DESIGN AND IMPLEMENTATION OF AN AUTOMATIC DOCUMENT CLASSIFICATION SYSTEM

Jie-Mein, Goh and Danny C.C. Poo
Department of Information Systems
School of Computing
National University of Singapore
Lower Kent Ridge Road, Singapore 119260
{ gohjm, dpoo}@comp.nus.edu.sg

ABSTRACT

With the explosive growth of information, it is an increasingly arduous task to locate and organize information. Document classification is thus an important component of Knowledge Management Systems as it helps users to manage and search information in enterprises more effectively. This paper describes the design and implementation of the initial prototype of an automatic document classification system to help users classify documents automatically that can be built on top of existing systems.

KEY WORDS

Software Tools, Artificial Intelligence Applications, Software Design, Classification

1. INTRODUCTION

Knowledge management systems have been created and applied to help organizations systematically manage and leverage on the stores of knowledge. A company's success relies on the availability of information at the right time and at the right place.

Document classification is an essential component of knowledge management systems. Document classification refers to the activity where documents are grouped according to their content and use [1] according to some predefined systematic arrangement or categories [2]. This systematic arrangement is usually referred as the knowledge taxonomy or enterprise taxonomy in enterprises.

Document classification according to the enterprise taxonomy is used to assist people by supporting the following activities:

1. Browsing – A user may browse through categories of information to find expected and unexpected information that he or she may be interested in. Documents classified according to various categories in a taxonomy provides a structure to the process of browsing.
2. Searching – Users with different degrees of understanding on a specific subject may be assisted when documents are classified according to the enterprise taxonomy. The response time to the search may be faster as the number of documents searched is reduced considerably.
3. Managing – A user is able to better manage his data in his system and recommend the locations to store newly created files.

In order to classify document without increasing the burden on knowledge workers, a new role of the knowledge librarian has been created [3]. In a corporate environment, knowledge taxonomies are created by knowledge librarians to aid organization of knowledge to achieve better maintenance and dissemination of knowledge.

The knowledge librarian is also responsible for transferring content into the knowledge repository by tagging user submissions with appropriate keywords or metadata and then categorizing the knowledge into the repository.

With the exponential increase in the volume of information on the World Wide Web and intranets, it is increasingly difficult to find and organize information. The creation of new roles for document classification is no longer cost-effective and the intervention of layman results in inconsistent and unreliable retrievals. Therefore, there is a need to create a tool to automatically classify documents using expert knowledge.

In our work, we examine automatic document classification methods to supplement or replace human efforts in organizing knowledge as the growth of information makes them unmanageable. We address this issue by employing machine learning classifiers for automatic document classification. The advantage of this approach is the accuracy achieved by machine learning techniques is comparable to that of human experts [4].

Our objective is to develop a system that will categorize documents without human intervention and utilizes the cataloguer's expert knowledge. Our system captures expert's knowledge by employing a machine learning technique called support vector machines (SVM) and uses the learned model to classify new documents.

This paper is organized as follows. In the next section, we first review related work. Following this, we explain our design objectives and describe a generic design of automatic document classification system that uses machine learning. Finally, we describe an initial prototype that we have developed which employs the design.

1.1 Previous Work

There have been several projects that attempt to apply various techniques for document classification in automated services. A wide range of information retrieval, knowledge base and machine learning techniques have been employed.

Closer to our work are the systems that have applied machine learning techniques to text categorization. There are several attempts to apply various machine learning techniques which include regression models [5-7], neural networks [5, 8-9], probabilistic Bayesian models [10], nearest neighbor classifiers [7], decision trees [11], symbolic rule learning [12-14] and support vector machines (SVM) [15-17].

Some efforts have employed regression models. Experiments conducted by Schutze [5], Fuhr [6] and Yang [7] have shown that using regression models for automatic document classification is effective. However, the disadvantage of using regression model is the high computing cost involved as compared to other techniques. Projects that have used neural networks for automatic text categorization have shown that non-linear neural networks [5] do not have much improvement over linear neural networks [8]. Similarly most of the rest of the machine learning methods have proved to be effective for classification.

More recent efforts on SVM have shown a leap in improvement of classification effectiveness. SVM has been introduced by Joachims [17] and subsequently used by Dumais et.al [15], Dumais and Chen [16] whose projects have shown that this technique was superior to other machine learning techniques. Hence, we have adopted SVMs in our prototype.

