Feature Selection with Rough Sets for Web Page Classification

Aijun An[1], Yanhui Huang[2], Xiangji Huang[1], and Nick Cercone[3]

[1] York University, Toronto, Ontario, M3J 1P3, Canada
[2] University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada
[3] Dalhousie University, Halifax, Nova Scotia, B3H 1W5, Canada

Abstract. Web page classification is the problem of assigning predefined categories to web pages. A challenge in web page classification is how to deal with the high dimensionality of the feature space. We present a feature reduction method based on the rough set theory and investigate the effectiveness of the rough set feature selection method on web page classification. Our experiments indicate that rough set feature selection can improve the predictive performance when the original feature set for representing web pages is large.

1 Introduction

With the rapid growth of information on the World Wide Web, automatic classification of web pages has become important for effective indexing and retrieval of web documents. One approach to automatic web page classification is to apply machine learning techniques to pre-classified web data to induce profiles of categories and compare the profiles of categories with the representation of a given document in order to classify the document. A major characteristic, or difficulty, of this application is the high dimensionality of the feature space. A common approach to representing a text document is to use a "bag of words" that appear in the document. Since a web page can contain thousands of words, the feature space for representing web pages is potentially huge. Few machine learning systems can handle such a large number of features. In addition, too many features may present noise to the learning system. Therefore, it is highly desirable to reduce the feature space in order to make use of existing learning systems, to improve classification accuracy, and to speed up the learning process. It is also desirable to achieve such a goal automatically, i.e., no manual selection or construction of features is required.

Automatic feature selection methods have been used in text classification. Lewis and Ringuette [6] used an information gain measure to reduce the document vocabulary in naive Bayes classification and decision tree learning. Yang [15] used the principal component analysis to find orthogonal dimensions in the vector space of documents. Wiener *et al* [14] used mutual information and a χ^2 statistic to select features for input to neural networks. Lang [4] used a minimum description length principle to select terms for news categorization.

It has been asserted that feature selection is the most critical stage of the learning process in text classification [6].

We investigate the effectiveness of feature selection by rough sets on web page classification. The rough set theory is a mathematical tool for modeling incomplete or imprecise information [9]. It has been used for both feature selection and knowledge discovery in a number of real world domains, including medicine, pharmacology, control systems, social sciences, switching circuits, and image processing [13][11]. In this paper, we apply the rough set theory to feature selection for web page classification. In our application, web pages in a training data set are first represented using top frequent words. Then a feature selection method based on rough sets is applied to remove redundant features from the training data. A rule induction system, named ELEM2 [1], is then used to learn classification rules from the reduced training data. Therefore, in our application the rough sets based feature selection is used as a pre-processing step for ELEM2. To evaluate the effectiveness of rough set feature selection on web page classification, we conduct experiments to compare the predictive performances of ELEM2 on web page classification with and without rough set feature selection. We describe our experiments and report the evaluation results.

The paper is organized as follows. In the next section, we describe the importance of web page classification and the problems that need to be solved for web page classification. We also present our data collection and representation methods. In section 3, we present the basic concepts in rough sets and describe an algorithm for computing a *reduct*, a non-redundant subset of features. The ELEM2 rule induction method is briefly introduced in section 4. Our method for classifying a web page is presented in Section 5. In section 6, we describe our evaluation methods and report experimental results. Finally, we conclude the paper in section 7.

2 The Problem of Web Page Classification

The World Wide Web contains an estimate of 968 million pages as of March 2002 in the Google search engine [8] and an estimate of 7 million or more pages being added daily [5]. Describing and organizing this vast amount of content is essential for realizing the web as an effective information resource. Text classification has become an important process for helping web search engines to organize this vast amount of data. For instance, most Internet search engines, such as Yahoo and Looksmart, divide the indexed web documents into a number of categories for the users to limit the search scope. Moreover, text classification makes the results easier to browse. If the results returned by the search engine have been classified into a specified category, the users can choose the interesting category to continue browsing. Traditionally, text classification is performed manually by domain experts. However, human classification is unlikely to keep pace with the rate of growth of the web.

Hence, as the web continues to increase, the importance of automatic web page classification becomes obvious. In addition, automatic classification is much cheaper and faster than human classification.

