Knowledge-poor Multilingual Sentence Compression

Josef Steinberger, Roman Tesar
University of West Bohemia in Pilsen,
Department of Computer Science and Engineering,
30614, Univerzitni 22, Plzen, Czech Republic
Tel.: +420 377 63 2401
E-mail: jstein@kiv.zcu.cz, romant@kiv.zcu.cz

Abstract

We present a feature-based method for sentence compression. Firstly, a summary is created by our summarization method based on latent semantic analysis. The compression approach then removes unimportant clauses from the summary sentences. For each sentence a set of its possible compressed forms (compression candidates) is created. The candidates are then classified using 8 proposed features into two classes: in the first class there are candidates in which the important information was removed by compression and in the second class the information was still contained. The shortest candidate from the latter group substitutes the full sentence in the summary. The features are knowledge-poor which enables them to work with whatever language and the method can be easily extended by other features.

Keywords

Sentence Compression, Text Summarization, Latent Semantic Analysis

I Introduction

Text Summarization is a research area that attracts many NLP research groups around the world. Its aim is to take a source text and present the most important content in a condensed form in a manner sensitive to the needs of the further task. Being able to produce summaries that would be close to human abstracts would affect many tasks in information retrieval. However, true abstraction was not that successful so far. Most present summarization systems do only sentence extraction. The first task that could get the extracts closer to human abstracts is sentence compression. It can compress the extract and it can make it more concise.

Sentence compression can be linked to summarization at the sentence level. Previous data-driven approaches [1, 2] relied on parallel corpora to determine what is important in a sentence. The models learned correspondences between long sentences and their shorter counterparts, typically employing a rich feature space induced from parse trees. In [3] an algorithm based on Rhetorical Structure Theory[1] is proposed. They use the fact that nuclei

[1] Rhetorical Structure Theory is a descriptive theory about text organization. The theory consists of a number of rhetorical relations that tie together text spans, and a number of recursive schemas specifying how texts are structurally composed in a tree-like representation. Most relations are binary and asymmetric: they tie together a nucleus (central to the writer's goal) and a satellite (less central material).

are more likely to be retained when summarizing than satellites. The output can be then obtained simply by removing satellites. The task is challenging since the compressed sentences should retain essential information and convey it grammatically.

In [4] we presented a summarization approach based on latent semantic analysis (LSA) [7]. The analysis identifies the most important topics of the document and the sentences that contain these topics are selected for the summary. In [5] we improved the method with anaphoric information. However, long sentences have better chance to be selected for the summary although they sometimes contain clauses that are unimportant for the summary. Being able to identify and remove these clauses would have a great impact on the quality of the summary. In [6] we proposed an approach that firstly identifies the compression candidates, the form of the sentence which the original one can be compressed to, and than the best candidate was selected to represent the sentence. The latter part worked with only two features: score of the candidate, assigned by summarization algorithm, and his length. The best candidate retained the most of the full sentence score and shortened the most the sentence. Practically, candidate's *quality* was measured as the ratio of candidate's score and candidate's length. Although the results were not that bad we concluded the paper that we need to find more features for the selection of the best candidate. And this is what we propose in this paper.

The requirements were to create a method that uses features that are not dependent on any particular language, that would be easily extended by other features, and that would use minimal knowledge resources. It could serve as a baseline for knowledge-rich sentence compression approaches.

In the next section we briefly overview our LSA-based summarization method. Section III discusses the proposed compression method and it is the crucial part of the paper. Firstly, the process of generation of compression candidates is described and secondly, the chosen features, including the one based on LSA score, are presented. The fourth section deals with experiments we performed. At the end we draw conclusions with some ideas for further development in this area.

II Overview of LSA-based Summarization Method

Our approach to summarization [4] follows what has been called a term-based approach [8]. In term-based summarization, the most important information in a document is found by identifying its main 'terms' ('topics' – combinations of terms), and then extracting from the document the most important information about these terms/topics.

LSA [7] is a fully automatic mathematical/statistical technique for extracting and representing the contextual usage of words' meanings in passages of discourse. The basic idea is that the aggregate of all the word contexts in which a given word does and does not appear provides mutual constraints that determine the similarity of meanings of words and sets of words to each other. LSA has been used in a variety of applications (e.g., information retrieval, document categorization, information filtering, and text summarization).

