E-Mail Document Categorization Using BayesTH-MCRDR Algorithm: Empirical Analysis and Comparison with Other Document Classification Methods

Woo-Chul Cho, Debbie Richards

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

Abstract. The research suggests the BayesTH-MCRDR algorithm for effective classification of E-mail documents. This is a compound algorithm which combines a naïve Bayesian algorithm using Threshold and the MCRDR (Multiple Classification Ripple Down Rules) algorithm. The significant feature of document classification using the BayesTH-MCRDR algorithm is the achievement of higher precision by first establishing a knowledge base of optimally related words generated from the document training set before going on to classify the set of test documents. Further, we demonstrate the system we have developed in order to compare a number of classification techniques.

1 Introduction

The amount of E-mail in usage is increasing by geometric progression with network growth, and e-mail is the favourite program of Internet users. With ongoing development of the Internet, e-mail, the representative communication instrument, costs little, and enables users to exchange information in real time so that many people choose to use it as a communication tool. At the moment private users and companies use it for marketing. This results in the problem of memory shortages for Internet service providers, and requires users to continually spend time removing numerous emails which they do not want to get, and to classify those documents that they are interested in [1][2].

Existing research for automatic document classification by machine learning uses a range of techniques such as probability [3][4], statistical methods [5][6], vector similarity [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 [7]. However, the naïve Bayes classifier [8] 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 (stop-word) and ambiguous results, thus affecting classification. This misclassification lowers the precision. So in order to increase precision TFIDF (Term Frequency Inverse

Document Frequency) is suggested which uses the Bayesian classification method [9][10]. 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 an e-mail 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 e-mail documents by using Information Gain [11]. Then the documents are classified temporarily by the Bayesian Algorithm, optimised by the MCRDR algorithm and then finally classified. In order to evaluate this system, we compare our approach to E-mail classification with the naïve Bayesian, TFIDF and Bayesian-Threshold algorithms.

2 Algorithms for E-Mail Document Classification

In this section we briefly introduce the key concepts underlying the BayesTH-MCRDR algorithm: Naïve Bayesian, Naïve Bayesian with Threshold, Term Frequency Inverse Document Frequency and Multiple Classification Ripple Down Rules. The final subsection describes how we have combined the two techniques.

2.1 Naïve Bayesian

Naïve Bayesian classification [12][13] uses probability based on Bayes Theorem. This system inputs a vector model of words (w_1 , w_2 , …… 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.

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

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

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

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 a same document (see Formula (2)) [12].

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

So, the naïve Bayesian Classification method decides the highest probability class according to formula (3).

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

2.2 Naïve Bayesian with Threshold

In the definition in section 2.1, the Threshold value of Naïve Bayesian algorithm is fixed. It results in lower precision when 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).

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

2.3 TFIDF (Term Frequency Inverse Document Frequency)

TFIDF [5], 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.

$$W_i = TF_i \bullet IDF_i \quad \textbf{(5)}$$

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).

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

2.4 MCRDR (Multiple Classification Ripple Down Rule

Kang [14] 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 [14]. The tree consists of a set of production rules in the form "If Condition Then Conclusion".

2.4.1 Creation of Rule

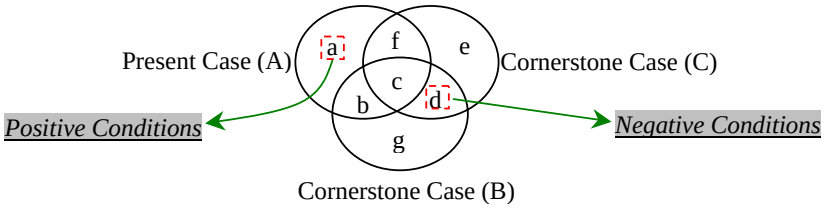


Fig. 1. Difference list {a, not d} are found to distinguish the Present Case (A) from two Cornerstone Cases (B) and (C) [14]

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)).

Positive Condition : (7)

Present Case (A) - (Cornersto ne Case (B) ⋃ Cornerston e Case (C))

Negative Condition :

(Cornersto ne Case (B) ⋂ Cornerston e Case (C)) – Present Case (A)

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 [14]. If any remaining cornerstone cases are satisfied with the newly added rule, then the cases become cornerstone cases of the new rule [14].

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.

