phrase frequency to identify salient sentences for extractive summarization. Since then, various models such as Naive Bayesian classification, neural networks etc. have been developed to generate these extractive summaries [3]. Support Vector Machines have also been proposed to identify and extract important sentences [6]. This project explores several machine learning models and their performance in extractive text summarization. We also propose a new method of generating extractive summarization datasets from human generated summaries based on the work of Nallapatti et. Al [1]. ConvNets have also been used to perform text summarization [5]. Two different CNNs are constructed in this project and their accuracies compared.

To evaluate the accuracy of the generated summaries, several metrics have been proposed. These include human evaluation of generated summaries and metrics that calculate the deviation of a generated summary from the standard human-created summary [4]. This project uses the ROUGE metric for evaluation which uses n-gram and longest common subsequence statistics to compare the similarity between the standard summaries and the generated summaries.

2. PROPOSED APPROACH

The aim of extractive summarization is to pick the important sentences from a text containing many sentences. To do this, we built a classifier that would accept a sentence as input and classify it as important (required to be included in the summary) or irrelevant (need not be included in the summary).

2.1 Creating a feature Vector Representation of a Sentence

We created a feature vector consisting of 10 features for every sentence. This was done to analyze sentences grammatically and to make inferences about the importance of a sentence based on its construction and placement. We identified a few main features which helps us determine the importance of a sentence. The 10 features we used are as follows.

2.1.1 First Sentence Indicator

In many pieces of text, the first sentence contains important information. For instance, the first sentence of most news articles is vital and gives the reader of a gist of the entire article. If the sentence being processed is the first sentence of a document, the value of this feature is 1, otherwise it is 0.

2.1.2 Presence of Redundant Words

The presence of certain words such as ‘furthermore’, ‘additionally’, ’moreover’ in a sentence indicates that the sentence is an elaboration of a previously established

thought or concept and need not be included in the summary. We collected a list of such words and if such a word is present in the sentence, the value of this feature is 1, otherwise 0.

2.1.3 Presence of Important Terms

Before processing and creating a vector for every sentence in the document, we processed the document to find terms that are important to the context of this document. To do that, we POS Tagged every sentence in the document using the Penn Treebank tag set and found all the terms corresponding to some form of a noun. We then picked the five most frequently occurring terms and labeled these as 'Important Terms'. If a sentence contained at least one of these terms, the values of this feature is 1, otherwise 0.

2.1.4 Relative Sentence Length

We found the length of the current sentence being processed and divided it by the length of the longest sentence in the document to get the relative length of this sentence.

2.1.5 Relative Sentence Position

We found the position of the sentence being processed and divided it by the number of sentences in the document to find the relative position of this sentence in the document.

2.1.6 Sentence-to-Center Sentence Cohesion

We found the sentence at center of the document and found the similarity between the sentence being processed and this sentence to get the Sentence-to-Center Sentence Cohesion. We used Cosine Similarity to compute similarity.

2.1.7 Sentence-to-Longest Sentence Cohesion

We found the longest sentence in the document and found the similarity between the sentence being processed and this sentence to get the Sentence-to-Longest Sentence Cohesion. We used Cosine Similarity to compute similarity.

2.1.8 Sentence-to-Sentence Cohesion

We found the similarity between the sentence being processed and every other sentence in the document and computed the total to get the total similarity score. We then divided this by the number of sentences in the document to get the value of this feature. We used Cosine Similarity to compute similarity.

2.1.9 Number of Nouns

We found the number of nouns (by POS Tagging the sentence and then detecting terms with a relevant noun tag) present in the sentence being processed and then divided it by a constant normalizing factor to get the value of this feature.

2.1.10 Number of Verbs

We found the number of verbs (by POS Tagging the sentence and then detecting terms with a relevant verb tag) present in the sentence being processed and then divided it by a constant normalizing factor to get the value of this feature.

2.2 Models Used for Classification

The model to be used for classification was to be decided after comparing the performance of five prospective models – a simple Artificial Neural Network, a Naïve Bayes Classifier, a Support Vector Machine, and two different Convolutional Neural Networks.

2.2.1 Artificial Neural Network

The first classifier built was a simple Artificial Neural Network. This consisted of an input layer with 10 neurons, a hidden layer with 16 neurons and an output layer consisting of 2 neurons.