The heart of the analysis in summarization background is a document representation developed in two steps. The first step is the creation of a term by sentence matrix $A = [A_1, A_2, \ldots, A_n]$, where each column A_i represents the weighted term-frequency vector of sentence i in the document under consideration. If there are m terms and n

sentences in the document, then we will obtain an $m \times n$ matrix A. The next step is to apply Singular Value Decomposition (SVD) to matrix A. The SVD of an $m \times n$ matrix A is defined as:

$$A = U \Sigma V^T, \quad (1)$$

where $U = [u_{ij}]$ is an $m \times n$ column-orthonormalmatrix whose columns are called *left singular vectors*. $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$ is an $n \times n$ diagonal matrix, whose diagonal elements are non-negative *singular values* sorted in descending order. $V = [v_{ij}]$ is an $n \times n$ orthonormal matrix, whose columns are called *right singular vectors*. The dimensionality of the matrices is reduced to r most important dimensions and thus, U is $m \times r$, Σ is $r \times r$ and V^T is $r \times n$ matrix.

From a mathematical point of view, SVD derives a mapping between the m-dimensional space specified by the weighted term-frequency vectors and the r-dimensional singular vector space. From an NLP perspective, what SVD does is to derive the *latent semantic structure* of the document represented by matrix A: i.e. a breakdown of the original document into r linearly-independent base vectors which express the main 'topics' of the document. SVD can capture interrelationships among terms, so that terms and sentences can be clustered on a 'semantic' basis rather than on the basis of words only. Furthermore, as demonstrated in [9], if a word combination pattern is salient and recurring in a document, this pattern will be captured and represented by one of the left singular vectors. 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 value with this vector. Assuming that each particular word combination pattern describes a certain topic in the document, each singular vector can be viewed as representing such a topic, the magnitude of its singular value representing the importance degree of this topic.

The summarization method proposed in [10] uses the representation of a document thus obtained to choose the sentences to go in the summary on the basis of the relative importance of the topics they mention, described by the matrix V^T. The summarization algorithm simply chooses for each topic the most important sentence for that topic: i.e., the k^{th} sentence chosen is the one with the largest index value in the k^{th} right singular vector in matrix V^T. We changed the selection criterion to include in the summary the sentences whose vectorial representation in the matrix $\Sigma^2 \cdot V$ has the greatest length, instead of the sentences containing the highest index value for each topic. Intuitively, the idea is to choose the sentences with greatest combined weight across all important topics, possibly including more than one sentence about an important topic, rather than one sentence for each topic. More formally: after computing the SVD of a term by sentences matrix, we compute the length of each sentence vector in $\Sigma^2 \cdot V$, which represents its summarization score as well (for details see [4]). The important property is that we get a score for each sentence that represents how much information is contained in the sentence. The most informative sentences are then selected for the summary. The modified method significantly outperformed the first LSA-based approach.

III Feature-based Sentence Compression Method

Naturally, long sentences with many significant terms are usually selected into the summary. However, they often contain clauses that are unimportant from the summarization point of view. We try to identify these clauses and then to remove them. Firstly, we need to create a set of possible compressed forms of each summary sentence. We call them compression candidates (CC). After the identification step, we try to select the best candidate for each sentence. This best candidate should preserve the grammaticality and the meaning of the full sentence. Finally, the best candidate will substitute the original sentence in the summary. The method need not change a sentence if all of the candidates lost some important information.

1 Generation of Compression Candidates

For this task we need a parser that can derive a sentence tree structure. For English texts we use Charniak parser [11] that is able to mark clauses and catch their dependencies. For Czech texts we work on a similar tool. In our experiments the structures were annotated by hand. We describe the identification algorithm on the following sentence:

