Journal of Intelligent Learning Systems and Applications, **2011, 3, 131-138**
doi:10.4236/jilsa.2011.33015 Published Online August 2011 (http://www.scirp.org/journal/jilsa)

Insertion of Ontological Knowledge to Improve Automatic Summarization Extraction Methods

Jésus Antonio Motta, Laurence Capus, Nicole Tourigny

Département D'Informatique et de Génie Logiciel, Université Laval, Québec, Canada.
Email: jesus-a.motta.1@ulaval.ca, {laurence.capus, nicole.tourigny}@ift.ulaval.ca

Received May 30th, 2011; revised July 20th, 2011; accepted July 27th, 2011.

ABSTRACT

The vast availability of information sources has created a need for research on automatic summarization. Current methods perform either by extraction or abstraction. The extraction methods are interesting, because they are robust and independent of the language used. An extractive summary is obtained by selecting sentences of the original source based on information content. This selection can be automated using a classification function induced by a machine learning algorithm. This function classifies sentences into two groups: important or non-important. The important sentences then form the summary. But, the efficiency of this function directly depends on the used training set to induce it. This paper proposes an original way of optimizing this training set by inserting lexemes obtained from ontological knowledge bases. The training set optimized is reinforced by ontological knowledge. An experiment with four machine learning algorithms was made to validate this proposition. The improvement achieved is clearly significant for each of these algorithms.

Keywords: *Automatic Summarization, Ontology, Machine Learning, Extraction Method*

1. Introduction

Research works on automatic summarization have greatly increased in recent years. Indeed, digital sources of information have become increasingly available. When a user runs a query on Internet, she/he must choose among the retrieved documents those containing relevant information for her/him. The task becomes more difficult when the number of documents increase. An automated system able to 'discover' the essential information is one of the challenges of artificial intelligence, especially in natural language processing. In some cases, methods of machine learning based on symbols are used to tackle this problem.

Automatic summarization can be seen as a problem of transforming one or more documents in a shorter version with preserving information content [1]. The methods used are divided into two main approaches: extraction and abstraction, respectively surface methods and deep methods in a more linguistic viewpoint. A summary obtained by extraction is composed of a set of sentences selected from the source document(s) by using statistical or heuristic methods based on information entropy of sentences. The summarization process by extraction is a relevant alternative, robust and independent of language, compared with the summarization process by abstraction [2]. An abstractive summary is obtained by semantic analysis in order to interpret the source text, and find new concepts to generate a new text that will be the summary. This method requires linguistic processing at a certain level [3]. In addition, a summary can be produced in a generic way to give a general idea of the contents of documents to be summarized. It can also be based on keywords supplied by the user. In this case, it will contain the most relevant information related to these keywords [4]. Automatic summarization process by abstraction is usually decomposed into three steps: interpretation of source document(s) to obtain representation, transformation of this representation, and production of a textual synthesis [5]. Both approaches have their advantages and drawbacks. For this research, we are only interested in automatic summarization process by extraction and how to improve it.

The main problem of this kind automatic summarization by extraction lies in identifying the most important information of the document sources [2]. Different methods have been used until now with more or less successful results according to measurements based on recall (the number of correct sentences selected on the total number of correct sentences) and precision (the number of correct sentences on the total number of selected sentences) [6]. Some methods use an ontology or ontological knowledge to analyze terms and relations [7]. More recently, other methods have been reported using machine learning algorithms for determining descriptions of concepts. These methods build a training set divided into two subsets: important sentences and non-important sentences [1]. This training set is next used to induce a classification function from the concepts description. This function will serve to classify future sentences to produce new summaries. Generally in classification problems, the set of attributes is very large and entropic, with much noise and irrelevant attributes. The well-known underlying problem is named the curse of dimensionality [8]. Indeed, data too scattered do not facilitate a good estimate, nor obtain good classification models. This problem is actually tackled by using heuristic methods based on linear approximations, which optimize the training set by reducing it or constructing a new smaller set from another series of attributes [9]. The obtained results so far, even if they have progressed, could be further improved.