To make the text classification process automatic, machine learning techniques can be applied to generate classification models from a set of text documents with pre-labeled categories. The classification model can then be used to automatically assign natural language texts to the predefined categories based on their contents. In order to apply a machine learning technique to web page classification, the following problems need to be solved. First, to build a web page classifier, we need to collect a set of web pages as training examples to train the machine learning system. These training examples should have pre-defined class labels. Second, the content of a web page in the training set should be analyzed and the page should be represented using a formalism that the learning system requires for representing training examples. This text representation issue is central to our application. Finally, how to classify new pages with induced rules is another challenge in our application. We use different sets of features to represent training pages of different categories. Therefore, the rules for different categories are expressed using different sets of features. When classifying a new page, these different sets of rules should be used together in some way to determine the category or categories of the new page.

2.1 Data Collection

We use the Yahoo web site to collect training examples for our leaning problem. Yahoo is best known for maintaining a web categorization directory. The web directory in Yahoo is a multi-level tree-structured hierarchy. The top level of the tree, which is the first level below the root of the tree, contains 14 categories. Each of these 14 categories contains sub-categories that are placed in the second level below the root. The third and fourth levels of the tree contain both further-refined categories and web pages. We use the top-level categories in Yahoo to label the web pages in our training set. Only 13 of the 14 top-level categories are used and one category, named "Regional", is excluded because it has too much overlap with other categories.

We randomly selected over 7600 pages from the Yahoo category. We originally planned to gather 500 example pages from each category. Unfortunately, some web pages that have links in Yahoo were eliminated or not connected to the Internet. In addition, some web pages contain very few terms after the removal of stop-words because these pages consist of a brief greeting sentence, image, Java script, flash, and other non-textual information. Hence, our number of training examples for each category is different. The distribution of these training examples among the 13 categories is shown in Table 1. The categories may overlap. For example, a document discussing sport action may be reasonably classified into Entertainment and Recreation & Sports categories.

Table 1. Distribution of the training data

Category	Number of web pages
Arts & Humanities	783
Business & Economy	997
Computers & Internet	745
Education	485
Entertainment	957
Government	229
Health	772
News & Media	747
Recreation & Sports	506
Reference	501
Society & Culture	253
Science	230
Social Science	510
Total	7,615

2.2 Representation of Web Pages

After training pages are selected, we apply Porter's stemming algorithm [10] to transfer each word in a web page into its stem. We then remove all the stop words according to a standard stop words list. For each category, we count the number of occurrences of each remaining word stem in all the pages that belong to the category. The word stems in each category are then sorted according to the number of occurrences. This process results in two sets of documents. One is 13 sorted lists of word stems, one for each category. These lists will be used to select features for the training data. The other set of results is the set of web pages, each represented by remaining word stems and their counts.

We use word stem counts to represent a web document. A web document may contain a huge number of words and not all the words in the global space appear in every document. If we use all the words in the global space to represent the documents, the dimensionality of the data set is prohibitively high for the learning system. In addition, even though our learning system can handle thousands of features, many of the features are irrelevant to the learning task. The presence of irrelevant features in the training data introduces noise and extra learning time. Therefore, it is necessary to reduce the dimensionality of the feature set by removing words with low frequencies. Removing infrequent words is also suggested in [7] and [16].

Different categories have different top frequent words. We collected top 60 frequent terms for each category. Since the top frequent terms differ among categories, there is no common set of features that we can use to represent all the documents in all categories. Therefore, even though our learning system can deal with multi-category learning directly, we transform our learning problem into multiple two-class learning problems. That is, for each web page category, we prepare the training data using top n ($n = 20, 30, 40, 50$ or 60 in our experiments) frequent words in the category and then learn a set of rules that can be used to predict whether a new page belongs to this category or not. Therefore, for a given n, we totally have 13 training sets, each of which contains 7,615 documents and is represented by top n frequent terms of the corresponding category. After applying our learning algorithm, the 13 training sets lead to the generation of 13 classifiers. The 13 classifiers will vote to determine which category or categories a new page belongs to.

