CS 297 Report

Text Summarization for Compressed Inverted Indexes and snippets

By

Mangesh Dahale

SJSU ID: 008661016

Email: mangeshadahale@gmail.com

Project Advisor: Dr. Chris Pollett

Department of Computer Science

San José State University

One Washington Square

San José, CA 95112

INDEX

Text Summarization for Compressed Inverted Indexes and snippets

Introduction

Search engines are the first source of information when we want to do any research. For getting this information, a search engine should understand your query and give results relevant to your query. Summarization is one of the key steps for obtaining these relevant results from the system. The major challenge in summarization lies in distinguishing the more informative parts of a document from the less informative ones. Text summarization is a technique to generate a concise summary of a larger text. In search engines, Text summarization can be used for generating compressed description of the web pages. For indexing, these can be used rather than whole pages when building inverted indexes. For query results, summaries can be used for snippet generation.

Text summarization is usually described as a three-step process: selection of salient portions of text, aggregation of the information for various selected portions and abstraction of this information, and finally, presentation of the final summary text. This process can be used in many applications such as information retrieval, intelligence gathering, information extraction, text mining, and indexing.[1]

In this project, I have implemented a summarization technique for the Yioop search engine. In the initial stage of the project, I implemented three different methods of text summarization and found the best method for the Yioop by evaluating their performances. CS297 was divided into three main deliverables as follows: Study different methods of Text Summarization, Code three methods so that we can evaluate their performances, Evaluate performance of these three methods to find best summarization method.

For Deliverable 1, I researched on the text summarization topic to find out which methods are being used for text summarization and studied three methods in depth so that I can implement them and choose one which is best suited for the Yioop search engine. For Deliverable 2, I implemented the three methods studied in Deliverable 1.

For Deliverable 3, I created a sample document set using which I can compare these three methods and choose the one with high performance and which is best suited for Yioop search engine.

Deliverable 1: Study different methods of Text Summarization

I researched the text summarization topic to find out which methods are being used for text summarization and studied three methods in depth so that I could implement them and choose one which is best suited for the Yioop search engine. The three methods are as follows: 1. Intersection method 2. Centroid method 3. TF-ISF method.

1. Intersection method

This method works on the principle that, if two sentences have a good intersection, they probably hold the same information. So if one sentence has a good intersection with many other sentences, it probably holds some information from each one of them- or in other words, this is probably a key sentence in our text.[5] We use an intersection function to calculate the intersection between two sentences and we create a key-value dictionary, where the sentence itself is the key and the value is the total score.

This method is based on "TextRank – a graph-based approach for text processing". We applied this model for sentence extraction for our summarizer. For this, we need to build a graph associated with the text where the graph vertices are representative for the units to be ranked. Here, our goal is to rank all the sentences which is why we add them as a vertex in the graph. Edges, in this graph, are the similarity between those two sentences where similarity is measured as the function of their content overlap. This overlap between two sentences can be calculated by simply counting the number of common tokens between given two sentences.

2. Centroid method

For this method, we start with the document we wish to summarize. This method gives us a word cloud while generating a summary for that document. For generating word cloud, we are using technique called Topic detection and Tracking which is used in MEAD (multi-document summarizer) to find all the documents with same topic and adding them to a cluster.

What is centroid?

"A centroid is a set of words that are statistically important to a cluster of documents. As such, centroids could be used both to classify relevant documents and to identify salient sentences in a cluster."[2]

In this method, we first find the centroid of the document i.e. we find the main topic of the document. Then we calculate the TF-IDF score of each sentence in the document so that we can get the weight of that sentence in a document. Each document is represented as a weighted vector of TF-IDF scores. After calculating weights, we calculate the cosine similarity between the word cloud i.e. main topic of the document and given sentence from the document by the following formula:

$$sim(D,C) = \frac{\sum_k (d_k \cdot c_k \cdot idf(k))}{\sqrt{\sum_k (d_k)^2}\sqrt{\sum_k (c_k)^2}}$$

After getting all similarity score between the word cloud and each sentence, we keep adding sentences which have the score above threshold in summary until the limit of the summary length is reached.