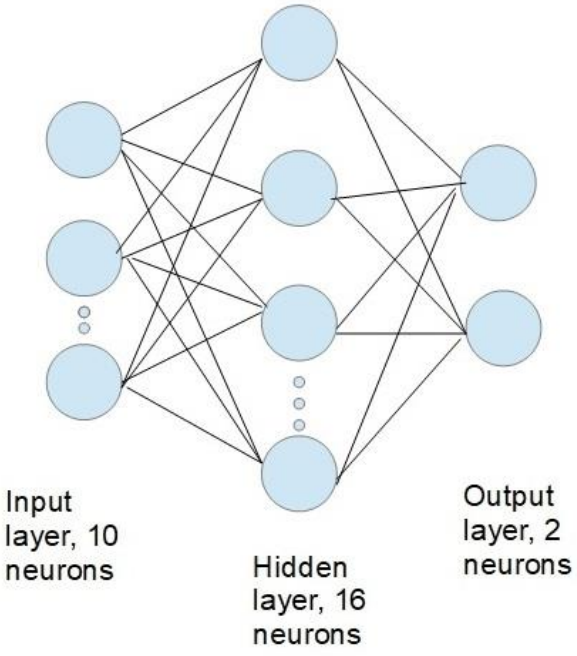

Diagram -1: Artificial Neural Network Used

2.2.2 Naïve Bayes Classifier

The Naïve Bayes Classifier has been used extensively for text categorization. It is a probabilistic classifier based on applying Bayes' theorem with a strong independence assumption between the features. Bayesian decision theory assumes a probabilistic approach to classify the sentences with maximum probability as necessary (required to be included in the summary) and discard the others while constructing the summaries.

2.2.3 Support Vector Machine

A Support Vector Machine is a supervised learning algorithm which can be used for classification and regression. It has been used widely for the purposes of linear classification and can even be used for non-linear classification. It constructs a hyperplane or a set of hyperplanes in a high or infinite dimensional space which it then uses to classify data, which in this case is used to identify if a sentence is to be included in the summary or not.

2.2.4 Convolutional Neural Network:

A convolutional network is used to classify the sentences for text highlighting. This approach is used to test if a convolutional neural network would give better results for text classification. The model used has the following layers- convolutional layer of a single dimension with 16 filters, a dense layer with 16 neurons, a flatten layer and a dense layer with 2 neurons. The optimizer used is an Adam optimizer and a categorical cross entropy loss function is used.

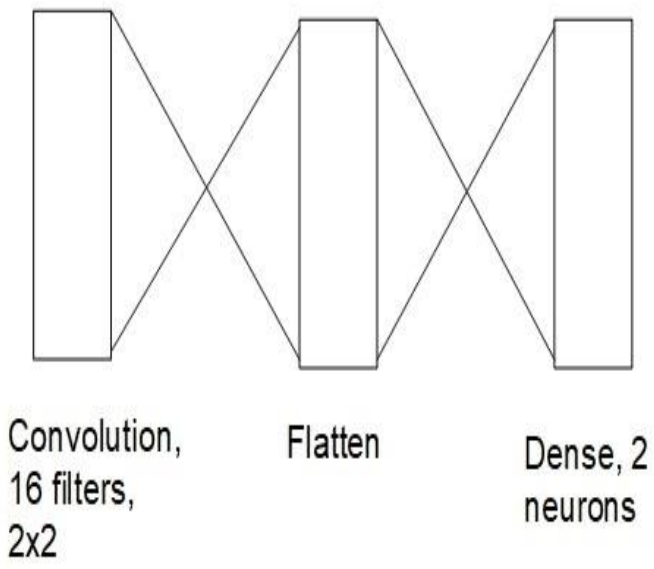

Diagram -2: Convolutional Neural Network Used

Two approaches are used to the vectorization of the words in the document.

2.2.4.1 Approach 1: Vectorization of Dataset using features

In this approach, the feature vectors used in the previous models are preserved. These feature vectors are built on the 10 features presented above.

2.2.4.2 Approach 2: Vectorization of Dataset using Word Embeddings

In this approach, the dataset is vectorized by creating word embeddings for each word. A hash of each word is created by using a one hot encoding. This creates an integer encoding for each word in the dataset. A vocabulary size is specified to create the hash.