In this paper, we propose to optimize the training set in an original way. We insert lexemes of ontological knowledge bases into the training set to form a conceptual space, which will be used by the learning algorithm. Our hypothesis is that is possible to obtain a reinforced set, by using ontological knowledge to select or transform the characteristics of the set. We validated our hypothesis with four machine learning algorithms. We compared their performance by using various evaluation indicators. The obtained results showed that our solution improves the performance; it is then promising for the suite of this research. In Section 2, we will describe the solution proposed. In Section 3, we will present the conducted experimentation to validate our solution. In Section 4, we will conclude our paper by giving future work.

2. Insert Ontological Knowledge in Summary Extraction Process

Automatic summarization by extraction is a broad topic that uses different approaches, methods or techniques. It seems important at first to give our research framework, *i.e.* the process that we have considered and decided to improve. Then, we explain what we mean by the insertion of ontological knowledge and how this insertion fits into the summarization process. Finally, we give the evaluation methods that have allowed us to validate our hypothesis.

2.1. Summarization Process Considered

The different methods used for automatic summarization by extraction can be grouped into three approaches: statistical, enriched statistical and machine learning [5]. In this research work, we are interested especially in machine learning approaches because the results obtained are relevant and promising. The key item of these approaches lies on the choice of the training set and its optimization, which will be used to induce the classification function for summarizing futures documents in function of information content. More precisely, the sentences of the documents are represented by vectors, which constitute an initial matrix [10-12]. This matrix corresponds to the training set. The induced classification function enables to classify sentences into two classes: class 1 for important sentences and class –1 for non-important sentences. The summary will be then composed of the sentences of class 1. The crucial problem of this process is the fact that the sentences of this matrix are very entropic. It is necessary to optimize the matrix in order that it becomes an efficient training set.

Although many efforts have been made to improve the quality of summaries obtained, thus approaching those achieved by humans, there are still gaps in terms of accuracy and precision of results. Moreover, most of the summaries obtained are built from a single document. The most evident explanation is that the problems of redundancy increases along with the number of documents to be summarized.

The idea of our research work is then to propose a solution to better optimize the training set, *i.e.* the set of selected sentences forming the initial matrix needed for inducing the classification function. We wanted to find a solution more efficient, which do not need initially summaries already written to constitute the training set and can be applied on several documents to be summarized.

2.2. Insertion of Ontological Knowledge

Before inserting ontological knowledge, we identify the sentences of the document(s) to be summarized. Next, we apply a syntactic analysis and delete stop words. We then create a matrix E, formed of words by sentences. Each item of the matrix contains the value $tf \times idf$ of the word i in the sentence j. This discriminatory value is based on the Salton *et al.*'s formula [13], which evaluates the value of a term compared to a corpus of documents. We insert ontological knowledge to this matrix in order to obtain the new matrix E_0. Our hypothesis lies on the fact that the training set is reinforced by new information, *i.e.* terms or items with more semantic content and potentially discriminatory. Such an insertion also enables to solve partially the problem of synonymy [1], one of main open problems of information retrieval. Briefly, the initial matrix is improved by adding a set of sub-trees of hypernym and hyponym, for each word.

We elaborated an algorithm to insert ontological knowledge that enables to find conceptual structures from ontology, to compute their importance in an information viewpoint and introduce them in each term of the matrix. This algorithm gives lexemes in function of the words of sentences and inserts their semantic values to the matrix. The optimum lexemes have a factor that adds a semantic value to items of the matrix, improving its performance for classification. More precisely, the algorithm begins to do a search in the ontology by subject and verb. Next, it identifies the various concepts of each sentence by analyzing different sub-trees of parts of the sentence. The sub-trees are built in function of semantic relations of hypernym and hyponym. The algorithm evaluates the various sub-trees and chooses the best one. To finish, it inserts the selected sub-tree in the training set. When all new components of the training sets are inserted we have a new conceptual space enriched.