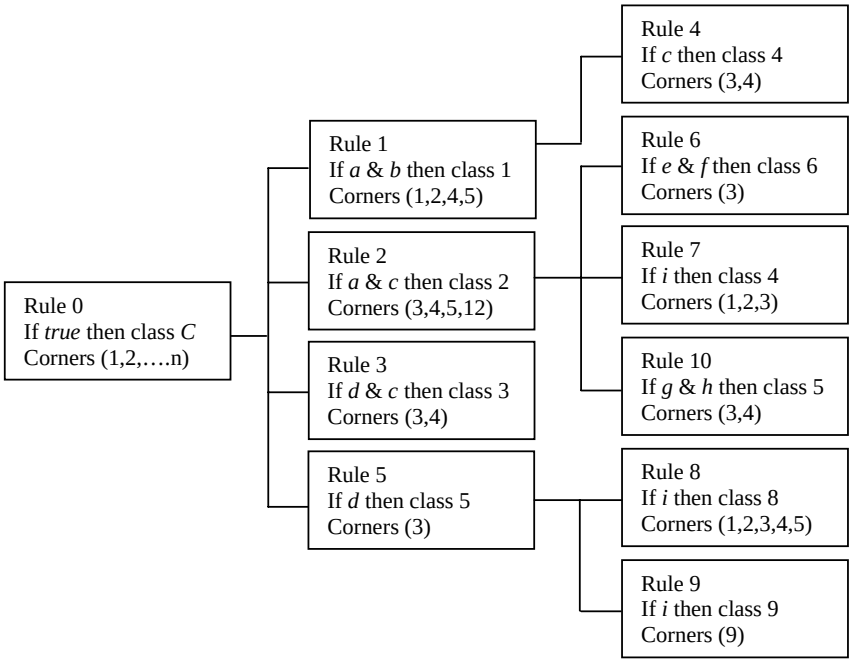


Fig. 2. Knowledge Base and Inference in MCRDR, Attributes: {a, c, d, e, f, h, k} [14]

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 BayesTH-MCRDR

The BayesTH-MCRDR algorithm combines the merits of both the Naïve Bayesian using Threshold (BayesTH) and MCRDR algorithms. As shown in figure 3, a new document can be extracted from feature keywords which are obtained through the Information Gain method (see Section 3.1.2). And then, the document is classified by the BayesTH algorithm into a temporary knowledge base (Table 1.1). 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. In BayesTH-MCRDR, rules stand for a condition for a case to be classified, class stands for a conclusion of a case.

Table 1.1. Table of Temporary Knowledge Base by BayesTH algorithm

Category	Class	Document No	Keyword
Database	mySQL	1	A, B
	pgSQL	2	X, Y, Z
	….	….	….

Table 1.2. Table of Knowledge Base by MCRDR algorithm

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

For example, the learning process for document 1 using the BayesTH-MCRDR algorithm into MySQL class is as follow:

Step 1: Document 1 creates rule A and rule B through BayesTH algorithm. In the BayesTH algorithm, the feature keywords are independent of one another and its created rules. That is, "If Rule A then Class MySQL" or "If Rule B then Class MySQL". Step 2: Document 1 creates rule A&C according to the creation rule process of MCRDR algorithm described above (see section 2.4.1). Step 3: Document 1 creates new rules by combining rules from Step 1 and Step 2. And then, the created rules get a Rule ID and document 1 is classified into MySQL according to the inferencing process described above (see section 2.4.2).

3 E-Mail Classification System

We now introduce the system and accompanying process that have been developed. Section 3.1 describes the preprocessing performed on the documents (email messages). 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 in the beginning of indexation and to get rid of those words. This system follows that technique.

3.1.2 Feature Extraction and Document Modelling

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 [15]. Feature extraction and document modelling are the most important factors affecting document classification efficiency when applying a classification-learning method. We note that there has been a lot of research into both feature extraction and document modelling due to their suitability for Information Retrieval, Information Filtering and Fusion.

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 [11] that selects words that have a large entropy difference as word features based on information theory.

$$V = \{w_1, w_2, w_3, w_4, w_5, ..., w_n\} \quad \textbf{(8)}$$

$$InforGain(w_k) = P(w_k)\sum_i P(c_i \mid w_k)\log\frac{P(c_i \mid w_k)}{P(c_i)} + P(\overline{w_k})\sum_i P(c_i \mid w_k)\log\frac{P(c_i \mid \overline{w_k})}{P(c_i)} \quad (9)$$

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).

