Extraction of Text Summary Using Latent Semantic Indexing and Information Retrieval Technique: Comparison of Four Strategies

Abdelghani Bellaachia*, Anand Mahajan*

*Computer Science Department
George Washington University
Washington DC, 20052
bell@gwu.edu
http://seas.gwu.edu/~bell

Abstract. In this paper, we present four generic text summarization techniques. Each technique extracts a text summary by ranking and extracting sentences from an original document. The first method, SUMMARIZER 1, uses standard information retrieval (IR) methods to rank sentences. The second method, SUMMARIZER 2, uses the Latent Semantic Analysis (LSA) technique to identify semantically important sentences, for summary creations. The third method, SUMMARIZER 3, uses a combination of the latent semantic analysis technique, reduction and relevance measure. The fourth method simply uses the TF*IDF (Term frequency * Inverse Document frequency) weighting scheme. Evaluations of the four methods are conducted using Document Understanding Conferences (DUC) datasets from NIST. We have compared the summary of each method with the manual summaries. Summarizer 4, with its lowest overhead, has comparable performance to summarizer 1. Analysis shows that a combination of LSA technique and the relevance measure (Summarizer 3) has the best performance on an average.

1. Introduction

The speed and the scale of information dissemination have dramatically increased with the explosive growth of the worldwide web. Using conventional information retrieval (IR) techniques to find relevant information effectively in a vast sea of accessible text documents on the Internet, has become more and more insufficient. Text search engines serve as information filters that sift out an initial set of relevant documents. Their keyword-based approach retrieves millions of hits by which the user is overwhelmed. Hence, there is a need for techniques to quickly identify the most relevant documents. Text summarizers can be used to help users identify final set of relevant documents. Text search and summarization are two essential technologies that complement each other. Presenting the user with a summary of each document greatly facilitates the task of finding the desired documents.

The goals of text summarizers can be categorized by their intent, focus and coverage [MCDONALD ET AL.]. Intent refers to the potential use of the summary. Firmin and

Chrzanowski divide a summary's intent into three main categories [FIRMIN AND CHRZANOWSKI, 1999]:

- Indicative,
- Informative and
- Evaluative.

Indicative summaries give an indication of the central topic of the original text or enough information to judge the text's relevancy. Informative summaries can serve as substitutes for the full documents. Evaluative summaries express the point of view of the author on a given topic. Focus refers to the summary's scope, whether generic or query relevant. Finally, coverage refers to the number of documents that contribute to the summary, whether the summary is based on a single document or multiple documents.

Text summaries can be either Query-relevant summaries or Generic summaries [Gong and Liu, 2001]. Creating a Query-relevant summary is a process of retrieving the query relevant sentences from the document and can be easily achieved by extending conventional IR technologies. As they are "query-biased", they do not provide an overall sense of the document content. However, a generic summary provides an overall sense of the document's contents and determines which category it belongs to. A good generic summary should contain the main topics of the document while keeping redundancy to a minimum. Since neither query nor topic is provided to the summarization process, it is quite a challenge to develop a high-quality generic summarization method.

In this paper, we present four generic text summarization methods. Each method creates a text summary by ranking and extracting sentences from an original document. The first method, SUMMARIZER 1, uses standard information retrieval (IR) methods to rank sentences. The second method, SUMMARIZER 2, uses the Latent Semantic Analysis (LSA) technique to identify semantically important sentences, for summary creations. The third method, SUMMARIZER 3, uses a combination of the latent semantic analysis technique, reduction and relevance measure. The fourth method simply uses the TF*IDF weighting scheme. We have also compared the performance of all these methods using the Document Understanding Conferences (DUC) datasets from NIST.

The paper is organized as follows. Next section discusses related work. Section 3 introduces the four summarization methods. Performance evaluations are presented in Section 4. Finally, Section 5 concludes the paper.