3. TF-ISF method

In this method, we will be representing the document as a weighted vector of TF*IDF as we did in centroid method. After that, we calculate the cosine similarity of each sentence with every other sentence from the document by using the following formula:

$$sim(s_i, s_j) = \frac{\sum_{k=1}^{m} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{m} w_{ik}^2 \cdot \sum_{k=1}^{m} w_{jk}^2}}, \quad i, j = 1,...., n$$

With cosine similarity scores, we also calculate coverage and diversity of the summary. We enforce coverage and diversity to make the summary more informative and concise by ensuring that it is covering all the topics from the document and removing redundant information from the summary.

Diversity

In diversity, we ensure that the sentences selected do not have the same information. Diversity is an important issue since sentences from different documents might convey the same information. A high quality summary should be informative and compact.

We model diversity with the following objective function:

$$f_{\text{diver}} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (1 - \text{sim}(s_i, s_j)) x_i x_j$$

Higher value of $f_{diver}(\cdot)$ corresponds to lower overlap in content between sentences s_i and s_j.

Coverage

In coverage, we ensure that the sentences in the summary cover all the topics from the document. We attempt to find a subset of the sentences S = {s_1, s_2,.. .,s_n} that covers the main content of the document collection. Generally, a document contains variety of information centered on a main topic, and covers different aspects of the main topic. In coverage, we ensure that all these subtopics are covered in the resulting summary.

$$f_{\text{cover}}(X) = \text{sim}(0, 0^S) + \text{sim}(0, s_i)$$

Here O and O^S denote the centers of the collection S = {s_1, s_2,..., s_n} and the summary

$$S = \bigcup_{i=1}^{n} s_i \, x_i$$

respectively, where x_i denotes a binary variable of the presence of sentence s_i in the summary and $\cup$ is the concatenation operation. Sentence concatenation is an operation of joining the sentences end-to-end. Higher value of $f_{cover}(\cdot)$ corresponds to higher content coverage of summary.

The k^{th} coordinate o_k of the mean vector O is calculated as:

$$o_k = \frac{1}{n} \sum_{i=1}^{n} w_{ik}$$

and the k^{th} coordinate o_k^S of the mean vector O^S we define as:

$$o_k^S = \frac{1}{|S|} \sum_{s_i \in S} w_{ik}$$

where |S| denotes the number of sentences in summary S and k = 1, . . ., m.

Single Objective function

In general, in a multi-objective optimization problem it is not possible to find a single solution that optimizes all the objectives, simultaneously. Therefore, we construct a single objective function.

maximize

$$f = \alpha \cdot \frac{f_{cover}+f_{diver}}{2} + (1-\alpha) \cdot f_{cover} + f_{diver}$$

subject to

$$\sum_{i=1}^{n} l_i x_i \leq L$$

$$x_i \in \{0,1\}, \forall i$$

where 'L' is the length of summary, 'l_i' is the length of sentence 's_i' and $\alpha \in [0, 1]$ is the weighting parameter, specifying the relative contributions of the arithmetic and harmonic means to the hybrid function.

Deliverable 2: Code three methods so that we can evaluate their performances

After studying the above mentioned three methods in depth, I started coding these three methods so that we can evaluate their performances to choose the best. Following is the explanation of those three methods:

1. Intersection method

For implementation of this method, I divided the complete file into sentences and then all those sentences into terms. For storing all the ranks of each sentence, I created a sentence dictionary which is a collection of key value pair where key is the sentence itself and value is score of that sentence.

I have implemented the intersection function to calculate the intersection (I) between each sentence and every other sentence in the document as follows:

$$I = \frac{Number\ of\ common\ terms\ in\ two\ sentences}{(Total\ number\ of\ terms\ in\ first\ sentences + Total\ number\ of\ terms\ in\ second\ sentences)/2}$$

The score is calculated based on this intersection. The score of a sentence is sum of all the intersections between that sentence and every other sentence in the document.

To decide the length of the summary, I implemented a graphical slider so that we can specify the length of the summary we want. Now, we start to add the sentences with the highest scores to the summary until the specified summary length is reached.