After inserting ontological knowledge, we do different steps to obtain the final training set and then induce the classification function. First, we filter entropic attributes with algebraic methods. To obtain our new set, we used a similarity transformation matrix, which enables to find smaller and less redundant subsets of attributes. By applying a transformation matrix to the matrices E and E_0, we identify principal components [8] and singular values [14] in order to reduce the entropy of the matrix and sort sentences in function their information content. The principal components of a matrix enable to identify groups of variables/words (principal component), greatly connected in the group, but without correlation between groups. The determining factor of this grouping is the variability, which represents the information or importance. We can then choose the first sentences, with the greatest variability, as important sentences and the last ones as non-important.

In detail, we represent the matrix E (for singular values for instance), by:

$$E = \left(u_1 u_2 \cdots u_n\right) \times \begin{pmatrix} s_1 0 & \mathrm{L} & 0 \\ \mathrm{M} & 0 & \mathrm{M} \\ 0 & \mathrm{L} & \mathrm{s}_n \end{pmatrix} \times \begin{pmatrix} v_1 \\ v_2 \\ \mathrm{M} \\ v_n \end{pmatrix}$$

$$= U \sum V^T = \operatorname{diag}\left(s_1 s_2 \cdots s_n\right)$$

Each value s_i corresponds to the variability of each sentence in all sentences. This variability is correlated to the information content of each sentence. We associate the values s_i, the greatest to the smallest, with the corresponding words of the matrix. Next, we label the sentences that contain high values s_i as important (class 1) and those that contain low values s_i as non-important (class –1). Finally, we use machine learning algorithm to induce the classification function, among those proposed by literature.

The induced classification function is able to differentiate sentences with much information (important) to sentences with insufficient information (non-important). So by applying the induced function on new documents, only the sentences containing the most important information are chosen to produce summaries.

2.3. Evaluation Method

To evaluate our solution and verify its efficacy, we created a contingency table, named confusion matrix. The inputs of this table correspond to considered classes, knowing that these values are given after applying the classification function to the training set. As the training set is a binary set, we obtain **Table 1** elaborated in function of the two classes to be determined: important and non-important.

When the process of classification is finished, we identify four categories of sentences among all those analyzed:

- TP: if the function predicts correctly a sentence labeled as important;
- TN: if the function predicts correctly a sentence labeled as non-important;
- FP: if the function predicts incorrectly a sentence labeled as important;
- FN: if the function predicts incorrectly a sentence labeled as non-important.

We used the information given by **Table 1** to obtain the values of three evaluation indicators known in automatic summarization, that are recall, precision and F-score, as well as ROC curves (Receiver Operation Characteristic) [15].

Recall (R) is the number of predictions TP divided by the true number of positive instances classified as positives [6]. It informs about the capability of the classification function to identify a sentence as important when it is really important. The following formula enables to compute recall:

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Precision (P) corresponds to the number of predictions TP divided by the total number of instances classified as

Table 1. Confusion matrix used to evaluate efficacy.

Class	**Predicted Class**	
	Important	*Non-important*
Important	True Positive Case (TP)	False Negative Case (FN)
Non-important	False Positive Case (FP)	True Negative Case (TN)

positives [6]. This indicator informs about the capability of the function to classify correctly a sentence according to all the sentences added to this category. It is computed with the following formula:

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

F_{score} corresponds to a harmonic average of recall and precision [6] and it is defined by:

$$F_{score} = 2 \times \frac{\mathrm{R} \times \mathrm{P}}{\mathrm{R} + \mathrm{P}}$$

We also represented our results by using ROC curves. These graphs enable to represent all the pairs of values TPR (True Positive Rate) or *sensitivity* and FPR (False Positive Rate) or 1-specificity, resulting of a continuous variation of the observation points in the whole row of observed results [15]. By simple observation of these graphs, we obtain a qualitative comparison. When we apply each model to be evaluated on the training set, the curve placed on the top and to the left has the greatest accuracy. Likewise, the area under the curve indicates the success probability of the model by identifying a sentence as important. The ROC curves then give indications on the accuracy of the classification model, as well as a unified criterion in the evaluation process. The mentioned values are obtained by the following formulas:

$$\text{sensitivy}(\mathrm{TPR}) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

$$\text{specificity}(\mathrm{TNR}) = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{TP}} = 1 - \mathrm{FPR}$$
$$\rightarrow \mathrm{FPR} = 1 - \text{specificity}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

