BayesTH-MCRDR Algorithm for Automatic Classification of Web Document

Woo-Chul Cho and Debbie Richards

Department of Computing,
Macquarie University, Sydney, NSW 2109, Australia
{wccho, richards}@ics.mq.edu.au

Abstract. Nowadays, automated Web document classification is considered as an important method to manage and process an enormous amount of Web documents in digital forms that are extensive and constantly increasing. Recently, document classification has been addressed with various classified techniques such as naïve Bayesian, TFIDF (Term Frequency Inverse Document Frequency), FCA (Formal Concept Analysis) and MCRDR (Multiple Classification Ripple Down Rules). We suggest the BayesTH-MCRDR algorithm for useful new Web document classification in this paper. We offer a composite algorithm that combines a naïve Bayesian algorithm using Threshold and the MCRDR algorithm. The prominent feature of the BayesTH-MCRDR algorithm is optimisation of the initial relationship between keywords before final assignment to a category in order to get higher document classification accuracy. We also present the system we have developed in order to demonstrate and compare a number of classification techniques.

1 Introduction

The soaring pace of development enables much information and data to be obtained through the Internet and many documents to be accessed through it by many organizations. Currently, this information exists on the Internet via the Web using the HTML format. To determine the classification of Web documents techniques from document classification are used. Many information societies make use of Web documents and need to classify and search effectively lots of information and data. So algorithms of document classification have been investigated in many scientific fields. A number of key algorithms to classify documents have been developed including: naïve Bayesian algorithm [1], TFIDF (Term Frequency Inverse Document Frequency) algorithm [2], FCA (Formal Concept Analysis) [3], MCRDR (Multiple Classification Ripple Down Rules) [4] and so on. Among these techniques, Bayesian document classification is the method achieving the most promising results for document classification in every language area [5]. However, the naïve Bayes classifier fails to identify salient document features because it extracts every word in the document as a feature. Further, it calculates a presumed value for every word and carries out classification on the basis of it. The naïve Bayes classifier produces many noisy term (stopwords) and ambiguous results, thus affecting classification. This

G.I. Webb and Xinghuo Yu (Eds.): AI 2004, LNAI 3339, pp. 344–356, 2004.
© Springer-Verlag Berlin Heidelberg 2004

misclassification lowers the precision. So in order to increase precision TFIDF is suggested which uses the Bayesian classification method. This produces less misclassification than the naïve Bayes classifier, but does not reflect the semantic relationships between words and fails to resolve word ambiguity. Therefore it cannot resolve misclassification of documents. In order to solve this problem, we have developed Web Document Classification system which combines both the Bayesian Dynamic Threshold algorithm and the MCRDR algorithm, to produce what we refer to as the BayesTH-MCRDR algorithm. This system applies both the Bayesian algorithm using Dynamic Threshold in order to increase precision and the MCRDR algorithm in order to optimise and construct a knowledge base of related words.

In short, our system first extracts word features from Web documents by using Information Gain [6]. Next, the terms are stemmed using Porter stemming algorithm [7]. Then the documents are classified temporarily by the Bayesian Algorithm, optimized by the MCRDR algorithm and then finally classified. In order to evaluate this system, we compare our approach to Web document classification with the naïve Bayesian, TFIDF, Bayesian-Threshold algorithms and FCA. In addition, we compared document classification by using the stemming algorithm with the document classification by not using it to determine the value of this additional step.

2 Techniques for Web Document Classification

In this section we briefly introduce the key concepts underlying the BayesTH-MCRDR algorithm: Naïve Bayesian, Naïve Bayesian using Threshold, TFIDF and MCRDR. The final subsection describes how we have combined the two techniques.

2.1 Naïve Bayesian

Naïve Bayesian classification [1] uses probability based on Bayes Theorem. This system inputs a vector model of words ($w_1, w_2, \ldots\ldots w_n$) for the given document (d), and classifies the highest probability (p) as the class (c) among documents that can observe the given document. That is, as shown in formula (1) the system classifies it as a highest conditional probability class. If we are concerned with only the highest probability class, we can omit Probability (P), because it is a constant and normalizing term. Also, this approach applies the naïve Bayesian assumption of conditional independence on each 'w_t' which is a feature belonging to the same document (see Formula (2)) [1]. So, the naïve Bayesian Classification method decides the highest probability class according to formula (3).