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

3.1.3 Learning and Classification

In order to do supervised learning and evaluate the accuracy of e-mail 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 e-mail 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 1), 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 has 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 e-mail documents, the second panel allows selection of the class (mySQL, pgSQL, PHP and so on) 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.

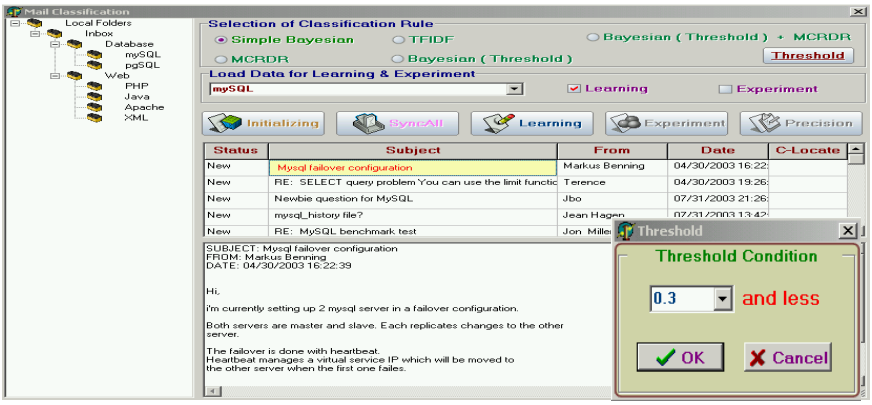


Fig. 3. E-Mail Classification System and Control of Threshold value

4 Experiment

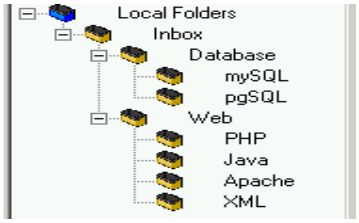
4.1 Aims

A key goal of any classification system is to avoid misclassification. Therefore to validate the precision of the BayesTH-MCRDR algorithm for e-mail classification, we carried out some experiments. And through the experiments, we compared the classification precision across four different learning methods.

4.2 Data Collection and Setup

We used a commercial FAQ (Frequently Asked Questions) E-mail archive as our experimental data in order to ensure fairness. This E-mail archive is available at the website called "Geocrawler.com[2]" and is owned by Open Source Development Network, Inc. We selected two categories, database and web, in order to evaluate the capability of our system. The 'Database' category has two subcategories, 'mySQL' and 'pgSQL', and the 'Web' category has four subcategories, 'PHP', 'Java', 'Apache' and "XML' (see Figure 4). We conducted five experiments for each of the six classes. We gave input learning data 100, 200, 300, 400, 500 into each class (total of 1,500 per class). For evaluating precision we used test sets of 500 experimental data at each experiment. The total number of Learning data and Experiment data was 9,000 and 3,000 each.

Category	Class
Database	MySQL
	PgSQL
Web	PHP



[2] http://www.geocrawler.com/ (viewed 20/4/2004)

	Java
	Apache
	XML

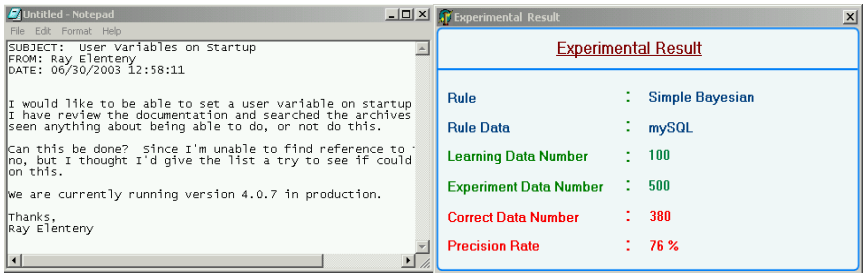
Fig. 4. Experimental Category and Class

Table 2. Data Set for Experiment

Algorithm Name	Class	Learning Data	Experiment Data	Correct Data	Precision
	mySQL	100,200,300,400,500	500		%
	pgSQL	100,200,300,400,500	500		%
	PHP	100,200,300,400,500	500		%
	Java	100,200,300,400,500	500		%
	Apache	100,200,300,400,500	500		%
	XML	100,200,300,400,500	500		%
Total	6 Class	9000	3000		

