Using the Essence of Texts to Improve Document Classification

Rada Mihalcea and **Samer Hassan**
Department of Computer Science
University of North Texas
rada@cs.unt.edu, samer@unt.edu

Abstract

This paper explores the possible benefits of the interaction between automatic extractive summarization and text classification. Through experiments performed on standard test collections, we show that techniques for extractive summarization can be effectively combined with classification methods, resulting in improved performance in a text categorization task. Moreover, comparative results suggest that the synergy between text summarization and text classification can be regarded as a new application-oriented evaluation testbed for automatic summarization.

1 Introduction

Text categorization is a problem typically formulated as a learning task, where a classifier learns how to distinguish between categories in a given set, using features automatically extracted from a collection of training documents. In addition to the learning methodology itself, the accuracy of the text classifier also depends to a large extent upon the classification granularity, and on how well separated are the training or test documents belonging to different categories. For instance, it may be a relatively easy task to learn how to classify documents in two distinct categories such as *computer science* and *music*, but it may be significantly more difficult to distinguish between documents pertaining to more closely related topics such as *operating systems* and *compilers*.

Intuitively, if the gap between categories could be increased, the classification performance would raise accordingly, since the learning task would be simplified by removing features that represent potential overlap between categories. This is in fact the effect achieved through feature weighting and selection (Yang & Pedersen 97), (Ng *et al.* 97), which was found to improve significantly over the case where no weighting or selection is performed. We propose a new approach for reducing the potential overlap between documents belonging to different categories, by using a method that extracts the **essence** of a text prior to classification. In this way, only the important sections of a document participate in the learning process, and thus the performance of the text classification algorithm could be improved.

In this paper, we present a graph-based method for automatic summarization through sentence extraction, and show how it can be successfully integrated with a text classifier. Through experiments on standard test collections, we show that significant improvements can be achieved on a text categorization task by classifying extractive summaries, rather than entire documents. We believe that these results have implications not only on the problem of text classification, where the method proposed provides the means to improve the categorization accuracy with error reductions of up to 19.3%, but also on the problem of text summarization, by suggesting a new application-based evaluation of tools for automatic summarization.

2 Graph-based Algorithms for Sentence Extraction

Algorithms for extractive summarization are typically based on techniques for sentence extraction, and attempt to identify the set of sentences that are most important for the understanding of a given document. Some of the most successful approaches to extractive summarization consist of supervised algorithms that attempt to learn what makes a good summary by training on collections of summaries built for a relatively large number of training documents, e.g. (Hirao *et al.* 02), (Teufel & Moens 97). However, the price paid for the high performance of such supervised algorithms is their inability to easily adapt to new languages or domains, as new training data are required for each new type of data.

In this section, we shortly overview an efficient unsupervised extractive summarization method using graph-based ranking algorithms, as proposed in our previous work (Mihalcea & Tarau 04). Iterative graph-based ranking algorithms, such as Kleinberg's HITS algorithm (Kleinberg 99) or Google's PageRank (Brin & Page 98), have been traditionally and successfully used in Web-link analysis, social networks, and more recently in text processing applications (Mihalcea *et al.* 04), (Mihalcea & Tarau 04). In short, a graph-based ranking algorithm is a way of deciding on

the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. The basic idea implemented by the ranking model is that of "voting" or "recommendation". When one vertex links to another one, it is basically casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex.

These graph ranking algorithms are based on a random walk model, where a walker takes random steps on the graph, with the walk being modeled as a Markov process – that is, the decision on what edge to follow is solely based on the vertex where the walker is currently located. Under certain conditions, this model converges to a stationary distribution of probabilities associated with vertices in the graph, representing the probability of finding the walker at a certain vertex in the graph. Based on the Ergodic theorem for Markov chains (Grimmett & Stirzaker 89), the algorithms are guaranteed to converge if the graph is both aperiodic and irreducible. The first condition is achieved for any graph that is a non-bipartite graph, while the second condition holds for any strongly connected graph. Both these conditions are achieved in the graphs constructed for the sentence extraction application considered in this paper.

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).

PageRank (Brin & Page 98) is perhaps one of the most popular ranking algorithms, and was designed as a method for Web link analysis.

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

where d is a parameter set between 0 and 1.

HITS (Hyperlinked Induced Topic Search) (Kleinberg 99) makes a distinction between "authorities" (pages with a large number of incoming links) and "hubs" (pages with a large number of outgoing links). For each vertex, *HITS* produces two sets of scores.