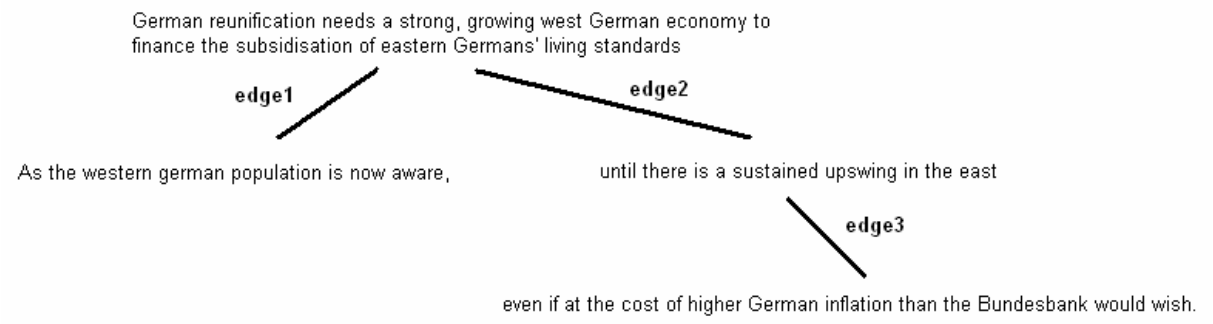

Figure 1: Tree structure of an example sentence.

We can find three edges there. If we cut the tree in an edge we get a compressed sentence where all subordinate clauses of the edge are removed. And more, we can cut the tree more than once - in a combination of edges. However, we can not delete dependent edges. If we apply these rules on the example sentence above we obtain the following five compression candidates:

CC1: German reunification needs a strong, growing west German economy to finance the subsidization of eastern Germans' living standards. (**edge 1** and **edge 2** were removed)

CC2: German reunification needs a strong, growing west German economy to finance the subsidization of eastern Germans' living standards until there is a sustained upswing in the east. (**edge 1** and **edge 3**)

CC3: German reunification needs a strong, growing west German economy to finance the subsidization of eastern Germans' living standards until there is a sustained upswing in the east even if at the cost of higher German inflation than the Bundesbank would wish. (**edge 1**)

CC4: As the western german population is now aware, German reunification needs a strong, growing west German economy to finance the subsidisation of eastern Germans' living standards. (**edge 2**)

CC5: As the western german population is now aware, German reunification needs a strong, growing west German economy to finance the subsidisation of eastern Germans' living standards until there is a sustained upswing in the east. (**edge 3**)

Other combinations of edges could not be applied due to the rules above.

2 Finding the Best Candidate

Next step is selecting the best candidate from the set obtained by the above identification approach. In some of the candidates some important information is removed or even the sense is changed. We designed several features that can help in deciding if the crucial information is retained or not in the particular candidate. This decision is left on a two-class classifier. The features can be categorized into five classes:

A) Length features

A1) Left Terms

The number of left terms in the sentence after performing compression can be a good indicator of the information the candidate contains. If only two or three terms are left in the sentence they hardly can contain the crucial information of the sentence. We count in only significant terms, stop-list words are ignored.

A2) Compression rate

Compression rate is another indicator of candidate's information. If the sentence is compressed to 30% there is a high probability that the most important information has been lost. The compression rate is measured in percentage.

B) Tree depth features

B1) Minimal removed depth

Depth in the clause tree could also work as an indicator. The higher is the depth of the removed clause probably the less important information is removed. Returning to the example sentence in section III.1 CC5 would get a value of 2 by this feature (top level = 0). However, CC2 gets 1 because of the first removed clause. That is where *minimal* in the name of the feature comes from.

B2) Leaf removed

If there is no clause under the removed one it is again an indicator of a possible deletion. The leaves of the clause tree get 1 by this feature, others get 0. According to this feature removing the clause connected by *edge1* will be preferred over the one connected by *edge2*, although both clauses have the same depth in the tree.

C) Co-reference features

C1) Removed entities

Number of entities (persons, organizations, geographical names, or anaphoric references on them) in the removed part of the sentence can serve for determining how much information is removed as well.

C2) Anaphoric interconnections

If the removed clause is connected with the left part by an anaphor (or by containing the same entity) it is a sign of a bad candidate. Like in this example:

> **Former DEA agent Mr. Arthur Sedillo, stationed in *Panama* from August 1977 until August 1979, read a cable from the then-chief administrator of the DEA, *Mr. Peter Bensinger***, in which *he* asked that Gen Noriega be congratulated for his role in an 'impressive seizure' of cocaine being smuggled to the US via *Panama*.

The bold text represents the candidate. It is connected with the removed part by 2 entities – *Bensinger* and *Panama*.

However, sometimes this feature works reversely. Consider the following:

> **The Mafia,** though retaining its strong Sicilian roots**, has spread to Rome and the main cities of northern Italy.**

Here it is connected by *the Mafia – its*. However, annotators proposed to remove the clause which is not bold.