4.3 Results

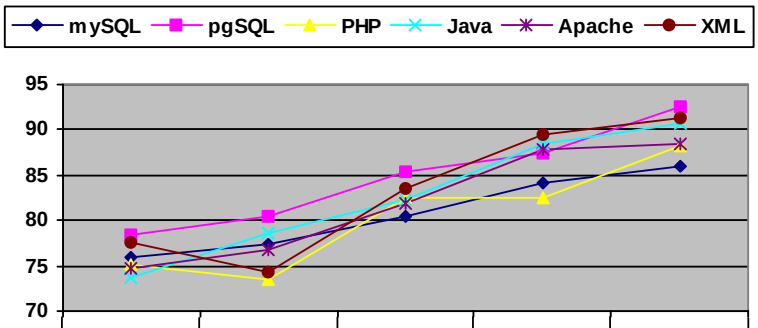
Figure 5(a) shows the formatting of E-mail text data provided by the system**.** To assist evaluation of the precision of each algorithm the user is provided with the Precision check function as shown in Figure 5(b).



(a) (b)

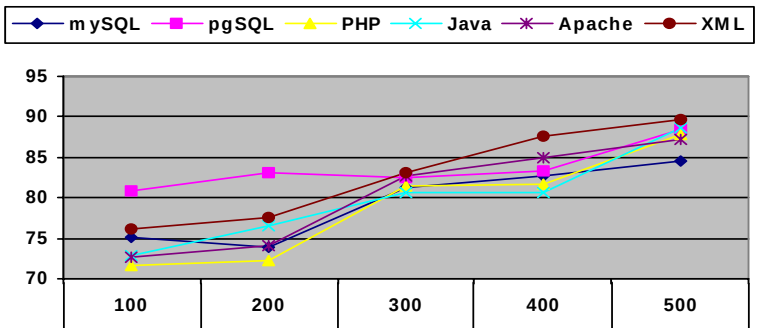
Fig. 5. E-Mail Document Data Format

Figures 6-10 provide the precision results for each of the five algorithms: simple naive Bayesian, TFIDF, Bayesian Threshold, MCRDR and BayesTH-MCRDR, respectively. Averages for all algorithms are given in Figure 11.



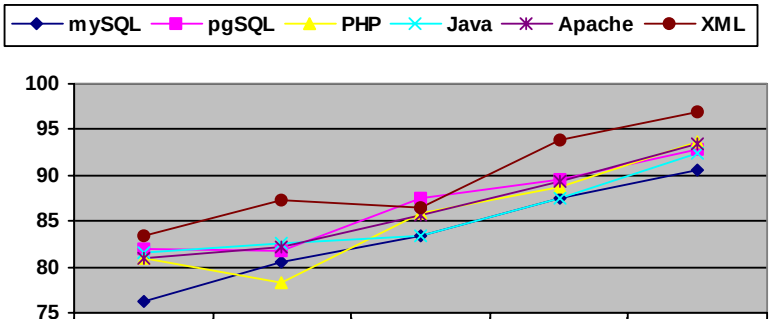
	100	200	300	400	500
mySQL	76	77.4	80.4	84.2	86
pgSQL	78.4	80.4	85.4	87.4	92.6
PHP	75.2	73.4	82.6	82.6	88.2
Java	73.6	78.6	82.2	88.4	90.6
Apache	74.8	76.7	81.8	87.9	88.4
XML	77.6	74.4	83.6	89.4	91.4

Fig. 6. Results of Experiment using simple naive Bayesian Algorithm



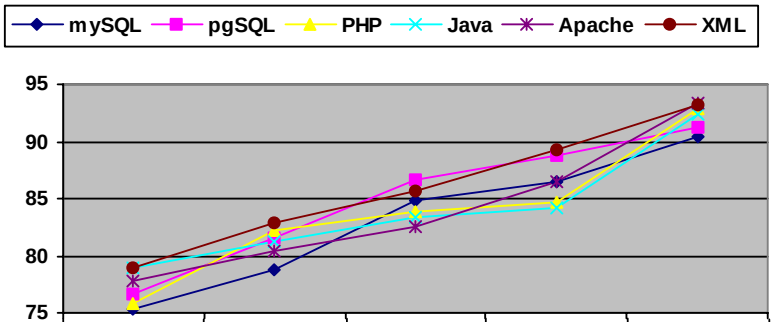
	100	200	300	400	500
mySQL	75.2	73.8	81.2	82.8	84.6
pgSQL	80.8	83.2	82.4	83.4	88.4
PHP	71.6	72.2	81.4	81.6	87.8
Java	72.8	76.6	80.6	80.6	88.6
Apache	72.6	74.2	82.8	85	87.2
XML	76.2	77.6	83.2	87.6	89.6