Recall, Precision and F_{score} as well as ROC curves allowed us to evaluate the improvement rates obtained by our solution

3. Experiment and Results

The experiment is conducted on a set of documents from Reuters Corpus [16], a news database that contains approximately 11000 documents, classified into 90 currents events subjects and grouped into two sets, respectively named training and test. Each document contains on average 120 words and 15 sentences. We chose a total number of 2000 documents for our experiment.

For the extraction of ontological knowledge, we used WordNet database [17], developed by Princeton university. This is a database oriented semantically with a very rich frame, greatly used in computational linguistic. It is composed of words related to names, verbs, adjectives and adverbs. Words are organized into sets of synonyms named synsets, related by semantic relations of hypernym, hyponym, meronymy and holonomy. WordNet database thus contains 155287 words and 117659 synsets.

We also chose four machine learning algorithms greatly used. The first algorithm is named Support Vector Machine [18]. It builds a hyper-plane in an *n*-dimensions space to classification, regression or other tasks. Intuitively, a good separation between classes is obtained when a hyper-plane has the greatest distance for all the nearest points of the training set. The second algorithm is a probabilistic classifier based on Bayes' theorem, Bayesian Classifier or Naïve Bayes [9], but with a great independence hypothesis. In other words, it assumes that the presence or absence of a characteristic is not related to the presence or absence of another characteristic. The third algorithm, *Random Tree* [19], realizes its classification by building a tree in which the total number of selected nodes is randomly chosen while being equal to:

$$\log_2(\text{attribute\_number} + 1)$$

Finally, the fourth algorithm is Multilayer Perceptron [20]. This is an artificial neuronal network with many layers. The activation function of each neuron is not linear. This neuronal network can be used to identify linearly inseparable classes. The function is learned from multilayers that are totally connected to each other.

The experiment conducted is composed of two parts. In the first part, we produced summaries by extraction from the chosen corpus without inserting ontological knowledge to the machine learning algorithm. We evaluated the obtained results for each algorithm in function of the evaluation methods given. In the second part, we produced summaries from the same corpus, this way by inserting ontological knowledge. We also evaluated the results obtained. By following, we present these results and discuss them.

3.1. Results for Recall, Precision and F_{score}

We used methods of random subsampling (1/3 for the test set and 2/3 for the training set) and cross-validation (10 crossings). We also based on Tanagra software of Lyon University [21], Weka software of Waikato University [22] and Orange software of Ljubljana University [23]. The two methods used gave similar results.

Tables 2 and **3** present the values of recall, precision and F_{score} as well as the confusion matrix for each of the four algorithms evaluated. In **Table 2**, the algorithms were applied to the training set obtained from principal components of the word matrix, while, in **Table 3**, the

Table 2. Predictions and confusion matrix with principal components.

Algorithm	Without ontological knowledge					With ontological knowledge				
	Prediction			Confusion matrix		Prediction			Confusion matrix	
	Recall	Precision	F_{score}	Important	Non-Important	Recall	Precision	F_{score}	Important	Non-important
Naïve Bayes			0.993					0.978		
Important	1.00	0.986		488	0	0.994	0.964		480	18
Non-important	0.837	1.00		7	36	0.818	0.964		18	81
SVM			1.00					0.999		
Important	1.00	1.00		488	0	1.00	0.998		483	0
Non-important	1.00	1.00		0	43	0.999	1.00		1	98
Random Tree			0.982					0.9521		
Important	1.00	0.972		488	0	0.988	0.919		472	6
Non-important	0.674	1.00		14	29	0.579	0.905		42	57
ML Perceptron			0.993					0.983		
Important	0.998	0.988		487	1	0.992	0.974		479	4
Non-important	0.861	0.974		6	37	0.869	0.956		13	86

Table 3. Predictions and confusion matrix with singular values.