$$\arg\max_{c \in C} P(c \mid d) = \arg\max_{c \in C} P(c \mid w_1, w_2, \ldots, w_n \mid c) \tag{1}$$

$$= \arg\max_{c \in C} P(w_1, w_2, \ldots, w_n \mid c)\, p(c)$$

$$P(w_1, w_2, \ldots, w_n \mid c) = \prod_{t=1,n} p(w_t \mid c) \tag{2}$$

$$\arg\max_{c \in C} P(c) \prod_{t=1,n} p(w_t \mid c) \tag{3}$$

2.2 Naïve Bayesian with Threshold

In the definition in section 2.1, the Threshold value of the Naïve Bayesian algorithm is fixed. It results in lower precision when the Naïve Bayesian algorithm classifies documents with low conditional probability. The Naïve Bayesian Threshold algorithm is able to increase the precision of document classification by dynamically calculating the value of the threshold as given in formula (4).

$$\begin{aligned}
&Category\ (Class)\ Set\ C = \{c_0, c_1, c_2, c_{3,....,} c_n\} \quad , \quad C_0 = unknown\ class \\
&Document\ Set\ D = \{d_0, d_1, d_2, d_{3,....,} d_i\} \\
&\Re(d_i) = \{P(d_i \mid c_1), P(d_i \mid c_2), P(d_i \mid c_3), ..., P(d_i \mid c_n)\} \\
&P_{\max}(d_i) = \max\{P(d_i \mid C_t)\} \quad , \quad t = 1, ..., n \\
&C_{best}(d_i) = \begin{cases} \{c_j \mid P(d_i \mid c_j) = P_{\max}(d_i),\ if\ P_{\max}(d_i) \geq T\} \\ \qquad\qquad whereT = 1 - \dfrac{P_{\max}(d_i)}{\sum_{t=1}^{n} P(d_i \mid c_t)} \\ c_0 \quad , \quad otherwise \end{cases}
\end{aligned} \tag{4}$$

2.3 TFIDF (Term Frequency Inverse Document Frequency)

TFIDF [2], traditionally used in information retrieval, expresses a weight vector based on word frequency of the given document 'd'. In this case, each word weight (W) is calculated by multiplying the Term Frequency (TF) in a given document 'd' and its reciprocal number, Inverse Document Frequency (IDF), of all documents having the word feature. This means that the higher the IDF, the higher the feature (see Formula (5)). That is, if there is a word which has a higher frequency in a certain document, and a lower frequency in other documents, then the word can express the document very well. For document classification we require a prototype vector expressing each class. The prototype vector (c) of each class is calculated as the average of the weight vector of its training document. Only if each class is expressed in a prototype vector, the similarity is calculated by applying the cosine rule between the weight vector of a given document 'd' and each class prototype vector as shown in formula (6).

$$W_i = TF_i \bullet IDF_i \tag{5}$$

$$\arg\max_{c \in C} \cos(c, d) = \arg\max_{c \in C} \frac{c}{\| c \|} \bullet \frac{d}{\| d \|} \tag{6}$$

2.4 MCRDR (Multiple Classification Ripple Down Rule)

Kang [4] developed Multiple Classification Ripple Down Rules (MCRDR). MCRDR overcomes a major limitation in Ripple Down Rules (RDR), which only permitted single classification of a set of data. That is MCRDR allows multiple independent classifications. An MCRDR knowledge base is represented by an N-ary tree [4]. The tree consists of a set of production rules in the form "If Condition Then Conclusion".

2.4.1 Creation of Rule

We consider a new *case (present case) A* and two *cornerstone cases B* and *cornerstone cases C*. The cornerstone case is the case that prompted the rule being modified (that is, the rule that currently fires on the present case but which is deemed to be incorrect) to be originally added. The present case will become the cornerstone case for the new (exception) rule. To generate conditions for the new rule, the system has to look up the cornerstone cases in the parent rule. When a case is misclassified, the rule giving the wrong conclusion must be modified. The system will add an exception rule at this location and use the cornerstone cases in the parent rule to determine what is different between the previously seen cases and the present case. These differences will form the rule condition and may include positive and negative conditions (see Formula (7)).

$$\begin{aligned} &\text{Positive Condition :} \\ &\quad \text{Present Case (A)} - (\text{Cornerstone Case (B)} \cup \text{Cornerstone Case (C)}) \\ &\text{Negative Condition :} \\ &\quad (\text{Cornerstone Case (B)} \cap \text{Cornerstone Case (C)}) - \text{Present Case (A)} \end{aligned} \tag{7}$$