2. Related Work

Summarization is a hard Natural Language Processing task. It requires semantic analysis, discourse processing and inferential interpretation (grouping of the content using world knowledge). Attempts of performing true abstraction -- creating abstracts as summaries -- have not been very successful. Text abstraction programs produce grammatical sentences that summarize a document's concepts. The concepts in an abstract are often thought of as having been compressed. While the formation of an abstract may better fit the idea of a summary, its creation involves greater complexity and difficulty [Hovy and Lin, 1998]. Fortunately, however, an approximation called extraction is more feasible today. To create an extract, a system simply needs to identify the most important/topical/central topic(s) of

the text and return them to the reader. An extracted summary remains closer to the original document, by using sentences from the text, thus limiting the bias that might otherwise appear in a summary [LUHN, 1958]. Although the summary is not necessarily coherent, the reader can form an opinion of the content of the original. Most automated summarization systems today produce extracts only.

A majority of the research studies have been focused on creating query-relevant text summaries. SUMMARIST is one such attempt to develop robust extraction technology [Hovy and Lin, 1998]. It produces extract summaries in five languages (and has been linked to translation engines for these languages in the MuST system). SUMMARIST is based on the following 'equation': Summarization = Topic Identification + Interpretation + Generation.

The text summarizer from CGI/CMU uses a technique called Maximal Marginal Relevance (MMR) that measures the relevance of each sentence in the document to the user provided query, as well as to the sentences that have been selected and added into the summary [Goldstain et al.]. Selecting sentences that are highly relevant to the user's query, but are different from each other creates the text summary. The Knowledge Management (KM) system from SRA International Inc. extracts summarization features using morphological analysis, name tagging and co-reference resolution [SRA]. They used a machine learning technique to determine the optimal combination of these features in combination with statistical information from the corpus to identify the best sentences to include in a summary. R. Barzilay and M. Elhadad developed a method that creates text summaries by finding lexical chains from the document [BARZILAY AND ELHADAD, 1997]. The Cornell/Sabir system uses the document ranking and passage retrieval capabilities of the SMART text search engines to effectively identify relevant passages in a document [BUCKLEY AND ET AL., 1999]. B. Baldwin and T.S. Morton developed a summarizer that selects sentences from the document until all the phrases in the query are covered [BALDWIN AND T.S. MORTON, 1998]. A sentence in the document is considered to cover a phrase in the query if they co-refer to the same individual, organization, event, etc. The paper by Yihong Gong and Xin Liu [Gong and Liu, 2001], compares the manual summaries with the automated summaries using Recall (R), Precision (P) along with F. They show that the IR-based method performs better on the average.

In this paper, we investigate the performance of four summarization methods that are discussed in detail in the next section.

3. Summarizers

Four generic text summarization methods are presented. They create text summaries by ranking and extracting sentences from the original documents. The methods are:

- **SUMMARIZER 1:** uses standard IR methods to rank sentence relevance ;
- **SUMMARIZER 2:** uses the LSA technique to identify semantically important sentences for summary creations ;
- **SUMMARIZER 3:** uses a combination of the latent semantic analysis technique, reduction and relevance measure ;
- **SUMMARIZER 4:** uses TF*IDF weighting scheme to rank sentences and selects top sentences to form a summary.

Each method tries to select sentences that cover the major topics of the document as much as possible and at the same time, keeps redundancy to a minimum. The initial steps taken for each are as follows:

1. Decompose the document into individual sentences ;
2. Create a weighted term-frequency vector for each sentence.

The weighted term-frequency vector $S_i = [s_{1i}\ s_{2i}\ \ldots\ s_{ni}]^T$ of sentence *i* is defined as:

$$s_{ji} = L(s_{ji}) \,.\, G(s_{ji}) \qquad (1)$$

where,
$L(s_{ji})$ is the local weighting for term j in sentence *i*,
$G(s_{ji})$ is the global weighting for term j in the whole document.

A document is described by a similarity matrix where each column represents the term-frequency vector of each sentence. The matrix can be either normalized or un-normalized and can use one of the following weighting schemes:

- The local weights are :
  - No weight: $L(s_{ji}) = tf(s_{ji})$,
  - Binary weight: $L(s_{ji}) = 1$, if $tf(s_{ji})$?1, $L(s_{ji}) = 0$, otherwise ,
  - Augmented weight: $L(s_{ji}) = 0.5 + 0.5 * (tf(s_{ji})/\ tf(\max))$ where, $tf(\max) = \max\{ tf(1i), tf(2i), \ldots , tf(mi)\}$,
  - Logarithm weight: $L(s_{ji}) = \log (1 + tf(ji))$
- The global weights are:
  - No weighting: $G(s_{ji}) = 1$,
  - Inverse document frequency (IDF): $G(j) = \log(N/n(j))$ where, *N* is the total number of sentences in the document, and *n(j)* is the number of sentences that contain term *j*.
- Normalization
  - Normalizes S_i by its length $| S_i |$.
  - Uses its original form S_i.