Algorithm	Without ontological knowledge					With ontological knowledge				
	Prediction			Confusion matrix		Prediction			Confusion matrix	
	Recall	Precision	F_{score}	Important	Non-important	Recall	Precision	F_{score}	Important	Non-important
Naïve Bayes			0.997					0.958		
Important	1.00	0.994		486	0	0.998	0.956		483	1
Non-important	0.858	1.00		3	18	0.776	.987		22	76
SVM			1.00					0.999		
Important	1.00	1.00		486	0	1.00	0.998		484	0
Non-important	1.00	1.00		0	21	1.0	1.00		1	97
Random Tree			0.992					0.957		
Important	1.00	0.984		486	0	1.0	0.917		484	0
Non-important	0.61	1.00		8	13	0.551	1.0		44	54
ML Perceptron			0.992					0.980		
Important	1.00	0.984		486	0	0.971	0.999		470	14
Non-important	0.619	1.00		8	13	0.949	0.869		5	93

training set was obtained from singular values. The values are given for each part of the experiment, *i.e.* before inserting ontological knowledge and after inserting it.

By observing **Tables 2** and **3**, we note that performances in terms of recall and precision are high. Naïve Bayes algorithm presents the greatest performance in the case of using principal components. Its performance is followed by the two other algorithms Support Vector Machine and MultiLayer Perceptron. When ontological knowledge is inserted then the greatest performance is those of Support Vector Machine algorithm followed by those of MultiLayer Perceptron algorithm. In the case where the space was based on singular values, this is Support Vector Machine algorithm that obtains the greatest performance, followed by those of Naïve Bayes algorithm. When the ontological knowledge is inserted, Support Vector Machine algorithm continues to occupy the first place. The performance of MultiLayer Perceptron algorithm is improved at the expense of those of Naïve Bayes algorithm.

3.2. Results for ROC Curves

In **Table 4**, we observe the AUC values (Area Under Curve) of each ROC curves, for each algorithm applied at a training set obtained from principal components. The results are given before and after inserting ontological knowledge. The last column indicates the relative improvement obtained.

Table 5 gives the same values, but this way by considering that the training set is obtained from singular values.

From the joint observation of **Tables 4** and **5**, we can say that the introduction of ontological knowledge in the training set, obtained using principal components or singular values, increases the quality of all algorithms. For instance, we note an improvement of 58.8% and 19.1% for respectively MultiLayer Perceptron and Random Tree algorithms when using principal components. Support Vector Machine and Naïve Bayes algorithms are improved of 14.8% and 10.33% respectively, when using singular values. The algorithm with the best performance is Support Vector Machine followed by Naïve Bayes, when using principal components. In the case of singular values are used, we also observe a very great improvement of the algorithm quality when the ontological knowledge is inserted. MultiLayer Perceptron algorithm obtains again the highest value with 27%. Support Vector Machine and Naïve Bayes algorithms occupy respecttively the second and the third places with 20.5% and 20.1%. The best quality algorithm is Naïve Bayes followed by Support Vector Machine.

Table 6 shows a comparison of AUC values of each algorithm after inserting ontological knowledge, depending on whether the training sets are obtained from principal components or singular values.

Table 6 enables to estimate the difference of improvement between a space based on principal components and another based on singular values. First, we observe that all algorithms studied improve their qualitative performance moving from a space of principal components to another of singular values. The greatest improvement is obtained by *Random Tree* algorithm with 12% followed by Naïve Bayes with 11.3%. From this table, we also conclude that the two best algorithms and the most promising to extract summaries with spaces ontologically reinforced, among the four ones studied, are Naïve Bayes and Support Vector Machine.

Figures 1 and **2** correspond to ROC curves obtained for each algorithm with training sets produced from principal components and singular values. In these two cases, the results are given without inserting ontological knowledge (1a), and with inserting ontological knowledge (1b).

As we already mentioned, the ROC curves offer a way of evaluating the quality of a classification algorithm in function of its capability to give good predictions. In our case, a good prediction corresponds to sentences classified as important, that should take part of summary, and discriminated to non-important sentences.

