Automatic Summarization of Scientific Articles

Abstract

We propose a pipeline for summarizing scientific articles. The pipeline consists of a baseline algorithm, TextRank, to rank sentences within a document based on relevance of the words within each sentence. To imporve the selection of important sentences, each ranked sentence is then given as input to an SVM classifier to decide whether the sentences should be included in the summary or not. The ROUGE results show an increase in the scores over the baseline and we also present an analysis of the why our algorithm performs better than the baseline.

0.1 Introduction

A summary is a text that represents and preserves the important informaiton conveyed by a document in a concise manner. Automatic summarization has been an area of research for the last couple of decades and has been tackled with different approaches. In this paper we will address the problem of summarization of scientific documents and put forward a solution in the form of a pipeline to process scientific articles and output their summaries.

In order to summarize a document, it is first necessary to identify what pieces of text within the document are important enough to represent the information conveyed by the document. Such pieces of text can range from single words (motivated by the problem of keyword extraction) to clauses and sentences within the document to be summarized. In this paper, we will focus on complete sentences for the reason that a sentence represents a complete coherent line of thought compared to using clauses. So it would be easier to plug the chosen sentences into the summary without worrying about any significant loss of continuity.

Another point to be considered is that summaries are generaly meant to be concise and limited in the number of words. It follows that not all information, that has been identified as important, might fit into a summary. Hence, there is a need to rank and choose the most important pieces of informaiton (sentneces) to be included into the summary.

Eventually, all the high ranked sentences can be clubbed together and presented as a summary in what is termed as *extractive summarization*. A further improvement over creating extracts for summaries is to create an abstract of the assembled text which falls in the domain of language generation.

In this study, we will focus on creating extractive summaries. We started our study by building a baseline to rank sentences based on the relevance of words that occur in each sentence. The measure of relevance depends intuitively on how often words appear in different sentences in the document under consideration. Based on this overlap of words among different sentences, the ranking algorithm gives a high score to a sentence if it has a large number of overlaps with other sentences in the same document. The details of this algorithm , TextRank, proposed by [**?**], are discussed in Section 3. To improve the information content of the summary,

0.2 Related Work

0.3 Methodology

The first step was to build a basic system that could be the basline for further experiments and improvements. Another factor for consideration was that the aim of the project was to create an extractive summary consisting of sentences from the same document that needs to be summarized. After studying various approaches used by researchers for single document summarization, I decided to use an unsupervised ranking algorithm. Following is a brief description of the TextRank algorithm, as introduced by *Mihalcea et. al.[]*, to explain how a graph based raking algorithm can be applied to rank sentences within a document.

TextRank

The basic idea implemented by a graph-based ranking model is that of 'voting' or 'recommendation'. When one vertex links to another one, it is basically casting a vote for the other vertex. The importance of a vertex is based on the number of votes casted towards that vertex.

Formally, let *G = (V, E)* be a directed graph with the set of vertices *V* and set of edges *E* where is a subset of *V x V*. For a given vertex V_i, let $In(V_i)$ be the set of vertices that point to it, and let $Out(V_i)$ be the set of vertices that V_i points to. The score of vertex V_i is defined as follows (*Brin and Page []*),

$$S(V_i) = (1 - d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

where *d* is the damping factor that can be set between 0 and 1. The factor *d* is generally set to *0.85* and this the value that was used for my experiments as well. Starting with arbitrary values assigned to each node in the graph, the computation iterates untill convergence below a given threshold is achieved. The final score associated with each vertex represents the *importance* of that vertex within the graph.

Two modifications were introduced by *Mihalcea []* to apply this algorithm to documents for ranking sentences. The first was to define the above mentioned algorithm for undirected graphs in which case, the out-degree of the vertex is equal to the in-degree of the vertex. The other modification was to incorporate the notion of *strength* of the connection between any two vertices as a weight w_ij added to the corresponding edge that connects the two vertices. Consequently, the new formula that takes into account these two modifications and gives the weighted score is

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_j i}{\sum_{V_k \in Out(V_j)} w_j k} WS(V_j)$$

TextRank for Summarization

0.4 Experiments and Results

0.5 Discussion

0.6 Conclusion