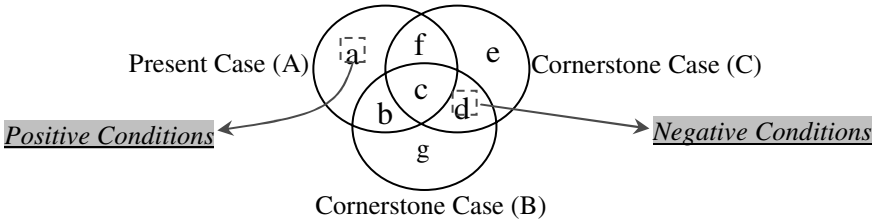


Fig. 1. Difference list {a, not d} are found to distinguish the Present Case (A) from two Cornerstone Cases (B) and (C) [4]

Figure 1 shows a difference list {a, NOT d} between the present case and two cornerstone cases. After the system adds a new rule with the selected conditions by the expert or system, the new rule should be evaluated with the remaining cornerstone cases in the parent rule [4]. If any remaining cornerstone cases are satisfied with the newly added rule, then the cases become cornerstone cases of the new rule [4].

2.4.2 Inference

The inference process of MCRDR is to allow for multiple independent conclusions with the validation and verification of multiple paths [14]. This can be achieved by validating the children of all rules which evaluate to true. An example of the MCRDR inference process is illustrated in Figure 2.

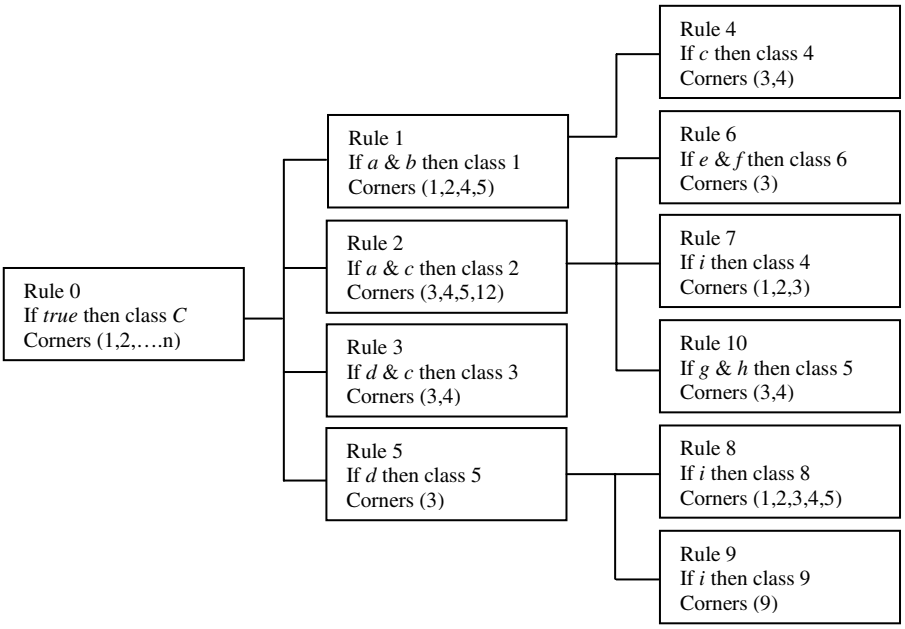


Fig. 2. Knowledge Base and Inference in MCRDR, Attributes: {a, c, d, e, f, h, k} [14]

In this example, a case has attributes {a, c, d, e, f, h, k} and three classifications (conclusion 3, 5 and 6) are produced by the inference. Rule 1 does not fire. Rule 2 is validated as true as both "a" and "c" are found in our case. Now we should consider the children (rules 6, 7, and 10) of rule 2. From comparison of the conditions in children rules with our case attributes, only rule 6 is evaluated as true. Hence, rule 6 would fire to get a *conclusion 6* which is our case classification. This process is applied to the complete MCRDR rule structure in Figure 2. As a result, rule 3 and 5 can also fire, so that *conclusion 3* and *conclusion 5* are also our case classifications.