D) Direct discourse feature

D1) Direct discourse out

This feature tries to eliminate compressions in which direct discourse is removed, like in the following example:

> **Officials said** the most significant arrest in Italy was that of Jose 'Tony the Pope' Duran, 38**.**

E) LSA feature

E1) LSA score residual

Our main feature is based on LSA score. The summarization process was described in section II. Each sentence gets a score (LSA score) that measures how much information it contains. However, we can compute this score even for each candidate. Firstly we need to create an input vector for the candidate. The process is the same as when producing the input matrix A (see section 2). The vector has to be transformed to the latent space:

$$v_q = \Sigma U^T q \,,$$

where q is the candidate's weighted term-frequency vector and v_q is the vector transformed to the latent space. Now we received another column in matrix V^T and we can compute the LSA score of the candidate. This feature then measures the ratio of the LSA score of the candidate and LSA score of the full sentence.

Within the set of candidates that were classified as they hold the most important information we chose the shortest one. In other words we substitute the full sentence by the shortest candidate that still contains the crucial information. Consider, for example, the sentence shown in section III.1. The sentence has 5 candidates. CC1 and CC2 could be classified as not losing some important information. By our approach CC1 would be selected to substitute the full sentence in the summary because it is shorter, measured by the number of words.

IV Experiments

1 Corpora

DUC2005

DUC[2] corpus for the year 2005 contains over 1500 newspaper documents organized in 50 clusters. In a cluster there are documents of the same topic. Our LSA-based summarizer produced single-document summaries in two lengths – 100 words and 250 words. Then we analyzed 267 of the summary sentences. This gave 428 candidates. Each candidate was annotated by 2 annotators. They classified it into two classes: if it lost some important information of the full sentence or not. Thus they did the same process as the automatic classifier does using the proposed features. The sentence compression task is subjective. We measured inter-annotator agreement. Percent agreement [12] was 74.1% and Kappa [13] was 0.45. It shows a fair agreement. For getting the compression candidates Charniak parser [11] was used. The sample of documents used for evaluation was annotated for correference by hand, however, we work on an automatic co-reference resolution tool.

CMDSC

One of the requirements on our compression approach was multilinguality. We started with two languages, English and Czech. However, there is not available any summarization corpus of Czech texts. That was the reason for a creation of a new one, according to DUC guidelines. Now, the corpus contains 5 clusters of documents from the main Czech news portals, totally 51 documents. We ran our summarizer, which is completely independent on the language, over those texts. 268 sentences were annotated for sentence compression in the same way as DUC sentences (see the previous paragraph). The 268 sentences gave 454 compression candidates which were annotated again by two annotators. who assigned each candidate to one of the two classes. Percent agreement was 83.5% and Kappa was 0.62. This shows a better agreement then in the case of English texts. Documents were annotated for co-reference by hand.

2 Results

The core of the success in the compression task is in the set of features and the classifier. The better the classifier would work the better the results would be. So the first set of results presented below looks at the task described in the paper from the classification point of view. In the case of English texts we have 428 samples and in the case of Czech texts we have 454 samples. The classifier tried to decide if the candidate still contains the most important information from the full sentence according to the 8 features (see section III.2).

Nowadays, many classification algorithms exist. Among them, mainly SVM (Support Vector Machines) [14] achieves outstanding results in various NLP tasks. Moreover, it accepts input values in the form of numbers, which is suitable, because most of the features presented in section III.2 are naturally numerals. Thus, in our case, SVM appeared to be the appropriate choice.

[2] The National Institute of Standards and Technology (NIST) initiated the Document Understanding Conference (DUC) series to evaluate automatic text summarization. Its goal is to further the progress in summarization and enable researchers to participate in large-scale experiments.

Within the frame of SVM single features were represented in their natural form as integer values. The only exceptions represented A2 (compression rate) and E1 (LSA score residual) features which were simply converted into the range from 0 to 1. This step was not essential. However, values in this range are generally more convenient for SVM classifier.

To train the classifier we used all the data from a corpus and then we classified the same data. One could object that the usual way is to employ the N-cross fold validation technique[3], but in our case we had a small number of candidates and we were afraid that a too small set of training examples could not be sufficient for the classifier. Indeed, employing different training and testing set would probably worsen the overall results presented below.