2. Centroid method

For implementation of this method, I started with formatting the document to remove special characters. This method also generates a word cloud which contains the terms that covers the main theme of the document. Therefore, we have to remove stop words from the document. Then, I have calculated weights of each term in sentences based on term-frequency (TF) and inverse document frequency (IDF).

After calculating weights, I took 20 terms which have the highest score and showed them on the user interface by changing their font sizes based on their weights in document so that the term having highest weight will appear the biggest among all the other terms.

These 20 terms are the centroid of the document. For scoring all the sentences in the document, I calculated the similarity measure between centroid vector and sentence vector. To specify the length of the summary, I also implemented the graphical slider similar to the slider implemented in the intersection method. For generating our final summary, we keep adding the sentences with the highest similarity with centroid in the summary until the specified summary length is reached.

3. TF-ISF method

For implementing this method, I formatted the document and removed stop words like I did in centroid method. Then, I have calculated the TF-ISF scores where TF is the term frequency of the term and ISF is the inverse sentence frequency of the term. After calculating the weights of each term for each sentence, I have calculated the similarity of each sentence with every other sentence in the document.

This method also enforces coverage and diversity to the summary. So, I have calculated coverage and diversity separately at first and then created a single objective function which mixes coverage and diversity and generates a summary which has good score for coverage and diversity. For implementing single objective function, I have used a simple genetic algorithm in which I am generating an initial population and generating next generation populations based on the score of coverage and diversity from the previous generation population.

Deliverable 3: Evaluate the performance of these three methods to find the best summarization method

To evaluate the performances of these methods, I took five documents related to sports from the Wikipedia and wrote a summary for each document by using my own judgment so that it can be considered as a human generated summary. Then, I ran all three methods on same set of documents. Now, I have both human generated and machine generated summary of each document. Then, I calculated the cosine similarity between the human generated summary and the summary generated by all three methods.

The above procedure gave me the performance of each method for same set of documents. While comparing these methods, I considered two factors, speed and quality of the summary.

In terms of speed, intersection method is at the top and in terms of quality of the summary, centroid method is at the top. Also, the centroid method has the feature of creating a word cloud which can be used to show in search results which will help users to identify the main theme of the webpage in the result.

According to the above performance analysis, we have decided to implement centroid method for Yioop search engine.

Conclusion

For Deliverable 1, I researched the text summarization topic to find out which methods are being used for text summarization and studied three methods in depth so that I can implement them and choose one which is best suited for the Yioop search engine. For Deliverable 2, I implemented the three methods studied in Deliverable 1. For Deliverable 3, I created a sample document set by which I can compare these three methods and choose the one with high performance and which is best suited for Yioop search engine.

According to the performance analysis done in Deliverable 3, I have found that intersection method is the fastest method among the three and the centroid method generates the best summary among the three. I calculated the quality of summary by comparing it with a human generated summary. Also, the centroid method generates a word cloud which helps the user to understand the main topic of the document by just looking at the word cloud. The TF-ISF method also generated a good summary. However, it is not practical in terms of speed. After doing this performance analysis, Professor and I have decided to implement centroid method for Yioop search engine.

In next semester, I will be implementing summarizer based on centroid method and integrate it into Yioop. After integrating, I will be testing Yioop for performance to see improvements in results and also in terms of speed.

References

1. Jones, K. (2007) Automatic summarising : a review and discussion of the state of the art. *University of Cambridge Computer Laboratory*.

2. Mihalcea, R., and Tarau, P. (2004) TextRank: Bringing order into texts. InProceedings of EMNLP.

3. Radev, D. R., Jing, H., Styś, M., & Tam, D.(2007) Centroid-based summarization of multiple documents. *Information Processing & Management, 40*(6), 919-938.

4. Rasim M. Alguliev, Ramiz M. Aliguliyev, Nijat R. Isazade. (2007) Formulation of document summarization as a 0–1 nonlinear programming problem. *Computers & Industrial Engineering, Volume 64, Issue 1 ISSN 0360-8352*.

5. The Tokenizer. Retrieved on Aug 30, 2013, from web site: http://thetokenizer.com/2013/04/28/build-your-own-summary-tool/