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j) \quad (2)$$

$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j) \quad (3)$$

For each of these algorithms, starting from arbitrary values assigned to each node in the graph, the computation iterates until convergence below a given threshold is achieved. After running the algorithm, a score is associated with each vertex, which represents the "importance" or "power" of that vertex within the graph. Note that the final values are not affected by the choice of the initial value, only the number of iterations to convergence may be different. Somewhat faster or more compact (but significantly more complex) implementations have been recently suggested (Kamvar *et al.* 03; Raghavan & Garcia-Molina 03), but the additional time or memory savings are more relevant in cases like the complete Web graph than it would be in the case of moderately large graphs like a text-based graph.

When the graphs are built starting with natural language texts, it may be useful to integrate into the graph model the *strength* of the connection between two vertices V_i and V_j, indicated as a weight w_{ij} added to the corresponding edge. Consequently, the ranking algorithm is adapted to include edge weights, e.g. for *PageRank* the score is determined using the following formula (a similar change can be applied to the *HITS* algorithm):

$$PR^W(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} w_{ji} \frac{PR^W(V_j)}{\sum_{V_k \in Out(V_j)} w_{kj}} \quad (4)$$

While the final vertex scores (and therefore rankings) for weighted graphs differ significantly as compared to their unweighted alternatives, the number of iterations to convergence and the shape of the convergence curves is almost identical for weighted and unweighted graphs.

For the task of sentence extraction, the goal is to rank entire sentences, and therefore a vertex is added to the graph for each sentence in the text. To establish connections (edges) between sentences, we are defining a similarity relation, where "similarity" is measured as a function of content overlap. Such a relation between two sentences can be seen as a process of "recommendation": a sentence that addresses certain concepts in a text, gives the reader a "recommendation" to refer to other sentences in the text that address the same concepts, and therefore a link can be drawn between any two such sentences that share common content.

The overlap of two sentences can be determined simply as the number of common tokens between the lexical representations of the two sentences, after removing the stopwords. Moreover, to avoid promoting long sentences, we are using a normalization factor, and divide the content overlap of two sentences with the length of each sentence.

[1] Watching the new movie, "Imagine: John Lennon," was very painful for the late Beatle's wife, Yoko Ono.
[2] "The only reason why I did watch it to the end is because I'm responsible for it, even though somebody else made it," she said.
[3] Cassettes, film footage and other elements of the acclaimed movie were collected by Ono.
[4] She also took cassettes of interviews by Lennon, which were edited in such a way that he narrates the picture.
[5] Andrew Solt ("This Is Elvis") directed, Solt and David L. Wolper produced and Solt and Sam Egan wrote it.
[6] "I think this is really the definitive documentary of John Lennon's life," Ono said in an interview.

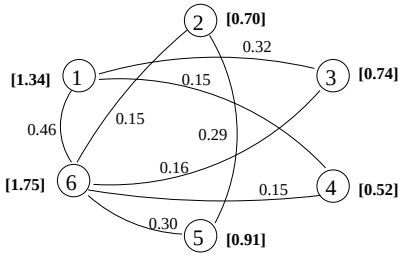

Figure 1: Graph of sentence similarities built on a sample text. Scores reflecting sentence importance, obtained with the graph-based ranking algorithm, are shown in brackets next to each sentence.

The resulting graph is highly connected, with a weight associated with each edge, indicating the strength of the connections between various sentence pairs in the text. The graph can be represented as: (a) simple *undirected* graph; (b) directed weighted graph with the orientation of edges set from a sentence to sentences that follow in the text (*directed forward*); or (c) directed weighted graph with the orientation of edges set from a sentence to previous sentences in the text (*directed backward*).

After the ranking algorithm is run on the graph, sentences are sorted in descending order of their score, and the top ranked sentences are selected for inclusion in the extractive summary. Figure 1 shows an example of a graph built for a sample text of six sentences.

Evaluation. The performance of the sentence extraction method was evaluated in the context of a single-document summarization task, using 567 news articles provided during the Document Understanding Evaluations 2002 (DUC 02). For each article, the method was used to generate a 100-words summary, which is the task undertaken by other systems participating in this single-document summarization task. The evaluation was run using the ROUGE toolkit, which is an evaluation method based on Ngram statistics, found to be highly correlated with human evaluations (Lin & Hovy 03). For each document, two manually produced reference summaries were provided, and used in the evaluation process. Table 1 shows results using the $HITS$ and $PageRank$ algorithms on graphs that are: (a) undirected, (b) directed forward, or (c) directed backward.