SVM is a complex classifier with many parameters, which can be possibly adjusted. On the basis of our preliminary experiments we decided to change only the type of kernel function to polynomial, because it provided significantly better results over the other kernel functions. Our results for the English corpus are presented in Table 1, results for the Czech corpus can be found in Table 2.

Used all features except	Overall number of mistakes	Minor mistakes	Major mistakes
A1 - Left Terms	102	45	57
A2 - Compression rate	70	45	25
B1 - Minimal removed depth	71	46	25
B2 - Leaf removed	74	47	27
C1 - Removed entities	76	46	30
C2 - Anaphoric interconnections	75	52	23
D1 - Direct discourse out	76	48	28
E1 -LSA score residual	73	45	28
All features used	71	46	25

Table 1: Classification results obtained for the English corpus

Used all features except	Overall number of mistakes	Minor mistakes	Major mistakes
A1 - Left Terms	103	60	43
A2 - Compression rate	80	70	10
B1 - Minimal removed depth	92	76	16
B2 - Leaf removed	80	61	19
C1 - Removed entities	99	86	13
C2 - Anaphoric interconnections	88	66	22
D1 - Direct discourse out	89	79	10
E1 -LSA score residual	86	74	12
All features used	81	72	9

Table 2: Classification results obtained for the Czech corpus

[3] To split data into N sets, then to use successively a set for testing and the other sets for training, the average value from the N obtained values should be then presented as the final result

The tables above present only the number of incorrectly classified candidates. Considering the fact that the English corpus contains 428 candidates, the classification accuracy when all features are employed is 83.41%, which corresponds to the situation when 357 candidates are classified correctly and 71, as shown in Table 1, incorrectly. Tables also present more details about the incorrectly classified candidates. The columns named *minor mistakes* represent the number of candidates which were marked by annotators to be removed, but the classifier decided to keep them. Similarly, columns named *major mistakes* contain the number of candidates which were marked to be important, but the classifier decided to remove them. Obviously, major mistakes represent a more serious incorrectness which can have a critical impact on the resultant summary. Thus our aim was to suppress the number of this kind of mistakes.

We were also interested in exploring single features and their impacts on the overall classification accuracy. For the English corpus, as we can see in Table 1, the most important feature is *A1-Left Terms*. When we omit this feature the number of minor mistakes decreases from 46 to 45 which is positive, but the number of major mistakes increases from 25 to 57. On the other hand, the usefulness of the feature *A2-Compression rate* in the case of English and Czech corpora appears to be rather questionable. Without the feature *A2-Compression rate* we can reach a slightly better overall classification accuracy 83.64% for the English corpus.

Similarly to the English corpus, also in the case of Czech corpus the feature *A1-Left Terms* seems to be the most important. The Czech corpus contains 454 candidates and the accuracy when all features are employed is 82.16%. Although the overall number 81 of incorrectly classified candidates is higher in comparison to the English corpus, the number of major mistakes is only 9. Also here when we omit the feature *A2-Compression rate* the accuracy improves to 82.38%, but the number of major mistakes increases.

Generally, the results in Table 1 and Table 2 indicate that omitting only single feature can significantly influence the number of minor mistakes and the more important number of major mistakes, either positively or negatively. Thus their choice and usage must be realized carefully.

From the summarization point of view in the case of English texts we have 267 sentences which we try to compress and 268 sentences in the case of Czech texts. Cosider, for example, a sentence that has 5 compression candidates and two of them are considered by the classifier that they still contain the most important information. We select for the substitution of the full sentence in the summary the shorter one. Here are the results:

English sentences: 35.6% of the sentences were correctly compressed. The annotators selected the same candidate for a sentence as the method did. In 20.6% of the sentences annotators proposed a compression but the method left the sentence uncompressed. In 34.5% of the sentences neither the annotators nor the method compressed the sentence. And finally, in 9.3% of the sentences the method made a considerable mistake when it compressed the sentence but annotators proposed to let it uncompressed, or the method and annotators compressed it in a different way. When we consider the length of the summary: 100-word summary was on average shorten to 83 words by the method and to 78 words by annotators; 250-word summary was on average shorten to 208 words by the method and to 196 words by annotators. This shows that the method is more careful

in compressing sentences to avoid the mistakes[4], shortens the summary by about 17%, but every 11th sentence is compressed incorrectly (approximately one sentence in each 2nd 100-word summary and one in each 250-word summary).