3.1. Summarizer 1

This summarizer takes as input, the document to summarize, the desired summary size, the choice of local and global weighting schemes, and the choice of relevance measure computation (Inner Product/Cosine Similarity/Jaccard coefficient, as defined later). It outputs, a summary that is an extract based on the most relevant sentences in the document. The main steps of SUMMARIZER 1 are:

1. Decompose the document into individual sentences and use these sentences to form the candidate sentence set *S*.
2. Create the weighted term-frequency vector A_i for each sentence *i* ? S and the weighted term-frequency vector *D* for the whole document.

3. For each sentence i ? S, compute the relevance measure between A_i and D, which is the Inner Product, or Cosine Similarity, or Jaccard coefficient between A_i and D.
4. Select sentence k that has the highest relevance score and add it to the summary.
5. Delete k from S, and eliminate all the terms contained in k from the document. Re-compute the weighted term-frequency vector D for the whole document.
6. If the number of sentences in the summary reaches the predefined value, terminate the operation: otherwise go to step 3.

In order to determine the Relevance Measure, the following functions are considered:

- *Inner Product (IP)*: $IP(A_i, D) = ? \sum_{k=1}^{t}(a_{ik}*d_k)$ where, a_{ik} is the weight of term k in sentence i and d_k is the weight of term k in the document.
- *Cosine Similarity (Cos)*: $Cos(A_i, D) = ? \sum_{k=1}^{t}(a_{ik}*d_k)/(? \sum_{k=1}^{t} a^2_{ik} * ? \sum_{k=1}^{t} d^2_k)$.
- *Jaccard Coefficient (JC)*: $JC(A_i, D) = ? \sum_{k=1}^{t}(a_{ik}*d_k)/(? \sum_{k=1}^{t} a^2_{ik} + ? \sum_{k=1}^{t} d^2_k - ? \sum_{k=1}^{t}(a_{ik}*d_k))$

In step 4, sentence k that has the highest relevance measure with the document is the one that best represents the major content of the document. Selecting sentences based on their relevance measures ensures that the summary covers major topics of the document. On the other hand, eliminating all the terms contained in k from the document in step 5 ensures that the subsequent sentence selection will pick the sentences with a minimum overlap with sentence k.

3.2. Summarizer 2

This summarizer selects the highest ranked sentences from each salient topic/concept using the Latent Semantic Analysis (LSA). LSA involves the application of singular value decomposition (SVD). Given an m ? n, terms-by-sentences matrix $A = [A_1\ A_2\ \ldots\ A_n]$ where, each column vector A_i represents the weighted term-frequency vector of sentence i in the document, m is the total number of terms and n is the total number of sentences. The SVD of A is defined as [PRESS AND ET AL., 1992]:

$A = U? V^T$ (2)

where,

$U = [u_{ij}]$ is an *mxn* column-orthonormal matrix whose columns are called left singular vectors;

$? = diag(?_1, \ldots, ?_n)$ is an *nxn* diagonal matrix whose diagonal elements are non-negative singular values sorted in descending order, and

$V = [v_{ij}]$ is an *nxn* orthonormal matrix whose columns are called right singular vectors. If $rank(A) = r$, then ? satisfies

$?_1 ? ?_2 ? \ldots ? ?_r > ?_{r+1} = \ldots = ?_n = 0$ (3)