	Graph		
Algorithm	Undirected	Forward	Backward
$HITS_A^W$	0.4912	0.4584	**0.5023**
$HITS_H^W$	0.4912	**0.5023**	0.4584
$PageRank$	0.4904	0.4202	**0.5008**

Table 1: Results for extractive summarization using graph-based sentence extraction. Graph-based ranking algorithms: $HITS$ and $PageRank$. Graphs: undirected, directed forward, directed backward.

By ways of comparison, a competitive baseline proposed by the DUC evaluators – consisting of a 100-word summary constructed by taking the first sentences in each article – achieves a ROUGE score of 0.4799. The best performing system in DUC 2002 was a *supervised* system which achieved a score of 0.5011.

The results are encouraging: the sentence extraction method applied on a *directed backward* graph structure exceeds the performance achieved through a simple (but competitive) baseline, and competes with the best performing systems from DUC 2002. Unlike other supervised systems, which attempt to learn what makes a good summary by training on collections of summaries built for other articles, the graph-based method is fully unsupervised, and relies only on the given text to derive an extractive summary. Moreover, due to its unsupervised nature, the algorithm can be easily adapted to other languages, genres, or domains.

3 Text Categorization Using Extractive Summarization

Provided a set of training documents, each document assigned with one or more categories, the task of text categorization consist of finding the most probable category for a new unseen document, based on features extracted from training examples. The classification process is typically performed using information drawn from entire documents, and this may sometime result in noisy features. To lessen this effect, we propose to feed the text classifier with summaries rather than entire texts, with the goal of removing the less-important, noisy sections of a document prior to classification.

The text classification process is thus modified to integrate an extractive summarization tool

that determines the top N most important sentences in each document. Starting with a collection of texts, every document is replaced by its summary, followed by the application of a regular text categorization algorithm that determines the most likely category for a given test document. Figure 2 illustrates the classification process based on extractive summarization.

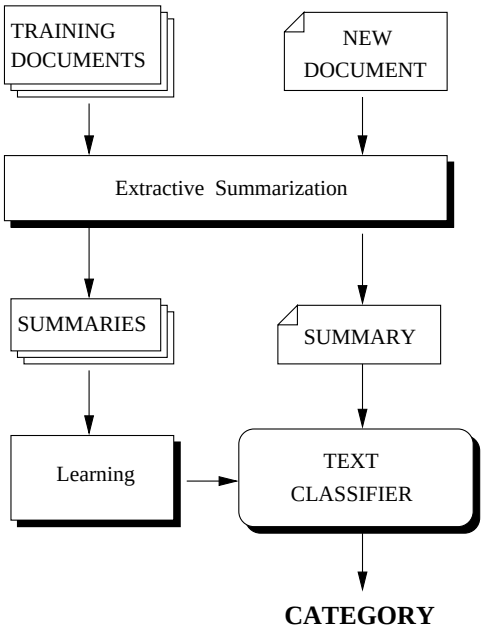

Figure 2: Text classification using extractive summarization.

3.1 Extractive Summarization

To summarize documents, we use the $HITS_A$ *directed backward* graph-based sentence extraction algorithm described in Section 2, which generates extractive summaries by finding the most important sentences in the text. The choice of the algorithm is motivated by several reasons. First, the decision to use an extractive summarization tool, versus more complex systems that include sentence compression and text generation, is based on the fact that in our experiments text summarization is not an end per se, but rather an intermediate step for document classification. The informativeness of a summary is thus more important than its coherence, and summarization through sentence extraction is sufficient for this purpose. Second, through evaluations conducted on standard data sets, the algorithm was found to work best among other graph-based algorithms for sentence extraction, and was demonstrated to be competitive with the state-of-the-art in text summarization. Finally, it is an algorithm that can produce a ranking over sentences in a text, and thus it is well suited for our experiments where we want to measure the impact on text classification of summaries of various lengths.

3.2 Algorithms for Text Classification

There is a large body of algorithms previously tested on text classification problems, due also to the fact that text categorization is one of the testbeds of choice for machine learning algorithms. In the experiments reported here, we compare results obtained with two frequently used text classifiers – Rocchio and Naïve Bayes, selected for the diversity of their learning methodologies.