3 Feature Selection with Rough Sets

We use frequent words to represent the web pages in our training data. However, some frequent words may not be very relevant to our learning task. These words may have little power in discriminating documents of different categories. Therefore, further selection of relevant features is important. We apply a rough set based feature selection method for this purpose. In this section, we first introduce some concepts of rough sets and then describe an algorithm for removing unnecessary attributes.

3.1 Basic Notations

A data set can be formally described using a *decision table*. A decision table (also called an information system [9]) is defined as a quadruple $< U, A, V, f >$, where $U = x_1, x_2, ..., x_N$ is a finite set of objects or examples; A is a finite set of attributes; the attributes in A are further classified into two disjoint subsets, *condition* attributes C and *decision* attributes D such that $A = C \cup D$ and $C \cap D = \emptyset$; $V = \bigcup_{a \in A} V_a$ is a set of attribute values and V_a is the domain of attribute a (the set of values of attribute a); $f : U \times A \rightarrow V$ is an information function which assigns particular values from domains of attributes to objects such that $f(x_i, a) \in V_a$, for all $x_i \in U$ and $a \in A$. In our application, $D = \{d\}$ is a singleton set, where d is the class attribute that denotes the classes of examples.

Given a decision table $DT =< U, A, V, f >$, let B be a subset of A, and let x_i and x_j be members of U, a relation $R(B)$, called an *indiscernibility relation* [9] over B, is defined as follows:

$$R(B) = \{(x_i, x_j) \in U^2 | \forall a \in B, f(x_i, a) = f(x_j, a)\} \quad (1)$$

Let C be a set of condition attributes and $R(C)$ be an indiscernibility relation on U, an ordered pair $AS =< U, R(C) >$ is called an *approximation space* based on C.

Let $Y \subseteq U$ be a subset of objects representing a concept, and $R^*(C) = \{X_1, X_2, ..., X_n\}$ be the collection of equivalence classes induced by the relation $R(C)$. The *lower approximation* [1] of a set Y in the approximation space AS denoted as $LOW_{R(C)}(Y)$, is defined as the union of those equivalence classes in the collection of $R^*(C)$ which are completely contained by the set Y, i.e.,

$$LOW_{R(C)}(Y) = \bigcup \{X \in R^*(C) : X \subseteq Y\}. \tag{2}$$

Let $R^*(D) = \{Y_1, Y_2, ..., Y_m\}$ be the collection of equivalence classes of the relation $R(D)$. A *positive region* $POS_C(D)$ with respect to $R^*(D)$ is defined as

$$POS_C(D) = \bigcup_{i=1,...,m} \{LOW_{R(C)}(Y_i) : Y_i \in R^*(D)\} \tag{3}$$

The positive region $POS_C(D)$ includes all examples of the equivalence classes of $R^*(C)$ in AS which can be certainly classified into classes of $R^*(D)$.

3.2 Attribute Reduction

Attribute reduction techniques eliminate superfluous attributes and create a minimal sufficient subset of attributes for a decision table. Such minimal sufficient subset of attributes, called a *reduct*, is an essential part of the decision table which can discern all examples discernible by the original table and cannot be reduced any more. A subset B of a set of attributes C is a *reduct* of C with respect to D if and only if

(1) $POS_B(D) = POS_C(D)$, and
(2) $POS_{B-\{a\}}(D) \neq POS_C(D)$, for any $a \in B$

A set C of condition attributes may contain more than one reduct. The set of common attributes shared by all the reducts of C is called *core*. The core contains all indispensable attributes of a decision table and can be defined as

$$CORE_C(D) = \{c \in C | \forall c \in C, POS_{C-\{c\}}(D) \neq POS_C(D)\} \tag{4}$$

A good procedure for computing a reduct for a decision table is to compute the core first and then check the other attributes one by one to see if they are essential to the system. If for any attribute $c \in C - CORE_C(D)$, $POS_{C-\{c\}}(D) \neq POS_C(D)$, then c can not be removed from C. Since the order in which the attributes are removed affects the result of reduction, a concept called *relative significance coefficient (RSC)* is introduced to rank the condition attributes. The *relative significance coefficient* (RSC) of the

attribute $c \in C$ based on the set of attributes C with respect to attributes D is defined as

$$RSC_c(C, D) = \frac{card(POS_{C-\{c\}}(D))}{card(POS_C(D))} \quad (5)$$

where *card* is a set cardinality. Our algorithm for computing a reduct is outlined as follows.