Fig. 7. Results of Experiment using TFIDF Algorithm



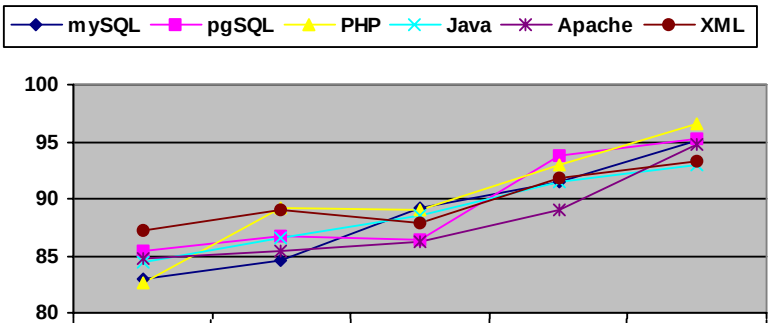
	100	200	300	400	500
mySQL	76.2	80.6	83.4	87.4	90.6
pgSQL	82	81.8	87.6	89.6	92.8
PHP	81	78.2	85.8	88.8	93.6
Java	81.6	82.6	83.4	87.6	92.4
Apache	81	82.2	85.6	89.4	93.4
XML	83.4	87.2	86.4	93.8	97

Fig. 8. Results of Experiment using naive Bayesian Threshold Algorithm



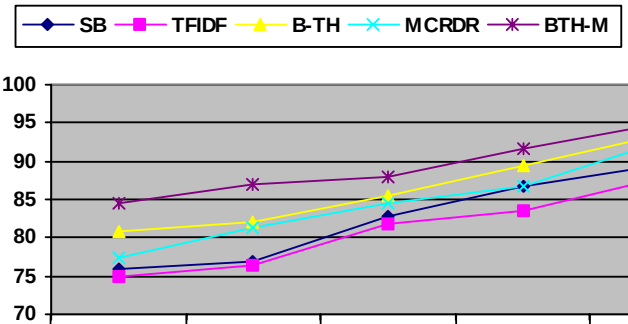
	100	200	300	400	500
mySQL	75.4	78.8	84.8	86.4	90.4
pgSQL	76.6	81.6	86.6	88.8	91.2
PHP	75.8	82.2	83.8	84.6	92.8
Java	79	81.2	83.4	84.2	92.4
Apache	77.8	80.4	82.6	86.4	93.4
XML	79	82.8	85.6	89.2	93.2

Fig. 9. Results of Experiment using MCRDR Algorithm



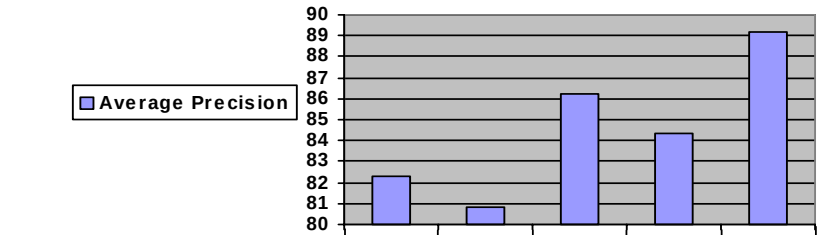
	100	200	300	400	500
mySQL	83	84.6	89.2	91.4	95.1
pgSQL	85.4	86.8	86.4	93.8	95.2
PHP	82.6	89.2	89	93	96.6
Java	84.4	86.6	88.6	91.4	93
Apache	84.8	85.4	86.2	89	94.2
XML	87.2	89	87.8	91.8	94.7

Fig. 10. Results of Experiment using BayesTH-MCRDR Algorithm



	100	200	300	400	500
SB	75.9	76.8	82.7	86.7	89.5
TFIDF	74.9	76.3	81.9	83.5	87.7
B-TH	80.9	82.1	85.4	89.4	93.3
MCRDR	77.3	81.2	84.5	86.6	92.3
BTH-M	84.6	86.9	87.9	91.7	94.8

(a)



	SB	TFIDF	B-TH	MCRDR	BTH-M
Average Precision	82.32	80.86	86.22	84.38	89.18

(b)