The interpretation of applying the SVD to the terms by sentences matrix A can be made from two different viewpoints. From transformation point of view, the SVD derives a

mapping between the *m*-dimensional space spanned by the weighted term-frequency vectors and the *r*-dimensional singular vector space with all its axes linearly independent. This mapping projects each column vector *i* in matrix *A*, which represents the weighted term-frequency vector of sentence *i*, to column vector $?_i = [v_{i1}\ v_{i2}\ \ldots\ v_{ir}]^T$ of matrix V^T, and maps each row vector *j* in matrix A, which tells the occurrence count of the term *j* in each of the documents, to row vector $?_j = [u_{j1}\ u_{j2}\ \ldots\ u_{jr}]^T$ of matrix U. Here each element v_{ix} of $?_{i,}\ u_{jy}$ of ?j is called the index with the *x*?th, *y*?th singular vectors, respectively. From semantic point of view, the SVD derives the latent semantic structure from the document represented by matrix *A* [DEERWESTER ET AL., 1990]. This operation reflects a breakdown of the original document into *r* linearly independent base vectors or concepts. Each term and sentence from the document is jointly indexed by these base vectors/concepts. The magnitude of the corresponding singular value indicates the importance degree of this pattern within the document. Any sentences containing this word combination pattern will be projected along this singular vector, and the sentence that best represents this pattern will have the largest index value with this vector. The main steps of SUMMARIZER 2 are:

1. Decompose the document *D* into individual sentences, and use these sentences to form the candidate sentence set *S*, and set $k = 1$.
2. Construct the terms by sentences matrix *A* for the document *D*.
3. Perform the SVD on *A* to obtain U, the singular value matrix ? , and the right singular vector matrix V^T. In the singular vector space, each sentence *i* is represented by the column vector $?_i = [v_{i1}\ v_{i2}\ \ldots\ v_{ir}]^T$ of V^T.
4. Select the k'^{th} right singular vector from matrix V^T.
5. Select the sentence that has the largest index value with the k'^{th} right singular vector, and include it in the summary.
6. If *k* reaches the predefined number, terminate the operation: otherwise, increment k by one, and go to Step 4.

In step 5, finding the sentence that has the largest index value with the k'^{th} right singular vector is equivalent to finding the column vector $?_i$ whose k'^{th} element v_{ik} is the largest. This operation is equivalent to finding the best sentence describing the concept/topic represented by the k'^{th} singular vector. Since the singular vectors are sorted in descending order of their corresponding singular values, the k'^{th} singular vector represents the k'^{th} important concept/topic. Because all the singular vectors are independent of each other, the sentences selected by this method contain minimum overlap.

3.3. Summarizer 3

This summarizer takes as input, the document to summarize, the desired summary size, the choice of local and global weighting schemes, the choice of relevance measure computation (Inner Product/Cosine Similarity/Jaccard Co-efficient as defined earlier). It outputs, a summary that is an extract of the desired size. Summarizer 3 performs singular value decomposition, followed by reduction and then by relevance measure computation to determine the sentences to add to the summary.

After performing SVD on *A* (as discussed in Summarizer 2), the singular values obtained signify the maximum possible weighted concepts in the collection of sentences. Equation (3)

above shows that the numbers of non-zero singular values '*r*' signify the number of weighted concepts. The Dimension Reduction of the SVD components i.e. U, ? and V^T is defined as follows:

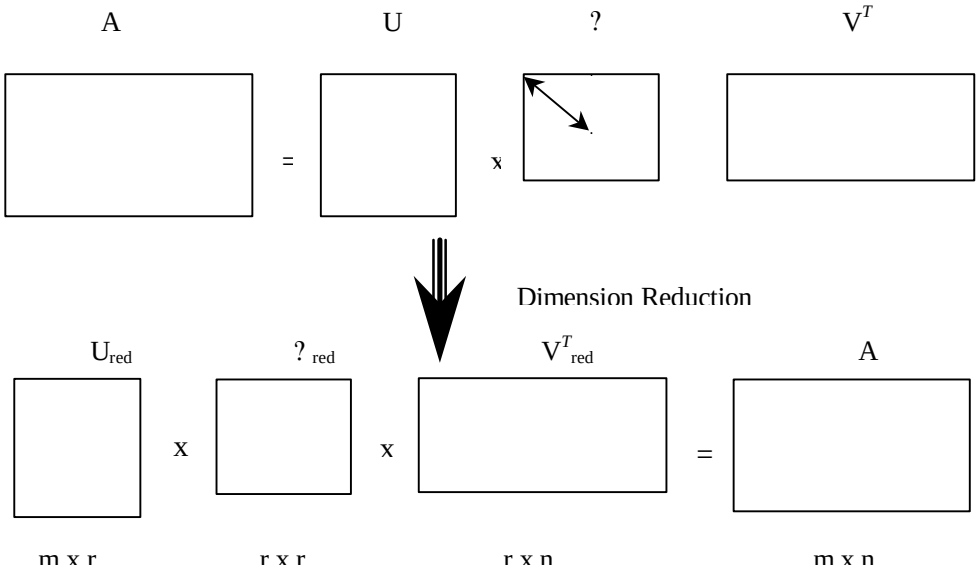

FIG. 1 SVD Dimension Reduction

After the decomposition, the dimension of each component is reduced based on the value of 'r' i.e. the non zero singular values as shown above. In our case we use r = n.

A is obtained again after re-multiplying the reduced components (matrices). *A* now contains the most weighted information in a very high dimensional feature space. Each column vector in *A* represents a sentence. Perform the Relevance Measure computation between each sentence and the Document *D*. Rank all the sentences and pick the sentence with the highest score and add it to the summary. Remove all terms in the sentence from the document and re-compute *D*. Repeat relevance measure computation and build the summary of the desired size.

The operation flow is as follows:

1. Decompose the document into individual sentences, and use these sentences to form the candidate sentence set *S*.
2. Construct the terms-by-sentences matrix *A* for the document.
3. Perform the SVD on *A* to obtain U, the singular value matrix ? , and the right singular vector matrix V^T. In the singular vector space, each sentence *i* is represented by the column vector $?_i = [v_{i1}\ v_{i2}\ \ldots\ v_{ir}]^T$ of V^T.
4. Perform Reduction where *r = n*.
5. Obtain *A* again by re-multiplying U, ? and V^T.

6. Now, use the desired Relevance Measure and add the highest ranked sentence *k* to the summary.
7. Remove all terms in *k* from *D* and *A* and re-compute *D*.
8. Repeat thru step 6 until the summary of desired size is formed.

The SVD operation is equivalent to finding the salient concepts/topics represented by the right singular vectors. Since the singular vectors are sorted in descending order of their corresponding singular values, the *r* right singular vectors represent the *r* important concepts/topics. After applying reduction ($r = n$) to each matrix, *A* contains the most weighted information in a very high dimensional feature space. Also, selecting sentences based on their relevance scores ensures that the summary covers major topics of the document. On the other hand, eliminating all the terms contained in *k* from the document in step 7 ensures that the subsequent sentence selection will pick the sentences with a minimum overlap with *k*.

3.4. Summarizer 4

This summarizer select sentences the TF*IDF weighting schema to select sentences. It is the simplest among all the proposed techniques. It works as follows:

1. Decompose the document into individual sentences and use these sentences to form the candidate sentence set *S*.
2. Create the weighted term-frequency vector A_i for each sentence *i* ? S using TF*IDF.
3. Sum up the TF*IDF score for each sentence and rank them.
4. Select the predefined number of sentences in the summary from A.

4. Performance Evaluation

In this section, we compare the automated summarization outputs (extracts) from each Summarizer, with the manual summaries (abstracts) generated by independent human evaluators. We have used Document Understanding Conferences (DUC) datasets from NIST for performance evaluation. The dataset includes three sets of documents from each independent human evaluator/selector. Each set has between 3 and 20 documents. Each selector builds summaries (abstracts) for each document in the set with an approximate length of 100 words. A sample of the DUC data was chosen for our test purposes. It comprises of 2 sets of documents (one set from each of 2 selectors). The set from Selector 1 consists of 5 documents, whereas the set from Selector 2 contains 4 documents. Each selector creates a summary (abstract) called "Original Summary", for each document in his/her set. Also a selector, other than the original selector of the document, creates a summary (abstract) called "Duplicate Summary", for each document. Thus there are two manual summaries (abstracts) for each document. Abstracts (Original/Duplicate) for the same document may be of different sizes (no. of sentences).