2.5 FCA (Formal Concept Analysis)

Wille [3] developed Formal Concept Analysis (FCA). FCA is based on the understanding of a concept as an entity of thought, which consists of an extension and intension. Lattice theory is used for representing concepts [8][9]. FCA provides a substitute graphical representation of tabular data that is somewhat instinctive to navigate and use [10]. The fundamental conceptual structure of FCA is the formal

context (K). A formal context comprises a set of objects and their attributes. A formal context constitutes a triple (G, M, I). 'G' is the set of objects, 'M' is the set of attributes and 'I' is a binary relation defined between 'G' and 'M'. Their relation is represented as $I \subseteq G \times M$. Hence, we can define a formal context (K) as: K = (G, M, I). If an object *g* has an attribute m then $g \in G$ is related I to m which is indicated by the relationship $(g , m) \in I$ or *g*I*m*. The notion of a formal concept (G, M, I) is represented as a pair (A, B). 'A' is the set of objects ($A \subseteq G$) and 'B' is set of attributes ($B \subseteq M$). Its two operators are:

2.6 BayesTH-MCRDR

The BayesTH-MCRDR algorithm combines the merits of both the Naïve Bayesian using Threshold (BayesTH) and MCRDR algorithms. A new document can be extracted from feature keywords which are obtained through the Information Gain method (see Section 3.1.3). And then, the document is classified by the BayesTH algorithm into a temporary knowledge base (Table 1). At this moment a document is classified, that is assigned a class. The MCRDR algorithm creates new rules based on the feature keywords in the document. In the BayesTH algorithm, the feature keywords are independent of one another. The MCRDR rules represent the semantic relationships between feature keywords and each rule builds each conclusion (classification) through inference of the MCRDR algorithm (Table 2).

Table 1. Table of Temporary Knowledge Base by BayesTH algorithm

Category	Class	Document No	Keyword
Database	Multimedia	1	A, B
	Data Format	2	X, Y, Z

Table 2. Table of Knowledge Base by MCRDR algorithm

Step	Document	Algorithm	Rules (Keywords)	Class
1	1	Bayesian Threshold	A, B	Multimedia
2	1	MCRDR	A&C	Multimedia
3	1	BayesTH-MCRDR	A, B, A&C, A&B, A&B&C	Multimedia

3 Web Document Classification System

We now introduce the system and accompanying process that have been developed. Section 3.1 describes the preprocessing performed on the documents. Section 3.2 describes the implemented system.

3.1 Data Pre- processing

Data preparation is a key step in the data mining process. For us this step involved deletion of stopwords, feature extraction and modeling and document classification. We describe how these were achieved next.

3.1.1 Deletion of Stopwords

The meaning of 'Stopwords' refers to common words like 'a', 'the', 'an', to', which have high frequency but no value as an index word. These words show high frequencies in all documents. If we can remove these words at the start of indexation, we can obtain higher speeds of calculation and fewer words needing to be indexed. The common method to remove these 'Stopwords' is to make a 'Stopwords' dictionary at the beginning of indexation and to get rid of those words. This system follows that technique.

3.1.2 Stemming

Stemming means the process that searches morphological variations for each keyword and transforms various keywords with the same meaning into a keyword. For example, the words "computers", "computing", "compute", "computed", "computable", "computation" in documents or query are transformed into on indexation word "compute". English has the "Stem" which has a certain kinds of meaning and the "Suffix" which stands for variations of morphology. The reason for stemming is to store stems instead of words, and it enables to reduce the size of the index file and to improve the speed of search. Moreover, it prevents the same key words from being represented differently, so it enables us to calculate the keywords precisely and results in improving the quality of the search. Porter [7] developed the Porter Stemming algorithm and it the most popular technique at the moment. The Porter's stemmer does not remove prefixes. It does however handle suffixes and perform string replacement. The Porter's stemmer has various rules for stemming. Each rule is defined by four sub-items which are: Rule Number, Suffix, Replacement String and Stem State. The Rule Number is a unique number allocated to a certain rule. When several rules are satisfied, the rule with the smallest number of conditions will be applied first. The following two conditions should be satisfied in order for a rule to be applied to a given word. Firstly, given words should include the suffix. In these cases, the suffixes in words may be transformed into the Replacement String specified in the rule. If there is NULL suffix, the word is connected with Replacement String, and if there is a NULL Replacement String, suffixes in words are removed.

3.1.3 Feature Extraction

The process of feature extraction is that of determining which keywords will be useful for expressing each document for classification learning. Document modelling is the process of expressing the existence or non-existence, frequency and weight of each document feature based on a fixed feature word [11].