1. Compute $CORE_C(D)$. For each condition attribute in C, remove it from C and check whether it changes the positive region. Let $CORE_C(D)$ be the set of all condition attributes whose removal changes the positive region.
2. Check whether $CORE_C(D)$ is a reduct of the rule set. If yes, stop and $CORE_C(D)$ is a reduct.
3. Let $T = C - CORE_C(D)$. Rank the attributes in T in descending order of their RSC value. Let a be the first attribute in T and let C' be C.
4. Check whether $POS_{C'-\{a\}}(D) = POS_C(D)$. If yes, remove a from C'.
5. Let a be the next attribute in T. If a exists, repeat step 4; otherwise, stop and C' is a reduct.

4 ELEM2 Rule Induction

ELEM2 [1] is a rule induction system that learns classification rules from a set of data. Given a set of training data, ELEM2 sequentially learns a set of rules for each class in the data set. To induce rules for a class C, ELEM2 conducts general-to-specific heuristic search over a hypothesis space to generate a disjunctive set of conjunctive rules.[1] ELEM2 uses is a *sequential covering* learning strategy; it reduces the problem of learning a disjunctive set of rules to a sequence of simpler problems, each requiring that a single conjunctive rule be learned that covers a subset of positive examples. The learning of a single conjunctive rule begins by considering the most general rule precondition, i.e., the empty test that matches every training example, then greedily searching for an attribute-value pair that are most relevant to the class C according to the following attribute-value pair evaluation function:

$$SIG_C(av) = P(av)(P(C|av) - P(C))$$

where av is an attribute-value pair and P denotes probability. The selected attribute-value pair is then added to the rule precondition as a conjunct. The process is repeated by greedily adding a second attribute-value pair, and so on, until the hypothesis reaches an acceptable level of performance.

[1] A conjunctive rule is a propositional rule whose antecedent consists of a conjunction of attribute-value pairs. A disjunctive set of conjunctive rules consists of a set of conjunctive rules with the same consequent. It is called disjunctive because the rules in the set can be combined into a single disjunctive rule whose antecedent consists of a disjunction of conjunctions.

In ELEM2, the acceptable level is based on the consistency of the rule: it forms a rule that is as consistent with the training data as possible. Since this "consistent" rule may be a small disjunct that overfits the training data, ELEM2 "post-prunes" the rule after the initial search for this rule is complete. To post-prune a rule, ELEM2 computes a rule quality value for the rule according to one of the rule quality formulas described in [2].[2] ELEM2 then checks each attribute-value pair in the rule in the reverse order in which they were selected to determine if removal of the attribute-value pair will decrease the rule quality value. If not, the attribute-value pair is removed and the procedure checks all the other pairs in the same order again using the new rule quality value resulting from the removal of that attribute-value pair to determine whether another attribute-value pair can be removed. This procedure continues until no pair can be removed.

After rules are induced for all the classes, the rules can be used to classify new examples. The classification procedure in ELEM2 considers three possible cases when matching a new example with a set of rules.

1. *Single match.* The new example satisfies one or more rules of the same class. In this case, the example is classified to the class indicated by the rule(s).
2. *Multiple match.* The new example satisfies more than one rule that indicates different classes. In this case, ELEM2 activates a conflict resolution scheme for the best decision. The conflict resolution scheme computes a decision score for each of the matched classes as follows:

$$DS(C) = \sum_{i=1}^{k} Q(r_i),$$

where r_i is a matched rule that indicates C, k is the number of this kind of rules, and $Q(r_i)$ is the rule quality of r_i. The new example is then classified into the class with the highest decision score.
3. *No match.* The new example is not covered by any rule. Partial matching is considered where some attribute-value pairs of a rule match the values of corresponding attributes in the new example. If the partially-matched rules do not agree on the classes, a partial matching score between an example e and a partially-matched rule r_i with n attribute-value pairs, m of which match the corresponding attributes of e, is computed as $PMS(r_i) = \frac{m}{n} \times Q(r_i)$. A decision score for a class C is computed as

$$DS(C) = \sum_{i=0}^{k} PMS(r_i),$$

where k is the number of partially-matched rules indicating class C. In decision making, the new example is classified into the class with the highest decision score.