Fig. 11. (a) Results of Average Precision for each experiment, SB: Simple Bayesian; B-TH: Bayesian Threshold; BTH-M: Bayesian Threshold and MCRDR (BayesTH-MCRDR).
(b) Results of average precision for each algorithm

The experimental results show high overall precision 80% - 89% 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 among classification learning methods. The system, BayesTH-MCRDR shows the highest precision 89.18%. On the contrary, TFIDF shows the lowest precision 80.86%. And TFIDF, naïve Bayesian, and MCRDR show 80.86%, 82.32%, and 84.38% respectively. We also note, that BayesTH-MCRDR outperforms all the other algorithms for all sizes of training sets and matures more quickly, achieving accuracy levels after 100 cases similar to the accuracy levels achieved by the other algorithms after seeing 300 cases. Looking at the individual results (in Figures 6-10), rather than the average precision (figure 11), we note that the two methods using MCRDR tend to have a smaller spread of results across classes. That is the standard deviation of results across the six classes is smaller (for example MCRDR had a range of 90.4-93.2 for 500 cases and BayesTH-MCRDR had a range of 93-96.9) than for the other techniques. In contrast, the Bayesian Threshold algorithm achieved the highest precision rate of 97 for XML using a training set of 500 cases but only achieved 90.6 accuracy for the mySQL class.

5 Conclusions and Future Work

The development of the Internet enables us to exchange many e-mail correspondences but also to receive many messages that we are not interested in and must expend time and energy to filter out. To make matters worse, the filtering process can result in the loss or misplacement of messages that we did need to respond to. To alleviate the amount of human effort involved, we suggest the BayesTH-MCRDR algorithm for effective e-mail classification. 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 and

simple Bayesian 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. Other research has shown in general that the Bayesian algorithm using a 'Threshold' has better results than the simple Bayesian algorithm. But this paper shows that the BayesTH-MCRDR algorithm has 3% higher precision than the Bayesian Threshold algorithm. If we can construct a related word database through the learning documents, we can get much higher accuracy of document classification.

References

1. Mladenic, D. and Grobelnik, M.: Feature selection for classification based on text hierarchy, *Proceedings of the workshop on Learning for Text and the Web*, Pittsburgh, USA, (1994)
2. John, G.H., Kohavi, Ron. and Rfleger K.: Irrelevant Features and the Subset Selection Problem, *Proceedings of ICML (94)*, Morgan Kaufmann Publishers, San Francisco, CA, (1994) 121-129
3. Lewis, D.D.: Naïve (Bayes) at forty: The Independence Assumption in Information Retrieval, *Proceedings of ECML-98*, (1998)
4. McMahon, J. and Smith, F.: Improving statistical language model performance with automatically generate word hierarchies, *Computational Linguistics*, Vol.22, No.2, (1995)
5. Joachims, T.: A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, *Proceedings of ICML-97*, (1997) 143-151
6. Han, K.R., Sun. B.G., Han, S.T. and Lim, G.W.: A Study on Development of Automatic Categorization System for Internet Documents, *KIPS-2000*, Vol. 07, No. 09, Autumn, South Korea, (2000) 2867-2875.
7. McCallum, A. and Nigram, K.: A Comparison of Event Models for Naïve Bayes Text Classification, *AAAI-98 Workshop on Learning for Tex Categorization*, (1998)
8. Michael, T.: Machine Learning, McGraw-Hill, (1997) 154-200
9. Han, J.G., Park, M.G., Cho, K.J. and Kim, J.T.: Improving the performance of Statistical Automatic Text Categorization by Phrasal Patterns and Keyword Sets, *KIPS-2000*, Vol. 07, No. 04, Autumn, South Korea, (2000) 1150-1159
10. Hur, J.H., Choi, J.H., Lee, J.H., Kim, J.B. and Rim, K.W.: An Automatic Classification System of Korean Documents Using Weight for Keywords of Document and Word Cluster, *KISS-2000*, Vol. 8-B, No. 05, Autumn, South Korea, (2000) 0447-0454
11. 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
12. Mitchell, T.: Machine Learning, McGraw-Hill, International Edition, (1995)
13. Sahami, M., Dumais, S., Heckerman D. and Horvitz, E.: A Bayesian Approach to Filtering Junk E-Mail, *Proceedings of AAAI-98 workshop on Learning for Text Categorization*, (1998)
14. 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)
15. Lewis, D.D.: Feature Selection and Feature Extraction for Text Categorization, *Proceedings of Speech and Natural Language Workshop*, (1992) 212-217