The most basic method to choose word features which describe a document is to use a complete vocabulary set which is based on all words in the document sets. But this requires extensive computation due to a greater number of word features than the

number of given documents, and the inclusion of a number of word features which do not assist classification but instead reduce classification power. Some words offer semantics which can assist classification. Selecting these words as word features from the complete word set for the set of documents will reduce effort. In this way we consider Feature Extraction to be Feature Selection or Dimension Deduction. There are various ways to achieve feature selection, but our system uses the well-known Information Gain approach [6] that selects words that have a large entropy difference as word features based on information theory.

$$V = \{w_1, w_2, w_3, w_4, w_5, ..., w_n\} \tag{8}$$

$$\begin{aligned} InforGain(w_k) = {} & P(w_k)\sum_i P(c_i \mid w_k)\log\frac{P(c_i \mid w_k)}{P(c_i)} \\ & + P(\overline{w_k})\sum_i P(c_i \mid w_k)\log\frac{P(c_i \mid \overline{w_k})}{P(c_i)} \end{aligned} \tag{9}$$

When the complete set of vocabulary (V) consists of rules (formula (8)) and n words, formula (9) shows the calculation of the information gain for each word w_k. Those words which have the largest information gain are included in the optimized set of word features (K) as in formula (10).

$$K = \{w_1, w_2, w_3, w_4, w_5, ..., w_L\}, K \subset V \tag{10}$$

3.1.4 Learning and Classification

In order to do supervised learning and evaluate the accuracy of Web document classification based on BayesTH-MCRDR we must provide classified documents as input. Our system uses the naïve Bayesian learning method as it is a representative algorithm for supervised learning. The Naïve Bayesian classification learning method classifies each Web document with the highest probability class. Where the conditional probability of a given document is low or there is a conflict the system asks the user to choose the most appropriate classification. In situations where either the difference between the two or more highest conditional probabilities is small or the highest conditional probability is low (for example, the highest conditional probability is 0.2 ~ 0.3 and less) we ask the user to intervene. Since precision and trust are closely related, we don't want the system to give an incorrect classification, resulting in the users loss of faith in the system. Hence, when the system can not clearly assign a class, the system assigns the document to 'Others' for the user to deal with (see Formula (4)). In our system the user is able to set the probability threshold 'T' (see Figure 3), above which the system will assign its own conclusion.

3.2 Implementation

The screen dump in Figure 3 displays the key elements of our system which have been developed to evaluate the performance of the implemented algorithms. The

screen consists of three parts; the top panel is for choosing which classification rule to apply to the set of Web documents, the second panel allows selection of the class (Communication and Networking, Data Formats, Programming and Development, Information Technology and Multimedia) of the data and whether training (learning) or testing (experiment) data is to be used. The third section on the screen (large lower panel) is used to display the contents of the data for the purposes of evaluating and confirming that the data has been classified into the correct class.