Rocchio. This is an adaptation of the relevance feedback method developed in information retrieval (Rocchio 71). It uses standard $TFIDF$ weighted vectors to represent documents, and builds a prototype vector for each category by summing up the vectors of the training documents in each category. Test documents are then assigned to the category that has the closest prototype vector, based on a cosine similarity. Text classification experiments with different versions of the Rocchio algorithm showed competitive results on standard benchmarks (Joachims 97), (Moschitti 03).

Naïve Bayes. The basic idea in a Naïve Bayes text classifier is to estimate the probability of a category given a document using joint probabilities of words and documents. Naïve Bayes assumes word independence, which means that the conditional probability of a word given a category is assumed to be independent of the conditional probability of other words given the same category. Despite this simplification, Naïve Bayes classifiers perform surprisingly well on text classification (Joachims 97), (Schneider 04). While there are several versions of Naïve Bayes classifiers (variations of multinomial and multivariate Bernoulli), we use the multinomial model (McCallum & Nigam 98), which was shown to be more effective.

3.3 Data

For the classification experiments, we use the *Reuters-21578*[1] and *WebKB*[2] data sets – two of the most widely used test collections for text classification. For *Reuters-21578*, we use the standard *ModApte* data split (Apte *et al.* 94), obtained by eliminating unlabeled documents, and selecting only categories that have at least one document in the training set and test set (Yang & Liu 99). For *WebKB*, we perform a four-fold

[1]Publicly available at http://www.daviddlewis.com/ /resources/testcollections/reuters21578/

[2]Publicly available at http://www-2.cs.cmu.edu/afs/ /cs.cmu.edu/project/theo-20/www/data/

cross validation after removing the *other* category, using the 3 : 1 training/test split automatically created with the scripts provided with the collection, which separates the data into training on three of the universities plus a miscellaneous collection, and testing on a fourth held-out university. Both collections are further post-processed by removing all documents with less than 10 sentences, resulting into final data sets of 1277 training documents, 436 test documents, 60 categories for *Reuters-21578*, and 350 training documents, 101 test documents, 6 categories for *WebKB*. This last step is motivated by the goal of our experiments: we want to determine the impact of various degrees of summarization on text categorization, and this is not possible with very short documents.

3.4 Evaluation Metrics

The evaluation is run in terms of accuracy, defined as the number of correct assignments among the document–category pairs in the test set. For the *WebKB* data set, since the classifiers assign exactly one category to each document, and a document can belong to only one category, this definition of accuracy coincides with the measures of precision, recall, and F-measure. For the *Reuters-21578* data set, multiple classifications are possible for each document, and thus the accuracy coincides with the classification precision. Evaluations figures are reported as *micro-average* over all categories in the test set.

4 Experimental Results

Classification experiments are run using each of the two learning algorithms, with documents in the collection represented by extracts of various lengths. The classification performance is measured for each experiment, and compared against a traditional classifier that performs text categorization using the original full-length documents.

Table 2 shows the classification results obtained using the *Reuters-21578* and the *WebKB* test collections. Note that these results refer to data subsets somewhat more difficult than the original test collections, and thus they are not directly comparable to results previously reported in the literature. For instance, the average number of documents per category in the *Reuters-21578* subset used in our experiments is only 25, compared to the average of 50 documents per category available in the original data set[3]. Similarly, the *WebKB* subset has an average number of 75 documents per category, compared to 750 in the original data set.

Figures 3 and 4 plot the classification accuracy on the two data sets for each learning algorithm, using extractive summaries of various lengths generated with the graph-based sentence extraction method. For a comparative evaluation, the figure plots results obtained with methods that create an extract by: (a) selecting the N most important sentences using the graph-based sentence extraction algorithm; (b) selecting the first N sentences in the text; (c) randomly selecting N sentences; (d) selecting the N *least* important sentences using the low-end of the ranking obtained with the same graph-based sentence extraction algorithm; (e) selecting the last N sentences in the text.