In addition to the information obtained by **Tables 4-6**, careful observation of figures enables to identify the optimum points of each algorithm, simply by placing the point at the highest position to the left. We also compare the accuracy between algorithms. For instance, if we want to compare Support Vector Machine algorithm when it reaches a little more than 90% of TP cases face to *Random Tree* algorithm then we see on the **Figure 2(b)** that Support Vector Machine algorithm has an appro-

Table 4. Improvement when using principal components.

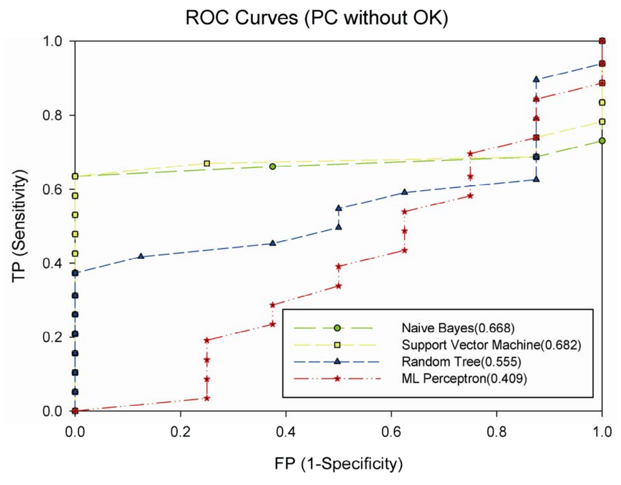
Algorithm	Values of ROC curves		Improvement (%)
	Without ontological knowledge	With ontological knowledge	
Naïve Bayes	0.668	0.737	10.33
SVM	0.682	0.783	14.8
Random Tree	0.555	0.661	19.1
ML Perceptron	0.449	0.713	58.8

Table 5. Improvement when using singular values.

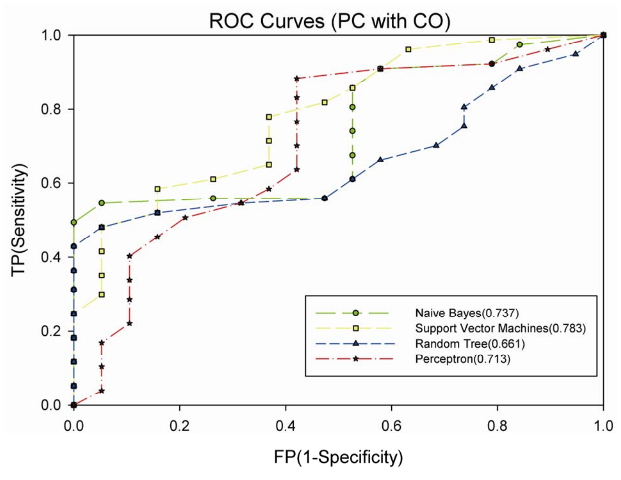
Algorithm	Values of ROC curves		Improvement (%)
	Without ontological knowledge	With ontological knowledge	
Naïve Bayes	0.683	0.820	20.06
SVM	0.673	0.811	20.51
Random Tree	0.697	0.740	6.17
ML Perceptron	0.589	0.748	26.99

Table 6. Difference of improvement between using principal components and using singular values.

Algorithm	Values of ROC curves		Improvement (%)
	Principal components	Singular values	
Naïve Bayes	0.737	0.820	11.30
SVM	0.783	0.811	3.60
Random Tree	0.697	0.740	12.00
ML Perceptron	0.589	0.748	4.90

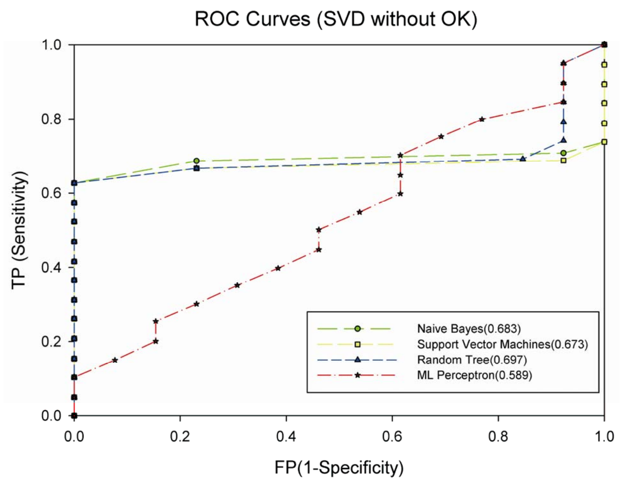


(a)