Although support vector machines have been used by Dumais [15,16], the major difference between her approach and ours is that in our system, nouns and noun phrases were extracted as features whereas they used an information gain metric to select features. Although SVM have shown to be efficient and effective for classification, it has not been previously explored with nouns and noun phrases as features. A series of experiments by Yang and Pederson [18] has shown that their metric has yielded poor performance. Thus, we have used a different set of features. Both document frequency and natural language processing techniques were employed to obtain features from documents in our initial prototype. We are interested to find out if using natural language processed features with SVM will improve results as it has been observed that nouns and noun phrases indicating objects and subjects of a document are likely to be the most meaningful.

2. DESIGN OBJECTIVES

There are 4 major design considerations of our system:

1. The system design must be independent of the user environment and interface.
2. The system design must be independent of the features and the learning method being used.
3. The system design should be independent of the document database that is being used.
4. The system design should allow any type of document format to be used.

With these objectives in mind, our design is kept as generic as possible with the ultimate goal of plugging the system into any document management system.

3. SYSTEM DESIGN AND IMPLEMENTATION

The overall design (Figure 1) of the system comprises of the following components:

1. The dataset grabber module connects to the document database that had been classified manually by human experts. These documents will be used as the dataset to train or test the classifier.
2. Feature engine module processes the dataset obtained by the dataset grabber module in 1.
3. The classifier module builds a learning classifier based on the documents that have processed by the feature engine.
4. The Graphical User Interface module handles the interactions between the user and the system.

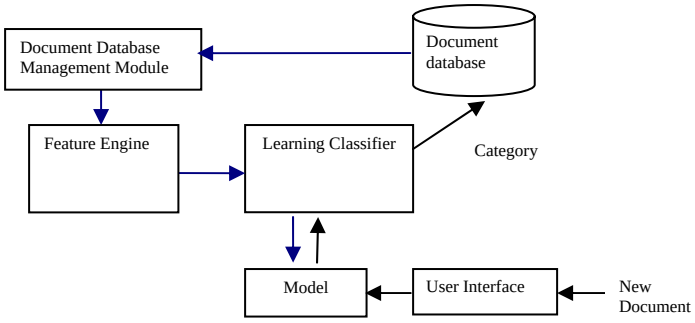

Figure 1. Generic Design

In the next few sections, we will discuss the architectural overview of our prototype system that adopts the above generic design. The graphical user interface of this prototype system has been implemented in Java supported by swing package.

3.1 Document Database Management Module

The document database management module contains functions connecting the system to the World Wide Web.

With the design objective of keeping the system independent of the document database that is being used and the applicability of any type of document format, the document database management module has two sub-components, the document handler and the filter. The document handler basically provides methods to retrieve or store classified or unclassified documents. For example, in many organizations, documents reside in a common repository. In such a case, the document handler basically contains methods to retrieve documents and store newly classified documents into the repository. The filter on the other hand, gets the document from the document handler to preprocess the documents into text format so that only plain text is handled by the feature engine module.

The document handler in our prototype as shown in Figure 2 actually performs the functions of a spider. It works by connecting to the hyperlink of a seed page in a web directory of a certain category that contains link to other web pages not belonging to the web directory such as Yahoo (http://www.yahoo.com) and Looksmart (http://www.looksmart.com). For our experiments, we used a collection of domain specific pages from LookSmart's web directory. The documents are then downloaded into the local computer with each directory as a category. In other words, documents belonging to the same category will reside in the same directory.

Filter objects are also created and added when the document database management module is instantiated. The filter comprises of the links extractor, script remover and the tags remover. The links extractor will remove irrelevant links to sieve out only documents that belongs to that category as classified by the web directory. After this, the script remover of the filter will remove the java scripts in the html documents. Finally, the tags remover will sieve out the tags leaving only plain text.

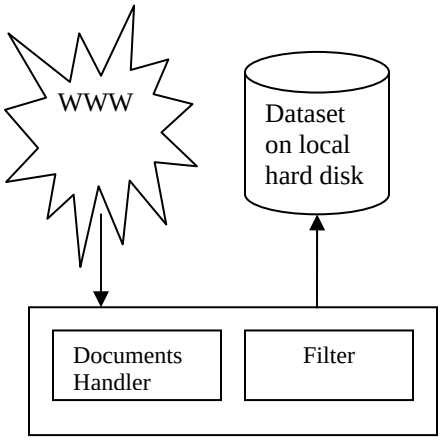

Figure 2. Document Database Management Module

3.2 Feature Engine Module

The feature engine module constructs the features from a document file in any format. This system design allows users to test different types of feature engineering methods. In our prototype, two feature extraction techniques are built into the feature engine module so that we can compare their performance. The first technique uses each term in the documents of a training corpus as a feature which is also known as the bag of words technique. Therefore, it is not difficult to see why this technique often result in a great number of attributes in the learning classifiers. In order to reduce the number of features, we have implemented a simplistic natural language processing technique. Our objective is to use as few features and preserve as much information of the content as far as possible. Studies have shown that phrases indicating the objects and subjects of a text are likely to improve search results and are useful in information retrieval.