From a **text categorization** perspective, the results de-monstrate that techniques for text summarization can be effectively combined with text classification methods to the end of improving **categorization performance**. On the *Reuters-21578* data set, the classification performance using the Rocchio classifier improves from an accuracy of 74.90% to 77.00% obtained for summaries of 6 sentences. Similar improvements are observed for the Naïve Bayes classifier, where the highest accuracy of 78.38% is obtained again for summaries of 6 sentences, and is significantly better than the accuracy of 75.01% for text classification with full-length documents. The impact of summarization is even more visible on the *WebKB* data set, where summaries of 5 sentences result in a classification accuracy of 79.24% using a Naïve Bayes classifier, significantly higher than the accuracy of 74.25% obtained when the classification is performed with entire documents. Similarly, the categorization accuracy using a Rocchio classifier on the same data set improves from 63.36% for full-length documents to 66.35% for summaries of 5 sentences[4]. The highest error rate reduction observed during these experiments was 19.3%, achieved with a Naïve Bayes classifier applied on 5-sentence extractive summaries.

Another aspect worth noting is the fact that these results can have important implications on the problem of text classification for documents for which only abstracts are available. As it was previously suggested (Hulth 03), many documents on the Internet are not available as full-texts, but only as abstracts. Similarly, documents that are typically stored in printed form, such as books, journals, or magazines, may have an abstract available in electronic format, but not the entire

[3]Using the same Naïve Bayes and Rocchio classifier implementations on the entire *Reuters-21578* data set results in a classification accuracy of 77.42% for Naïve Bayes and 79.70% for Rocchio, figures that are closely comparable to results previously reported on this data set.

[4]The improvements observed in all classification settings are statistically significant at $p < 0.05$ level (paired t-test).

Summary length	*Reuters-21578*		*WebKB*	
	Rocchio	Naïve Bayes	Rocchio	Naïve Bayes
1 sentence	70.18%	72.94%	60.40%	61.39%
2 sentences	72.48%	73.85%	62.38%	71.29%
3 sentences	73.17%	75.46%	60.40%	74.26%
4 sentences	72.94%	75.46%	63.37%	76.24%*
5 sentences	75.23%	75.92%	66.35%*	79.24%*
6 sentences	77.00%*	78.38%*	65.37%*	77.24%*
7 sentences	76.38%*	77.61%*	65.34%*	76.24%*
8 sentences	76.38%*	77.38%*	64.36%	76.21%*
9 sentences	75.23%	77.61%*	65.34%*	75.25%
10 sentences	75.23%	77.52%*	65.35%*	75.25%
full document	74.91%	75.01%	63.36%	74.25%

Table 2: Classification results for *Reuters-21578* and *WebKB*, using Rocchio and Naïve Bayes classifiers, for full-length documents and for extractive summaries of various lengths. Statistically significant improvements ($p < 0.05$, paired t-test) with respect to full-document classification are also indicated (*).

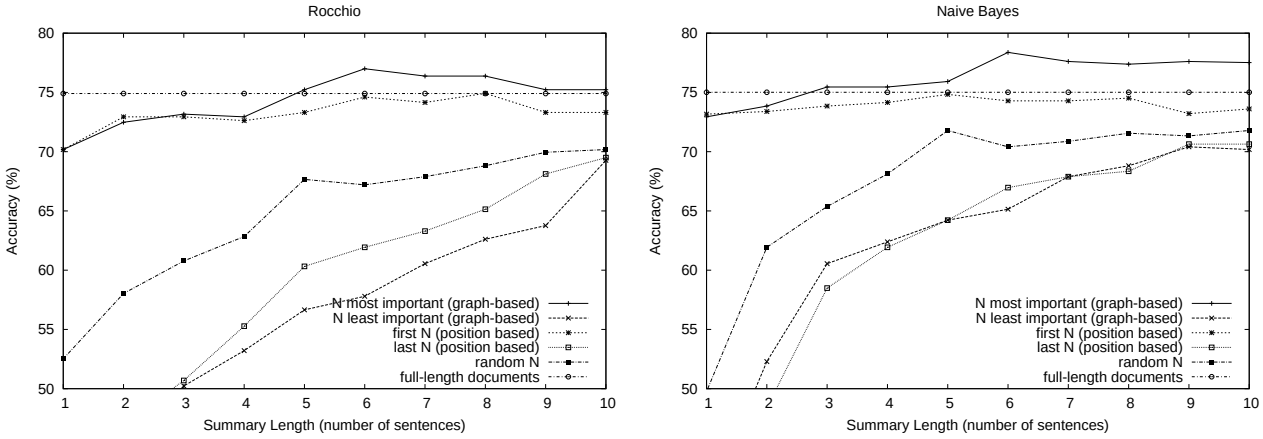

Figure 3: Classification accuracy for *Reuters-21578* for extractive summaries of various lengths.

text. This means that an eventual classification of such documents has to rely on **abstract categorization**, rather than traditional full-text categorization. The results obtained in the experiments reported here suggest that the task of text classification can be efficiently performed even when only abstracts are available for the documents to be classified.