[2] We use the C2 rule quality formula in the experiments described in the paper.

5 Classification Method

ELEM2 can learn rules from data with multiple classes and classify a new example into one of the classes. However, since a web page can belong to more than one category and the top frequent words for different categories are different, we transform our learning problem into multiple two-class learning problems. For each of the 13 web page categories, we have a separate training data set, represented by the top frequent words of the category. The web pages in the different training sets are the same among all the categories. For each category, we use ELEM2 to learn a binary classifier. When classifying a new page, the 13 binary classifiers vote to determine which category or categories the new example belongs to. The voting method is as follows. We apply the 13 classifiers to the new example to make binary decisions. The binary decisions are then combined by summing up the scores for each category. The category or categories that have the highest score are chosen to be the predicted category or categories for the new example. Table 2 shows the results from the 13 binary classifiers and the voting result for a test page. In the table, C1, C2, ... and C13 denote categories and B1, B2, ... and B13 denote the binary classifiers. Since category C2 has the most votes, the test page is classified into C2.

Table 2. Classification and voting results for a test page

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
B1	0	1	1	1	1	1	1	1	1	1	1	1	1
B2	0	1	0	0	0	0	0	0	0	0	0	0	0
B3	1	1	0	1	1	1	1	1	1	1	1	1	1
B4	1	1	1	0	1	1	1	1	1	1	1	1	1
B5	1	1	1	1	0	1	1	1	1	1	1	1	1
B6	1	1	1	1	1	0	1	1	1	1	1	1	1
B7	1	1	1	1	1	1	0	1	1	1	1	1	1
B8	1	1	1	1	1	1	1	0	1	1	1	1	1
B9	1	1	1	1	1	1	1	1	0	1	1	1	1
B10	1	1	1	1	1	1	1	1	1	0	1	1	1
B11	1	1	1	1	1	1	1	1	1	1	0	1	1
B12	1	1	1	1	1	1	1	1	1	1	1	0	1
B13	1	1	1	1	1	1	1	1	1	1	1	1	0
Votes	11	13	11	11	11	11	11	11	11	11	11	11	11

6 Evaluation Methods and Results

6.1 Evaluation Methods

Our objective is to investigate the effectiveness of our rough set feature selection method on web page classification. Two groups of tests are conducted in our experiments. In the first group of experiments, ELEM2 is used without using rough set feature selection. In the second group, a reduct is computed for each of the training data sets. ELEM2 is then applied to the reduced sets of features to generate rules. In each group of experiments, to learn a binary classifier for each of the 13 categories, we represent the training data using the top n frequent words for that category. n is set to be 20, 30, 40, 50 and 60 in the experiments, Therefore, we have 13$\times$ 5 training sets in total. Rules learned from each training set are then applied to the examples in a test data set to make binary classifications for each example in the test set. Finally, the voting method is used to combine the binary classification results for each example into a final prediction for the example.

6.2 Test Data

The test data set contains a random sample of 370 web pages from each of the 13 Yahoo categories (excluding the web pages used in the training phase). The total number of test pages is thus 4810. Each of these pages was turned into a separate testing example for each of the 13 binary classifiers. The example is represented using the feature set corresponding to that classifier. Therefore, 13 binary classifications are made for a test web page. The final combined prediction for the web page is compared with the true membership of the page.

6.3 Performance Measures

In the machine learning community, it is common to consider the accuracy of a classifier on a test set as a good indication of the classifier's performance. The testing accuracy is defined simply as the number of correct classifications divided by the total number of classifications. However, the text classification problem is different from typical machine learning problems in two aspects: examples may be given multiple class labels, which means separate binary classifiers must be trained for each class, and the positive examples of each class are usually in a very small minority [12]. These two characteristics implies that a plain accuracy statistic is not adequate to evaluate the performance of a text classifier because high accuracy can be achieved by always predicting the negative class in a binary classification. To deal with this unbalanced nature of classes, we use *precision* and *recall* instead of accuracy. In a binary classification system, *precision* is the proportion of examples labeled

positive by the system that are truly positive, and *recall* is the proportion of truly positive examples that are labeled positive by the system.