We create automated summaries (similar in size to the manual summaries) for all the nine documents from the two selectors. We then compare the automated summaries with the

manual summaries using "Cosine Similarity" measure to see which summary matches the document more closely. Due to the lack of space in this paper, we only present the results of comparison between automated summaries and original summaries. Other results can be found in [Bellaachia and Mahajan, 2003]. Figure 2 and 3 shows the performance of the four summarizers using inner product and NTN:

N = No local weighting,
T = IDF for global weighting and
N = No normalization (see weighting schemes)

The figures show that SUMMARIZER 1 and 3 have comparable performance. Note that SUMMARIZER 4 has the lowest overhead among all other summarizers.

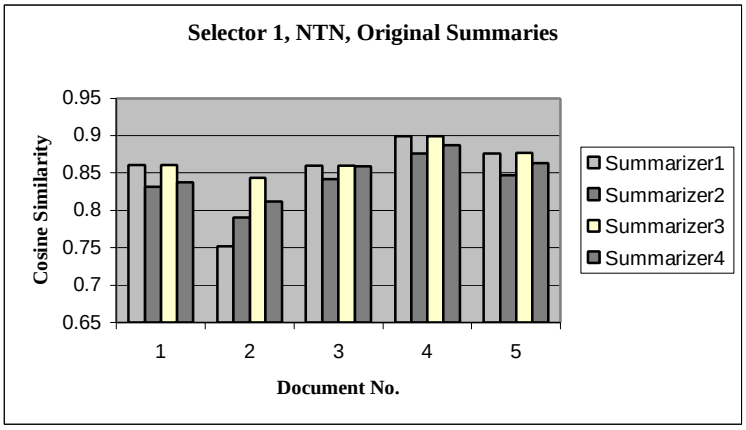

FIG. 2 - Selector 1 Results

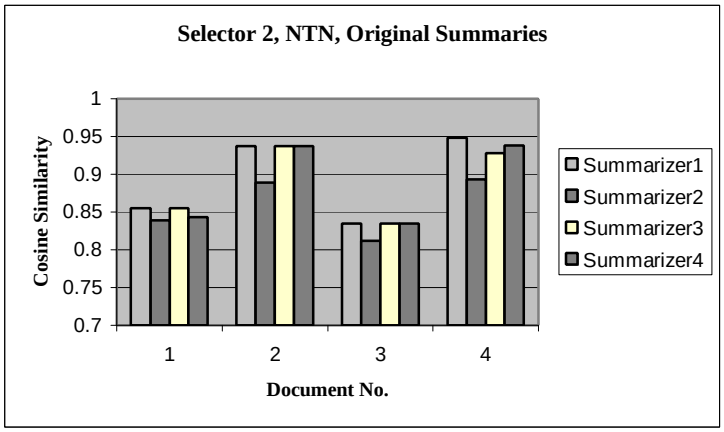

FIG. 3 – Selector 2 Results

Figure 4 shows the cosine similarity of a sentence (within an automated summary generated by each summarizer) with the input document. SUMMARIZER 2 has the lowest measure and Summarizer 3 has the highest measure, while SUMMARIZER 1 and 4 have comparable performance.

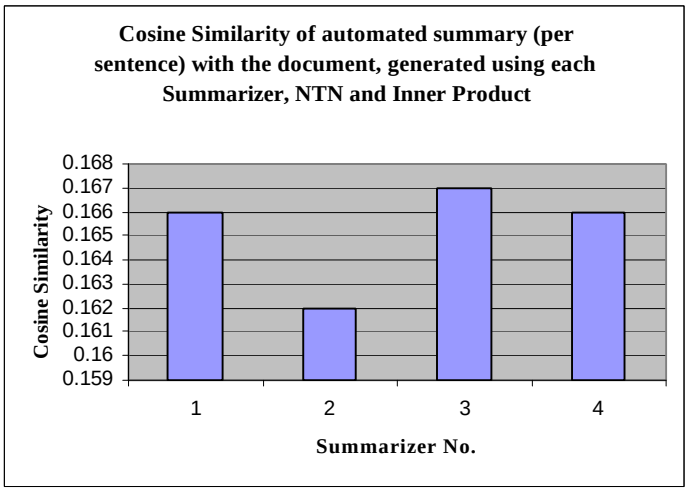

FIG. 4 - Sentence Performance

5. Concllusion

This paper presented four text summarization methods that create generic text summaries by ranking and extracting sentences from the original documents. The first method uses standard information retrieval methods to rank sentence relevance, while the second method uses the LSA technique to identify semantically important sentences. The third method, SUMMARIZER 3, uses a combination of the latent semantic analysis technique, reduction and relevance measure. The fourth method simply uses the TF*IDF weighting scheme.