Classification efficiency is also significantly improved when the classification is based on summaries, rather than full-length documents. A clear computational improvement was observed even for the relatively small test collections used in our experiments. For instance, the Naïve Bayes classifier takes 22 seconds to categorize the full-length documents in the *Reuters-21578* subset on a Pentium IV 3GHz 2GB computer, compared to only 6 seconds when the classification is done using 5-sentence summaries[5]

The results have also important implications on the problem of **text summarization**. As seen in Figures 3 and 4, regardless of the classifier, a performance peak is observed for summaries of 5-7 sentences, which give the highest increase in classification accuracy. This result can have an interesting interpretation in the context of text summarization, as it indicates the optimal number of sentences required for "grasping" the content of a text. From this perspective, the task of text classification can be regarded as an **objective** way of defining the "informativeness" of an abstract, and

[5]One could argue that the summarization-based classification implies an additional summarization overhead. Note however that the summarization process can be parallelized and needs to be performed only once offline. The resulting summaries can be then used in multiple classification tasks.

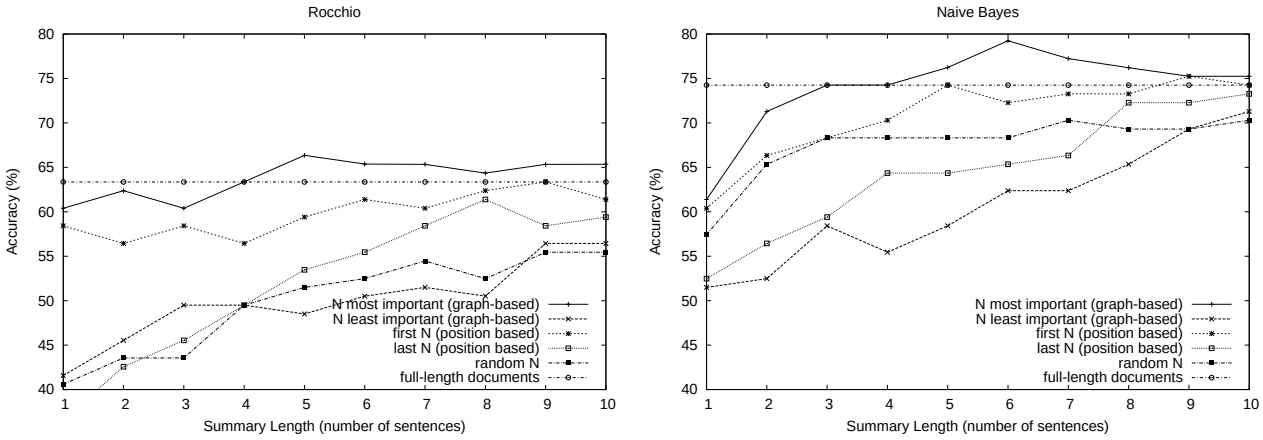

Figure 4: Classification accuracy for *WebKB* for extractive summaries of various lengths.

could be used as an alternative to more subjective human-assessed evaluations of summary content.

The comparative plots from Figures 3 and 4 also reveal another interesting aspect of the synergy between text summarization and document classification. Different automatic summarization tools – graph-based extractive summarization, heuristics that extract sentences based on their position in the text, or a simple random sentence selection baseline – have different but **consistent** impact on the quality of a text classifier, which suggests that text categorization can be used as an application-oriented evaluation testbed for automatic summarization. The graph-based summarization method gives better text classification accuracy as compared to the position-based heuristic, which in turns performs better than the simple sentence selection baselines. This comparative evaluation correlates with rankings of these methods previously reported in the literature (Erkan & Radev 04), which were based on human judgment or other automatic evaluation metrics such as ROUGE (Lin & Hovy 03).