In our web page classification, *precision* is the number of correct categories assigned divided by the total number of categories assigned, and serves as a measure of classification accuracy. The higher the precision, the smaller the amount of false categories. *Recall* is the number of correct categories assigned divided by the total number of known correct categories. Higher recall means a smaller amount of missed categories. To compute *precision* and *recall* for a test data set of web pages, we calculate the precision and recall for each example in the test set and then take the averages of precisions and recalls among all the examples in the test set. Suppose the set of real categories of a test example is RC and the set of predicted categories is PC. The precision on this example is $\frac{RC \cap PC}{PC}$. The recall on this example is $\frac{RC \cap PC}{RC}$. Table 3 shows the precisions and recalls for five sample test web pages.

Table 3. Precisions and recalls for some test pages

Page	Real Categories	Predicted Categories	Precision	Recall
P1	{2, 13}	{2}	1	1/2
P2	{8, 13}	{3, 8}	1/2	1/2
P3	{8, 9}	{2, 8, 9}	2/3	1
P4	{3}	{4, 6}	0	0
P5	{3, 4}	{3, 4}	1	1

6.4 Experimental Results

We applied the rough set feature selection method to each of the 13× 5 training data sets. Table 4 shows the number of eliminated attributes for each training set. The average number for each n, where n is the number of top frequent words used to represent the training data, is shown at the bottom of the table. The following observations are made. If the top 20 words are used to represent the training data, no attributes are considered redundant by our rough set attribute reduction algorithm. Therefore, no attribute is removed. However, as more frequent words are used to represent the training data, more attributes are considered redundant and thus removed.

Figures 1 and 2 compare the classification results on the test data set in terms of precision and recall with and without using the rough set feature reduction. The results depend on the number of top frequent words used to represent the training examples. If the number of original features is small (30

Table 4. Number of attributes eliminated by rough set feature selection

Category	Top 20	Top 30	Top 40	Top 50	Top 60
1	0	0	2	2	2
2	0	0	0	0	0
3	0	0	0	1	0
4	0	1	1	2	4
5	0	1	1	4	5
6	0	1	2	1	4
7	0	0	0	0	4
8	0	1	1	1	1
9	0	0	0	1	3
10	0	0	0	0	1
11	0	0	2	1	5
12	0	2	2	2	3
13	0	0	0	1	0
Average	0	0.46	0.85	1.23	2.46

or 40), use of feature selection does not help in terms of precision. It actually decreases the prediction precision. However, as the number of original features becomes bigger (50 or 60), rough set feature selection improves the prediction precision. In terms of recall, in three of the four categories of feature sets, an increase in recall is observed. Only for the top-40 category, we observe a decrease in recall by using the feature selection method. For the top-50 category, the increase is the most significant. Therefore, we can conclude that the rough set feature selection method is effective, that is, it can lead to better precision and recall, when the number of original features is large. One explanation for the results is that the larger the set of original features, the more likely it contains redundant or irrelevant features. Using the rough set technique can remove some redundant or irrelevant features and thus improve the predictive performance.

7 Conclusions

Automated categorization of web pages can lead to better web retrieval tools with the added convenience of selecting among properly organized directories. A challenge in automated web page classification or text classification in general is how to deal with the high dimensionality of the feature space. We have presented an application of machine learning techniques to web page