We have used Document Understanding Conferences (DUC) datasets from NIST for performance evaluation. Two sets of documents were chosen for our performance evaluation. Summarizer 4, with its lowest overhead, has comparable performance to summarizer 1. The LSI technique, as used in SUMMARIZER 2 and Summarizer 3, does improve text summarization. The combination of several techniques used in Summarizer 3 has the best performance on an average.

Acknowledgement: We would like to thank Mr. Avinash K. Kanal for his implementation of LSA technique.

References

[Gong and Liu, 2001] Yihong Gong and Xin Liu. Generic Text Summarization Using Relevance Measure and latent Semantic Analysis. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 19 – 25, 2001.

[Hovy and Lin, 1998] E.Hovy and C. Lin. Automated text summarization in summarist. In *Proceedings of the TIPSTER Workshop*, Baltimore, MD, 1998.

[Goldstain et al.] J. Goldstain, M. Kantrowitz, V. Mittal and J. Carbonell. Summarizing text documents: Sentence selection and evaluation metrics. In *Proceedings of ACM SIGIR '99*, Berkeley, CA, Aug 1999.

[SRA] http://www.SRA.com.

[Barzilay and Elhadad, 1997] R. Barzilay and M. Elhadad. Using lexical chains for text summarization", in Proceedings of the Workshop on Intelligent Scalable Text Summarization, Madrid, Spain, Aug. 1997.

[Buckley and et al., 1999] C. Buckley and et al.. The smart/empire tipster ir system. In *Proceedings of TIPSTER Phase III Workshop*. 1999.

[Baldwin and T.S. Morton, 1998] B. Baldwin and T.S. Morton. Dynamic coreference based summarization. In Proceedings of The Third Conference on Empirical Methods in Natural Language *Processing (EMNLP3)*, Granada, Spain, June 1998.

[Luhn, 1958] Luhn, H.P. The Automatic Creation of Literature Abstracts. in Maybury, M.T. ed. *Advances in Automatic Text Summarization*. The MIT Press, Cambridge, 1958, 15-22.

[FIRMIN AND CHRZANOWSKI, 1999] Firmin, T. and Chrzanowski, M.J. An Evaluation of Automatic

Text Summarization Systems. in Maybury, M.T. ed. *Advances in Automatic Text Summarization*, The MIT Press, Cambridge, 1999.

[McDonald et al.,] D. McDonald et al., "Using Sentence-Sentence Heuristics to Rank Text Segments in TXTRACTOR," MIS Dept., U. of Arizona, Tucson, AZ.

[Press and et al., 1992] W. Press and et al., Numerical Recipes in C: The Art of scientific Computing. Cambridge, England: Cambridge University Press, 2 ed., 1992.

[Deerwester et al., 1990] S. Deerwester et al., "Indexing by latent semantic analysis," *JASIS*, vol. 41, pp 391-407, 1990.

[Bellaachia and Mahajan, 2003] Abdelghani Bellaachia and Anand Mahajan**.** Comparison of Three Text Summarization Methods. The 12th International Conference on Intelligent and Adaptive Systems and Software Engineering, San Francisco, California, July, 2003.

Résumé

Ce papier présente quatre méthodes d'extraction de résumé automatique de texte. Elles sont présentées chacune à leur tour et comparées à un processus manuel de résumé (par extraction de phrases pertinentes). Les méthodes 2 et 3 utilisent la décomposition aux valeurs singulières (documents divisés en phrases) pour mieux sélectionner les phrases les plus typiques. Les quatre méthodes ont été évaluées utilisant une collection de texte de NIST.