5 Related Work

To our knowledge, the impact of content-based text summarization on the task of text categorization was not explored in previous studies. Sentence importance was considered as an additional factor for feature weighting in work reported in (Ko *et al.* 02), where words in a text were weighted differently based on the score associated with the sentence they belong to. Experiments with four different text categorization methods have shown that a weighting scheme based on sentence importance can significantly improve the classification performance. In (Kolcz *et al.* 01), text summarization is regarded as a feature selection method, and is shown to improve over alternative algorithms for feature selections. Finally, another related study is the polarity analysis through subjective summarization reported in (Pang & Lee 04), where the main goal was to distinguish between positive and negative movie reviews by first selecting those sentences likely to be more subjective according to a min-cut algorithm. The focus of their study was the analysis of text style (subjective versus objective), rather than classification of text content. We consider instead the more general text classification problem, combined with a typical text summarization task, and evaluate the role that text summaries can play in document categorization.

6 Conclusions

In this paper, we investigated the interaction between document summarization and text categorization, through comparative classification experiments relying on full-length documents or summaries of various lengths. First, we showed how graph-based algorithms can be successfully applied to the task of extractive summarization, resulting in state-of-the-art results as measured on standard data sets. Next, we showed that techniques for automatic summarization can be used to improve the performance of a text categorization task, with error rate reductions of up to 19.3% obtained with a Naïve Bayes or a Rocchio classifier when applied on short extractive summaries rather than full-length documents. Finally, we suggested that the interaction between text summarization and document categorization can be regarded as an application-oriented evaluation testbed for tools for automatic summarization, as summaries produced by different summarization tools were shown to have different impact on the performance of a text classifier.

References

(Apte *et al.* 94) C. Apte, F. Damerau, and S. M. Weiss. Towards language independent automated learning of text categorisation models. In *Proceedings of the 17th ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.

(Brin & Page 98) S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7), 1998.

(DUC 02) DUC. Document understanding conference 2002, 2002. http://www-nlpir.nist.gov/projects/duc/.

(Erkan & Radev 04) G. Erkan and D. Radev. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, July 2004.

(Grimmett & Stirzaker 89) G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, 1989.

(Hirao *et al.* 02) T. Hirao, Y. Sasaki, H. Isozaki, and E. Maeda. Ntt's text summarization system for duc-2002. In *Proceedings of the Document Understanding Conference 2002*, 2002.

(Hulth 03) A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Japan, August 2003.

(Joachims 97) T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, Nashville, US, 1997.

(Kamvar *et al.* 03) S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Extrapolation methods for accelerating PageRank computations. In *Proceedings of the 12th International World Wide Web Conference*, 2003.

(Kleinberg 99) J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

(Ko *et al.* 02) J. Ko, J. Park, and J. Seo. Automatic text categorization using the importance of sentences. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, Taipei, Taiwan, August 2002.

(Kolcz *et al.* 01) A. Kolcz, V. Prabakarmurthi, and J. Kalita. Summarization as feature selection for text categorization. In *Proceedings of the 10th International conference on Information and knowledge management*, Atlanta, Georgia, 2001.

(Lin & Hovy 03) C.Y. Lin and E.H. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of Human Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, May 2003.

(McCallum & Nigam 98) A. McCallum and K. Nigam. A comparison of event models for Naive Bayes text classification. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

(Mihalcea & Tarau 04) R. Mihalcea and P. Tarau. TextRank – bringing order into texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, Barcelona, Spain, 2004.

(Mihalcea *et al.* 04) R. Mihalcea, P. Tarau, and E. Figa. PageRank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20st International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland, 2004.

(Moschitti 03) A. Moschitti. A study on optimal paramter tuning for Rocchio text classifier. In *Proceedings of the European Conference on Information Retrieval*, Pisa, Italy, 2003.

(Ng *et al.* 97) H.T. Ng, W.B. Goh, and K.L. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia, PA, July 1997.

(Pang & Lee 04) B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July 2004.

(Raghavan & Garcia-Molina 03) S. Raghavan and H. Garcia-Molina. Representing Web graphs. In *Proceedings of the IEEE International Conference on Data Engineering*, March 2003.

(Rocchio 71) J. Rocchio. *Relevance feedback in information retrieval*. Prentice Hall, Ing. Englewood Cliffs, New Jersey, 1971.

(Schneider 04) K. Schneider. A new feature selection score for multinomial naive bayes text classification based on kl-divergence. In *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July 2004.

(Teufel & Moens 97) S. Teufel and M. Moens. Sentence extraction as a classification task. In *ACL/EACL workshop on "Intelligent and scalable Text summarization"*, pages 58–65, Madrid, Spain, 1997.

(Yang & Liu 99) Y. Yang and X. Liu. A reexamination of text categorization methods. In *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999.

(Yang & Pedersen 97) Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, Nashville, US, 1997.