Automatic Document Summarization:
A Project for the Data Structures Subject

David Pinto           Paolo Rosso

May 8, 2007

Abstract

Automatic document summarization is the process of creating a short version of a given document (text) by means of a computer program. The aim is to find the most important issues of the original text. There are two broadly accepted techniques for conforming the text summary: (i) extraction and, (ii) abstraction. On the one hand, the extraction techniques just “extract” those parts of the document considered the most important, for instance, keywords, keyphrases, sentences or paragraphs. On the other hand, abstraction can condense a text more strongly than extraction, providing coherence to the summary which requires the use of more complex techniques of natural language generation technology. In this project, the student will use two different approaches in order to obtain a document extract. The extract will consist of the k most representative sentences for the given document. The extracted sentences will be considered a kind of summary. The first approach will be the “TextRank model”, whereas the second will use a simple ranking method based on the Jaccard coefficient.

1 Introduction

The project is aimed to show the skills obtained in the data structures subject and, therefore, it will be considered the efficacy of each implementation. For instance, the use of hashtables and the optimization of calculations is encouraged. It is allowed to used as many already implemented Java classes as needed.

2 Dataset

In order to test the implemented approaches, the DUC dataset will be provided to you. Please contact dpinto@dsic.upv.es in order to get access to this resource. Special indications also will be given in order to correctly evaluate the results of the experiments carried out.
3 Jaccard Similarity Approach

Given a text $T_1$ made up by $n$ sentences ($S_i$), each sentence conforming by a set of terms which ends with the special punctuation symbol ".". The similarity (Sim) for a pair of sentences, $S_i$ and $S_j$, of $T_1$ is obtained as follows:

$$Sim(S_i, S_j) = \frac{|S_i \cap S_j|}{|S_i \cup S_j|}$$

The ranking of the sentence $S_i$ is then performed in the following manner:

$$Ranking(S_i) = \frac{1}{n} \times \sum_{j=1,j<>i}^{n} Sim(S_i, S_j)$$

In order to obtain the $k$ most important sentences, a simple sort algorithm can be applied (we suggest to use Quicksort).

4 The TextRank Model

The TextRank algorithm [2] is based on the PageRank introduced by Page and Brin in 1998 [1]. Although in [2] it is said that this algorithms can be performed when modeling words or phrases to vertex in a graph, we will consider only phrases (sentences) so that we can compare it with the previous approach.

Each sentence is then considered as a graph vertex and, thereafter, the similarity of a given sentence with respect to all the other sentences is calculated. The resulting graph is then introduced to the PageRank algorithm in order to iterate until convergence below a given threshold is achieved.

The following paragraph was taken as is from [2]:

“Formally, let $G = (V, E)$ be a directed graph with the set of vertices $V$ and set of edges $E$, where $E$ is a subset of $V \times V$. For a given vertex $V_i$ let $In(V_i)$ be the set of vertices that point to it (predecessors), and let $Out(V_i)$ be the set of vertices that vertex $V_i$ points to (successors). The score of a vertex $V_i$ is defined as follows:

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

where $d$ is a damping factor that can be set between 0 and 1, usually set to 0.85.”

The edge $E$ between two vertices (sentences) is then the similarity among the two sentences. In order to improve the results it will be required to eliminate weak links from the graph, i.e., those edges whose value be less than the average of all the similarities.

Once the convergence has been obtained, the $S(V_i)$ scores give the final ranking values for the $i$-th sentence and, therefore, as with the previous approach, the $k$ most important sentences can be obtained by sorting these values.
5 Remarks

The Java classes implemented for the above approaches must receive each two parameters as input: (i) the number of sentences (extracts) to obtain as a summary and, (ii) the document from which it is desired to obtain the summary.

In order to improve the obtained results, it is encouraged to conduct the experiments over a pre-processed version of the input document. The preprocessing will consist in the elimination of punctuation symbols (except '.') and the removing of the meaningless document words, named stopwords (please contact dpinto@dsic.upv.es in order to obtain this word list).

References