Figure 3 shows the design of the feature engine module for the natural language processing subcomponent to extract nouns and noun phrases. It comprises of three major parts. The first is a tokenizer component that reads in each sentence and tokenizes the sentence to produce tokens. The tokens or words are then passed to the tagger module which attaches a part-of-speech tag to every word in the documents. The nouns and noun phrase analyzer filters the tagged words to obtain only the nouns and noun-phrases. No further stemming was done and these selected terms are then assembled into a global list.

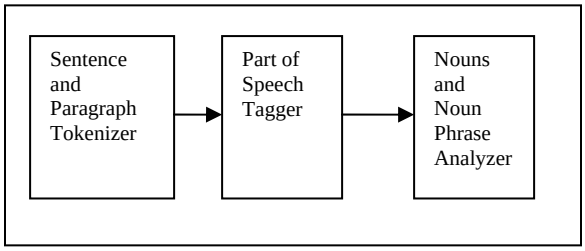

Figure 3. Feature Engine Module Components

3.3. Classifier Module

The classifier module contains methods that build the learning classifier. There are many learning algorithms that can be implemented in the classifier module. In our prototype, this module contains methods that create support vector machines. A support vector machine algorithm was used as the classifier as previous studies have shown that this technique is effective and relatively fast for text classification problems.

A linear SVM is a hyperplane that separates a set of positive examples from a set of negative examples with maximum distance from the hyperplane to the nearest of

the positive and negative examples. Figure 4 shows the graphical representation of a linear SVM. More details can be found in [20].

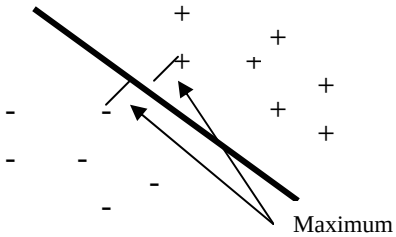

Figure 4. Linear SVM

We used the Platt's [21] sequential minimal optimization algorithm to process the linear SVM more efficiently. This algorithm breaks the large quadratic programming problem into smaller sub-problems.

The feature engine module passes the features to be learned from the training corpus and the model for each category that has been built to the classifier module. When new documents are to be classified, the learned weights are then used to find the most relevant category by computing the dot product of the vector of learned weights and the input vector for the new document.

3.4 User Interface Module

We have developed the user interface of the system in two modes: command line and graphical user interface using Swing. Here we will describe our simplistic graphical user interface briefly. Our graphical interface consists of the following interface windows:

1. The build model interface allows users to enter the web directory that they would like to collect documents as their dataset (Figure 5). These documents are then stored locally on the hard disk and used as the training corpus to obtain a classifier model.
2. The document input window allows the user to classify a new document which can either be a web document or a file residing on the local hard disk (Figure 6). To generate the category, the user clicks on the run button. Once the classifier completes the processing, the results window appears.
3. The results window that outputs the category of the new document onto the window (Figure 7).