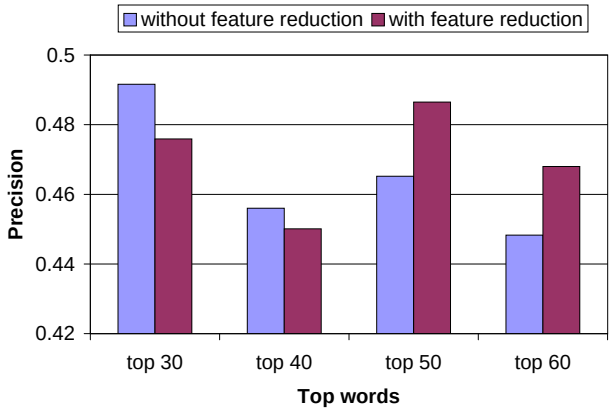

Fig. 1. Comparison in terms of precision

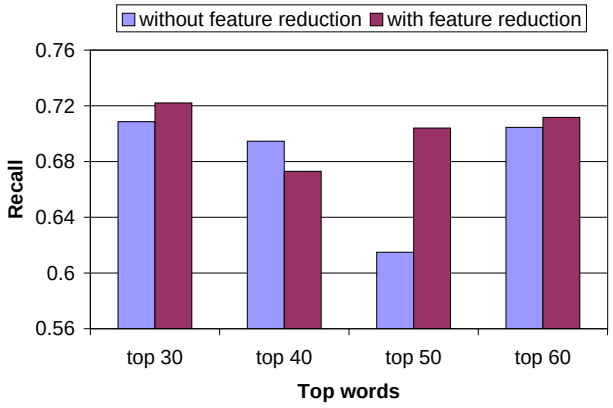

Fig. 2. Comparison in terms of recall

classification. In particular, we used a feature selection method based on the rough set theory to reduce the number of features used to represent a web page. We evaluated the effectiveness of this rough set feature reduction method by comparing the predictive performances of a learning system with and without using the feature selection method. We observed that the rough set feature selection method is effective when the set of features used to represent web pages is large. It can help increase the precision and recall by eliminating redundant or irrelevant features. When the feature set is small (under 50 top frequent words), the feature selection method may not help and may decrease the predictive performance. In our experiments, the rough set feature selection method is used as a pre-processing step for the ELEM2 learning system. However, it can be integrated with other learning methods to remove redundant and irrelevant features. In the future, we shall compare the rough set feature selection method with some statistical feature selection methods on web page classification.

Acknowledgments

This research was supported by grants from the Natural Sciences and Engineering Research Council (NSERC) of Canada and the Institute for Robotics and Intelligent Systems (IRIS).

References

1. An, A. and Cercone, N. (1998) ELEM2: A Learning System for More Accurate Classifications. Proceedings of the 12th Biennial Conference of the Canadian Society for Computational Studies of Intelligence (AI'98). Vancouver, Canada.
2. An, A. and Cercone, N. (2001) Rule Quality Measures for Rule Induction Systems: Description and Evaluation. Computational Intelligence. **17:3**, 409–424.
3. Huang, Y. (2002) Web-based Classification Using Machine Learning Approaches. Master's Thesis, Department of Computer Science, University of Regina, Regina, SK.
4. Lang, K. (1995) Newsweeder: Learning to filter netnews. Proceedings of the Twelfth International Conference on Machine Learning.
5. Lawrence, S. and Giles, L. (1999) Accessibility and distribution of information on the Web. Nature. **400**, 107–109. (http://wwwmetrics.com)
6. Lewis, D.D. and Ringuette, M. (1994) Comparison of two learning algorithms for text categorization. Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94).
7. Mladenic, D. (1998) Feature subset selection in text learning. Proceedings of the 10th European Conference on Machine Learning (ECML98).
8. Notess, G.R. (2002) Search engine statistics: database total size estimates. http://www.searchengineshowdown.com/stats/sizeest.shtml.
9. Pawlak, Z. (1991) Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer.

10. Porter, M.F. (1980) An Algorithm for Suffix Stripping. Program **14:3**, 130–137.
11. Raghavan, V.V. and Sever, H. (1995) The state of rough sets for database mining applications. Proceedings of 23rd Computer Science Conference Workshop on Rough Sets and Database Mining (T.Y. Lin, ed.). 1–11.
12. Scott, S. and Matwin, S. (1998) Text Classification Using WordNet Hypernyms. Proceedings of the Conference on the Use of WordNet in Natural Language Processing Systems.
13. Slowinski, R. ed. (1992) Intelligent decision support - Handbook of advances and applications of the rough set theory. Boston, MA: Kluwer Academic Publishers.
14. Wiener, E. Pedersen, J.O. and Weigend, A.S. (1995) A neural network approach to topic spotting. Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95).
15. Yang, Y. (1995) Noise reduction in a statistical approach to text categorization. Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95).
16. Yang, Y. and Pedersen, J. (1997) A comparative study on feature selection in text categorization. Proceedings of the 14th International Conference on Machine Learning (ICML'97). 412–420.