Czech sentences: 20.9% of the sentences were correctly compressed. The annotators selected the same candidate for a sentence as the method did. In 19.4% of the sentences annotators proposed a compression but the method left the sentence uncompressed. In 56.3% of the sentences neither the annotators nor the method compressed the sentence. And finally, in 3.4% of the sentences the method made a considerable mistake when it compressed the sentence but annotators proposed to let it uncompressed, or the method and annotators compressed it in a different way. When we consider the length of the summary: 100-word summary was on average shorten to 91 words by the method and to 86 words by annotators; 250-word summary was on average shorten to 227 words by the method and to 216 words by annotators. This shows that fewer compressions were proposed even by annotators for the Czech sentences, which can denote that in Czech texts there were not as many insignificant clauses as in English texts. And again, the method is more careful in compressing sentences to avoid the mistakes, shortens the summary by about 9%, but every 30th sentence is compressed incorrectly (about one sentence in each 6th 100-word summary and in each 3rd 250-word summary).

V Conclusion

We presented a feature-based approach to sentence compression that removes unimportant clauses from summary sentences. A set of compression candidates is determined for each summary sentence. A classifier then decides if some important information was removed by the compression or not and the shortest candidate where the important information was not removed is taken to represent the sentence. It is easily extensible by other features. We designed only knowledge-poor features because we needed to create a multilingual method. We experimented with Czech and English texts, but the principles remain the same for any other language. The results showed that the summary is shortened by 17% in the case of our English summaries and by 9% in the case of our Czech summaries at the cost of a low probability of getting an incorrectly compressed sentence. Sentence compression was applied on single-document summarization. However, it can be easily changed to work with multi-document summarization. The only change is in the computation of the LSA feature. It would have to be computed from an SVD of a cluster of documents instead of one document.

Acknowledgement

This research was partly supported by project 2C06009 (COT-SEWing).

References

[1] K. Knight and D. Marcu: Summarization beyond sentence extraction: A probabilistic approach to sentence compression. In *Artificial Intelligence*, 139(1):91–107, 2003.

[4] It is the result of preferring the precision over recall in the classification step.

[2] S. Riezler, T. H. King, R. Crouch and A. Zaenen: Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of HLT/NAACL*, Edmonton, Canada, 2003.

[3] C. Sporleder and M. Lapata: Discourse chunking and its application to sentence compression. In *Proceedings of EMNLP*, Vancouver, Canada, 2005.

[4] J. Steinberger and K. Jezek: Text summarization and singular value decomposition. In *Proceedings of ADVIS*, Izmir, Turkey, 2004.

[5] J. Steinberger, M. A. Kabadjov, M. Poesio and O. Sanchez-Graillet: Improving LSA-based summarization with anaphora resolution. In *Proceedings of EMNLP*, Vancouver, Canada, 2005.

[6] J. Steinberger and K. Jezek: Sentence Compression for the LSA-based Summarizer. In *Proceedings of ISIM*, Prerov, CR, 2006.

[7] T. K. Landauer and S. T. Dumais: A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. In *Psychological Review*, 104, 211–240, 1997.

[8] E. Hovy and C. Lin: Automated text summarization in SUMMARIST. In *ACL/EACL Workshop on Intelligent Scalable Text Summarization*, Madrid, Spain, 1997.

[9] M. W. Berry, S. T. Dumais and G. W. O'Brien: Using Linear Algebra for Intelligent IR. In *SIAM Review*, 37(4), 1995.

[10] Y. Gong and X. Liu: Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis. In *Proceedings of ACM SIGIR*. New Orleans, 2002.

[11] E. Charniak: A maximum-entropy-inspired parser. In *Proceedings of NAACL*. Philadelphia, 2000.

[12] D.R. Radev, S. Teufel, H. Saggion, W. Lam, J. Blitzer, H. Qi, A. Celebi, D. Liu, E. Drabek: Evaluation Challenges in Large-scale Document Summarization. In *Proceeding of the 41st meeting of the Association for Computational Linguistics*, pp. 375–382, Sapporo, Japan, 2003.

[13] S. Siegel, N.J. Castellan Jr.: *Nonparametric Statistics for the Behavioral Sciences*. Berkeley, CA: McGraw-Hill, 2nd edn., 1988.

[14] T. Joachims: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of the European Conference on Machine Learning*, Springer, 1998.