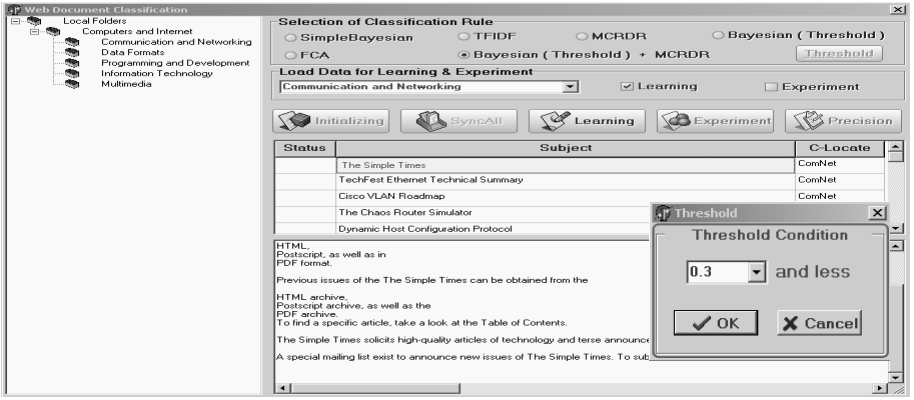


Fig. 3. Web Document Classification System and Control of Threshold value

4 Experiment

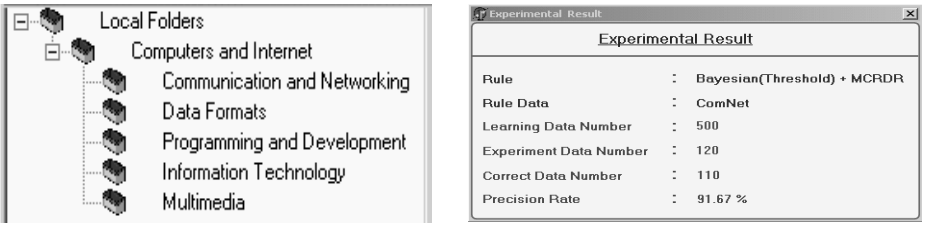
4.1 Aims

A key goal of any classification system is to avoid misclassification. Therefore to validate the precision of the BayesTH-MCRDR algorithm for Web document classification, we carried out some experiments. And through the experiments, we compared the classification precision across six different learning methods, and compared the document classification by using stemming algorithm with the document classification by not using it, to discover the differences.

4.2 Data Collection and Set Up

For the experiments we used Web documents provided by the Web directory service at the Yahoo site[1]. We selected one category: Computer and Internet, in order to evaluate the capability of our system. The 'Computer and Internet' category has five sub-categories which are 'Communication and Networking', 'Data Formats', 'Programming and Development', 'Information Technology' and 'Multimedia' (see Fig. 4).

[1] http://au.dir.yahoo.com/computers_and_internet/



(a) (b)

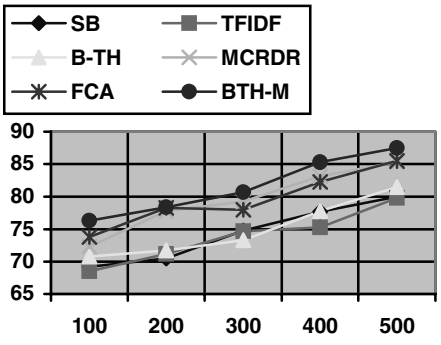
Fig. 4. Classified Category (a) for Experiment and Result (b) in the System

We conducted five experiments for each of the five classes. We gave input learning data sets of 100, 200, 300, 400, 500 into each class (increasing 100 per class when experiment each). The total number of learning data is 500 per class. For evaluating precision we used test sets of 120 experimental data at each experiment. The total number of Learning data and Experiment data was 2,500 and 600 each.

4.3 Results

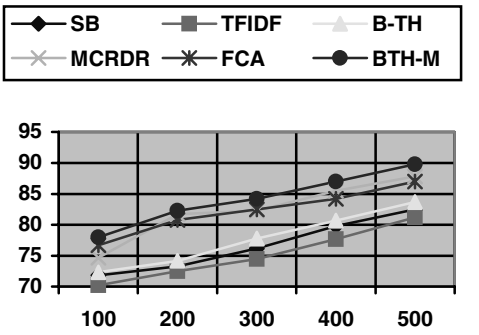
Figure 4(b) shows the evaluation of the precision of each algorithm the user is provided with the precision check function as shown.

	100	200	300	400	500
SB	69.2	70.5	74.8	77.7	80.0
TFIDF	68.5	71.2	74.7	75.3	79.8
B-TH	70.8	71.7	73.3	77.8	81.5
MCRDR	72.3	77.8	79.3	83.2	85.3
FCA	73.8	78.3	78.0	82.3	85.5
BTH-M	76.3	78.4	80.7	85.3	87.5

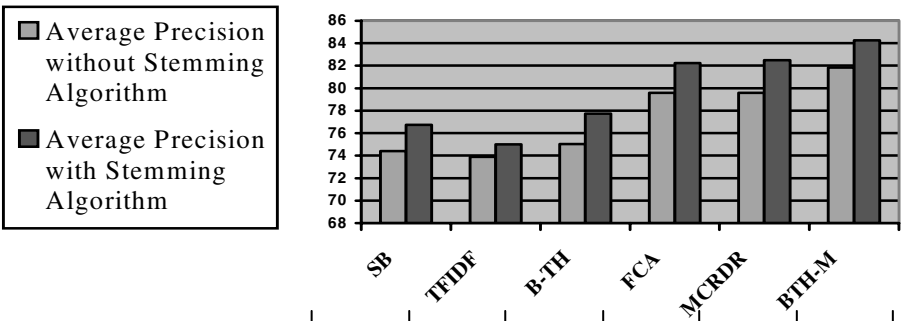


(a)