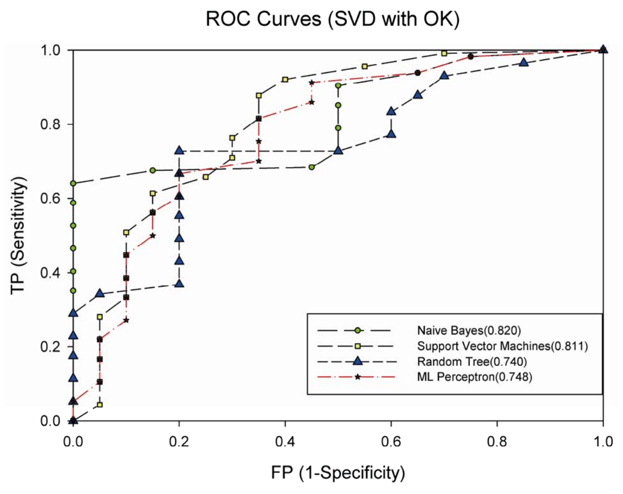


(b)

Figure 1. (a) Principal components before inserting ontological knowledge; (b) Principal components after inserting ontological knowledge.



(a)



(b)

Figure 2. (a) Singular values before inserting ontological knowledge; (b) Singular values after inserting ontological knowledge.

ximate FP rate of 37% and Random Tree algorithm has a rate of 70%.

The obtained results show a significantly improvement to the classification function after inserting ontological knowledge, whatever the machine learning algorithm used. More, these results give the two best algorithms. The Naïve Bayes and Support Vector Machine algorithms should be then applied to automatic summarization with a training set produced from singular values and reinforced by ontological knowledge.

4. Conclusions

There are still great opportunities for deepening and development research to find suitable methods for summarizing. In this paper, we studied the behaviour of four machine learning algorithms that induce classification function from training sets. These sets were reinforced by inserting ontological knowledge and used to discriminate the important sentences, from one or several documents, of those which are not. The used algorithms are Naïve Bayes, Support Vector Machine, Random Tree and Multilayer Perceptron. By analyzing the results of experimentation, we concluded that all considered algorithms may be used to produce summaries. We also note that using principal components or singular values to select the training set may be successfully retained to induce the learning functions of the four studied algorithms. The insertion of ontological knowledge gives qualitative improvements of performance remarkable. This insertion enables to propose good classification functions, which are able to discriminate sentences between important and non-important. The sentences discriminated as important constitute the future summary. Likewise, we observe that ontological knowledge produces more great effects on the classifier quality, if the training set is obtained from singular values rather than from principal components. From this final analysis, we conclude that the two best algorithms that should be applied to automatic summarization by extraction are Naïve Bayes and Support Vector Machine, from a set of singular values reinforced by ontological knowledge.

As future work, we think that it would be interesting to evaluate the performance of classification algorithms on more reduced spaces, *i.e.* optimized by means of techniques different to those used in this experimentation. It would be also interesting to explore their behaviour on sets reinforced by ontological knowledge.

5. Acknowledgements

The authors would thank Natural Sciences and Engineering Research Council of Canada (NSERC) for its financial support.

6. References

[1] A. Sharan and H. Imran, "Machine Learning Approach for Automatic Document Summarization," *Proceedings of World Academy of Science, Engineering and Technology*, 2009, pp. 103-109.

[2] R. A. García-Hernandez, R. Montiel, Y. Ledeneva, E. Rendón, A. Gelbukh and R. Cruz, "Text Summarization by Sentence Extraction Using Unsupervised Learning," *Proceedings of the 7th Mexican International Conference on Artificial Intelligence*: *Advances in Artificial Intelligence*, 2008, pp. 133-143.