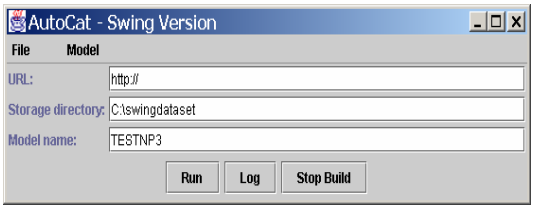

Figure 5. Build Model Interface

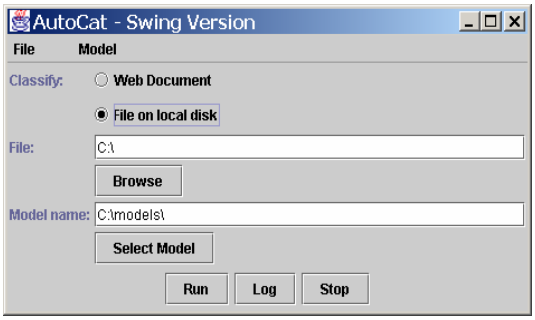

Figure 6. Document Input Window

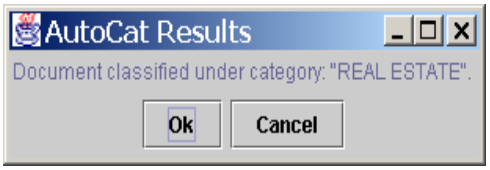

Figure 7. Results Window

4. CONCLUSION

We have discussed the design of the system that is independent of the feature processing and machine learning classifier that is being used. This paper first described a generic design of an automatic document classification system followed by an application of the design. We have shown how we applied the design when building a prototype of an automatic document classification system.

This system provides an intelligent automatic document classification service that can be integrated into knowledge management systems to help to automatically categorize documents based on support vector machines with natural language processing techniques. Further experiments will be carried out on the effectiveness of the automatic document classification system using a reduced set of features.

REFERENCES

1. Smith III, D. Milburn, *Information and Records Management: A Decision-Maker's Guide to Systems Planning and Implementation* (Quorom Books, 1986).

2. Gill, Suzanne, *File Management and Information Retrieval Systems* (USA:Libraries Unlimited, 1986).

3. J. Hahn and M. R. Subramani, A Framework of Knowledge Management Systems: Issues and Challenges for Theory and Practice, *Proceedings of the 21st International Conference on Information Systems*, Brisbane, 2000, 302-312.

4. F. Sebastiani, Machine Learning in Automated Text Categorization. *ACM Computing Surveys, 34(1)*, 1- 47, 2002.

5. H. Schütze, D. Hull, and J.O. Pedersen, A Comparison of classifiers and document representations for the routing problem: *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*, 229-237, 1995.

6. N. Fuhr, S. Harmanna, G. Lustig, M. Schwantner, and K. Tzeras, Air/X-A rule-based multi-stage indexing system for large subject fields: *Proceedings of RIAO'91*, 606-623, 1991.

7. Y. Yang and Y. Lui, A re-examination of text categorization methods: *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, 42-49, 1999.

8. H.T Ng, W.B. Goh and K.L. Low, 1997. Feature Selection, peceptron learning and a usability case study for text categorization: *Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval*, Philadelphia, P.A., 1997, 67-73.

9. Y. Yang, Expert network: Effective and efficient learning from human decisions in text categorization and retrieval: *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, 13-22, 1994.

10. D. Koller and M. Sahami, 1997. Hierarchically classifying documents using very few words: *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, 170-178, 1997

11. D.D Lewis and M. Ringuette, A comparison of two learning algorithms for text categorization: *Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, 81-93, 1994.

12. C. Apte, F. Damerau and S. Weiss, Automated learning of decision rules for text categorization, ACM Transactions on Information Systems, 12(3), 233-251, 1994.

13. W.W. Cohen and Y. Singer, Context-sensitive learning methods for text categorization: *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96)*, 307-315, 1996.

14. H. Li, and K. Yamanishi, Text Classification using ESC-based stochastic decision-lists. *Proceedings of the 8th ACM International Conference on Information and Knowledge Management (CIKM'99)*, 122-130, 1999

15. S. T. Dumais, J. Platt, D. Heckerman and M. Sahami, Inductive learning algorithms and representations for text categorization: *Proceedings of the Seventh International Conference on Information and Knowledge Management (CIKM'98)*, 148-155, 1998.

16. S. Dumais, and H. Chen, Hierarchical Classification of Web Content: *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, 148-155, 1998.

17. T. Joachims, Text categorization with support vector machines: Learning with many relevant features. *Proceedings of European Conference on MachineLearning (ECML'98)*, 1998.

18. Y Yang, and J.O. Pederson, A comparative study on feature selection in text categorization: *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*, 412-420, 1997.

19. S. Scott and S. Matwin, Feature Engineering for Text Classification: *Sixteenth International Conference on Machine Learning (ICML'99)*, Bled, Slovenia, June 27-29, 1999.

20. V. Vapnik, *The Nature of Statistical Learning Theory* (Springer-Verlag, 1995).

21. J. Platt, *Fast training of support vector machines using sequential minimal optimization. In Advances in Kernel Methods – Support Vector Learning* (MIT Press, 1999).