	100	200	300	400	500
SB	71.8	73.3	76.2	79.8	82.5
TFIDF	70.2	72.5	74.5	77.7	81.2
B-TH	72.3	74.2	77.8	80.7	83.7
MCRDR	74.7	81.7	82.3	85.5	87.8
FCA	76.7	80.8	82.5	84.2	87.0
BTH-M	78.0	82.3	84.2	87.0	89.8



(b)



Average Precision without Stemming Algorithm	**74.42**	**73.90**	**75.03**	**79.60**	**79.58**	**81.83**
Average Precision with Stemming Algorithm	**76.73**	**75.02**	**77.74**	**82.23**	**82.50**	**84.27**

(c)

Fig. 5. *SB: Simple Bayesian; B-TH: Bayesian Threshold; BTH-M: Bayesian Threshold and MCRDR (BayesTH-MCRDR)*, (a): Results of average precision for each algorithm per experiment without stemming algorithm. (b): Results of average precision for each algorithm per experiment with stemming algorithm. (c): Compare both results of average precision for each algorithm per experiment without stemming algorithm and results of average precision for each algorithm per experiment with stemming algorithm.

The experimental results show overall precision 74% - 85% for all algorithms even though there are some differences according to the method of classification learning. Specifically, the more documents used in training the higher the classification accuracy, as we expected. Also there are clear differences in classification accuracy (precision) among classification learning methods. Instance that experiment without stemming algorithm, BayesTH-MCRDR shows the highest precision 81.83% in the system. However, TFIDF shows the lowest precision 73.90%. The system, MCRDR and FCA shows similar precision 79.58% and 79.60 respectively. And simple naïve Bayesian and naïve Bayesian algorithm using Threshold show 74.42% and 75.03%, respectively. On the other hand, in the case of the experiment using stemming algorithm, there is an increase in precision of approx. 2.8% in the system. The system, BayesTH-MCRDR shows the highest precision 84.27% and TFIDF shows the lowest precision 75.02%. The simple naïve Bayesian, Bayesian using Threshold, MCRDR and FCA show 76.73%, 77.74%, 82.50% and 82.23% respectively in the system.

5 Conclusions and Further Work

As presented in the paper, we have achieved higher precision by using the BayesTH-MCRDR algorithm than existing classification methods like simple Bayesian classification method, TFIDF classification method, simple Bayesian using Threshold classification method and FCA classification method. The specific feature of this algorithm which enables it to achieve higher precision is the construction of a related word knowledge base from the learning documents before applying the learnt knowledge to the classification of the test set of documents. Also, we found that classification using the Stemming algorithm achieves higher precision than without using the Stemming algorithm. Other research has shown in general that the Bayesian algorithm using a 'Threshold' has better results than the simple Bayesian algorithm. We have achieved even better results using the BayesTH-MCRDR algorithm which has 3% higher precision than the Bayesian Threshold algorithm.

References

1. Mitchell, T.: Machine Learning, McGraw-Hill, International Edition, (1995)
2. Joachims, T.: A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, *Proceedings of the 14th International Conference on Machine Learning (ICML-97)*, (1997) 143-151
3. Wille R.: Restructuring lattice theory: an approach based on hierarchies of concepts, *Ordered sets*, 1982, pp.445-470
4. Kang, B.H., Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules, PhD dissertation, School of Computer Science and Engineering at the University of New South Wales, (1995)
5. McCallum, A. and Nigram, K.: A Comparison of Event Models for Naïve Bayes Text Classification, *AAAI-98 Workshop on Learning for Tex Categorization*, (1998)

6. Yang, Y. and Pedersen, J.O.: A Comparative Study on Feature Selection in Text Categorization, *Proceedings of the 14th International Conference on Machine Learning*, (1997) 412-420
7. http://snowball.tartarus.org/porter/stemmer.html
8. Birkhoff, G.: *Lattice Theory 3rd edition*, American Mathematical Society, Incremental Clustering for Dynamic Information Processing, ACM Transactions on Information Processing Systems, 11, 1993, pp. 143-164
9. Ganter, B. and Wille.: R, *General lattice theory 2nd edition*. Birkhauser Verlag, Basel, (1998) 591-605.
10. Ganter, B. and Wille, R.:Formal Concept Analysis – mathematical Foundations Berlin, *Springer Verlag*,(1999).
11. Lewis, D.D.: Feature Selection and Feature Extraction for Text Categorization, *Proceedings of Speech and Natural Language Workshop*, (1992) 212-217