[3] I. Mani and E. Bloedorn, "Machine Learning of Generic and User-Focused-Summarization," *Proceedings of the Tenth Conference on Innovative Applications of Artificial Intelligence*, Menlo Park, 1998, pp. 821-826.

[4] J. Goldstein, "Evaluating and generating summaries using normalized probabilities," *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, 1999, pp. 121-128. doi:10.1145/312624.312665

[5] K. S. Jones, "Automatic Summarising: The State of Art," *Information Processing and Management*, Vol. 43, No. 6, 2007, pp. 1449-1481. doi:10.1016/j.ipm.2007.03.009

[6] R. R. Korfhage, "Information Storage and Retrieval," Wiley, New York, 1997.

[7] L. Hennig, W. Umbrath and R. Wetzker, "An Ontology-Based Approach to Text Summarization," *IEEE/WIC /ACM Proceedings of the International Conference on Web Intelligence and Intelligent Agent Technology Workshops*, 2008, pp. 291-294.

[8] R. Bellman, "Introduction to Matrix Analysis," McGraw-Hill, New York, 1997.

[9] M. Steinbach, "Introduction to Data Mining", Pearson Education, Boston, 2006.

[10] M. Ikonomakis, S. Kotsiantis and V. Tampakas, "Text Classification Using Machine Learning Techniques," *Proceedings of the 9th WSEAS International Conference on Computers*, Stevens Point, 2005, pp. 966-974.

[11] I. Mani, "Recent Development in Text Summarization," *Proceedings of the Tenth International Conference on Information and Knowledge Management*, McLean, 2001, pp. 529-531.

[12] H. Xuexian, "Accuracy Improvement of Automatic Text Classification Based in Feature Transformation and Multi-classifier Combination," *Proceedings of AWCC*'2004, Zhenjiang, 2004, pp. 463-464.

[13] G. Salton and C. Buckley, "Term-Weighting Approaches in Automatic Text Retrieval," *Information Processing and Management*, Vol. 24, No. 5, 1988, pp. 513-523. doi:10.1016/0306-4573(88)90021-0

[14] G. H. Golub, "Calculing the Singular Values and Pseudo-Inverse of a Matrix," *Journal of the Society for Industrial and Applied Mathematics*, Vol. 2, No. 2, 1965, pp. 205-224. doi:10.1137/0702016

[15] T. A. Lasko, J. G. Bhagwat, K. H. Zou and L. Ohno-Machado, "The Use of Receiver Operating Characteristic Curves in Biomedical Informatics," *Journal of Biomedical Informatics*, Vol. 38, No. 5, 2005, pp. 404-415. doi:10.1016/j.jbi.2005.02.008

[16] A. Saleh, "Reuters Corpus (Offered by Reuters News Agency)," 2004. http://about.reuters.com/researchandstandards/corpus/

[17] C. D. Fellbaum, "WordNet (A Lexical Database for English)," Princeton University, 1985. http://wordnet.princeton.edu//

[18] V. Vapnik, "The Nature of Statistical Learning Theory," Springer-Verlag, New York, 1995.

[19] R. Bellman, "Algorithms, Graphs and Computers", Academic Press, New York, 1970.

[20] F. Rosenblatt, "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms," Spartan Books, Washington DC, 1961.

[21] R. Rakotomalala, "Tanagra: Un Logiciel Gratuit Pour L'enseignement et la Recherche," *Proceedings of the EGC*'2005 *Conference*, Amsterdam, 2005, pp. 697-702.

[22] G. Holmes, A. Donkin and I. H. Witten, "Weka (A Software Developed by Machine Learning Group)," Universty of Waikato, 1994. http://www.cs.waikato.ac.nz/ml/weka/

[23] J. Demzar and B. Zupan, "Orange (A Software Developed at Laboratory of Artificial Intelligence)," Faculty of Computer and Information Science, University of Ljubljana, 2010. http://orange.biolab.si/