3. Extractive Training from Abstractive Summaries

Most summarization datasets contain abstractive human written 'gold standard' summaries. These abstractive summaries are far easier for humans to generate than extractive summaries. However, the dataset required to train our classifier has to be extractive in nature. We needed a dataset containing documents along with a tag of 1 (important and should be included in the summary) or 0 (irrelevant and need not be included in the summary) for every sentence. Very few datasets contained data in this form and moreover these datasets were relatively smaller than other abstractive summarization datasets.

To solve this problem, we devised an algorithm to generate an extractive 1 or 0 dataset, given an abstractive summarization dataset (one that contained a 'gold standard' summary for every document).

This algorithm was inspired by the technique used in SummaRuNNer [1]. While their algorithm picks sentences based on a Greedy approach with the aim of increasing the ROUGE score, our algorithm works in a different manner. Our algorithm first generates all n-grams (we used n=3) from the 'gold standard' summary for a certain document. It then picks all the sentences in the document that contain at least 1 n-gram in common with the list of n-grams we had previously generated. This way, we are able to generate extractive summaries that are good representations of the documents.

4. THE DATASET

We required a large dataset to be able to effectively train and test our classifiers. We use the CNN/Daily Mail corpus [2]. This was initially used for a reading comprehension task as outlined in paper [2]. However, this dataset has been used for summarization tasks as demonstrated by SummaRuNNer[1].

Each article of this dataset contains a set of highlights of article which we treated as a 'gold standard' summary. We then use our algorithm that allows extractive training from abstractive summaries to generate extractive summaries of every article where every sentence of the article is tagged either as important (required to be included in the summary) or irrelevant (need not be included in the summary).

5. EXPERIMENTS

First, we generated an extractive summarization dataset using our algorithm from the CNN/Daily Mail corpus[2]. The ROUGE recall scores, computed for the extractive summaries generated by our algorithm compared with the 'gold standard' summaries, are tabulated below.

Table -1: ROUGE recall scores

ROUGE - 1	ROUGE - 2	ROUGE - L
0.3695	0.1596	0.3434

We selected a subset of the data and equalized the data so that there were an equal number of sentences in both classes. There was a total of 806042 sentences that were used for training and testing. The sentences were then vectorized into a vector of size 10 and all the classifiers were trained and tested with a data split of 90%-10%. The results have been tabulated below.

Table -2: Results of testing

Results				
Classifier	**Accuracy**	**F1 Score**	**Precision Score**	**Recall Score**
ANN	0.7076	0.7067	0.7102	0.7075
Naïve Bayes	0.6204	0.5567	0.7764	0.6190
SVM	0.7074	0.7069	0.7089	0.7074
CNN: Approach 1	0.7104	0.7095	0.7128	0.7102

In addition to these classifiers, we also tested CNN: Approach 2. But due to limited available computational power, we tested this classifier with a vector of size 8 and of a dataset with 13106 sentences.

Table -3: Results of testing

Results				
Classifier	**Accuracy**	**F1 Score**	**Precision Score**	**Recall Score**
CNN: Approach 2	0.6521	0.6444	0.6629	0.6501

6. CONCLUSION AND FUTURE WORK

After testing the first four classifiers we concluded that our first CNN model (CNN: Approach 1) had the highest accuracy. The SVM also came very close in terms of accuracy and can be considered if this project is to be implemented in a device with lower computational power.

We believe that in the future, this system can be extended to other domains, after training it with relevant datasets. We also believe that if the hash based Convolutional Neural Network (CNN: Approach 2) was trained with a larger vector size and on the complete dataset, it would yield better results.

REFERENCES

[1] R. Nallapati, F. Zhai and B. Zhou, "SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents, Published at AAAI 2017", The Thirty-First AAAI Conference on Artificial Intelligence (AAAI-2017)

[2] D. Chen, J. Bolton, C. Manning, "A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task",ACL 2016

[3] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, Krys Kochut, "Text Summarization Techniques: A Brief Survey", arXiv:1707.02268

[4] Dipanjan Das, Andr´e F.T. Martins, "A Survey on Automatic Text Summarization"

[5] Yong Zhang, Meng JooEr, Mahardhika Pratama, "Extractive document summarization based on convolutional neural networks", IECON 2016 – 42nd Annual Conference of the IEEE Electronics Society

[6] Nadira Begum, Mohamed Fattah, Fuji Ren, (2009), "Automatic text summarization using support vector machine", International Journal of Innovative Computing, Information and Control. 5. 1987-1996.