Automatic construction of a concept hierarchy to assist Web document classification

Woo Chul Cho
Macquarie University
Sydney, Australia
Email: wccho@ics.mq.edu.au

Dr Debbie Richards
Macquarie University
Sydney, Australia
Email: richards@ics.mq.edu.au

Abstract

In this paper, we present a new technique which is the Admixture MCRDR-FCA (AMF) algorithm for Web document classification. The technique offers a practical approach to new Web document classification by combining and extending a number of the current techniques. The AMF algorithm has a number of noteworthy features: firstly, it provides a structured conceptual correlation between keywords and secondly it is optimised. Finally, the algorithm creates refined multiple new rules in order to achieve higher accuracy in the conclusions relating to document classification. This is achieved by clarifying the relationships between one concept and another concept before going on to provide a final classification to some category. To evaluate the AMF algorithm, we have developed a demonstration system that permits easy comparison with a number of other classification techniques.

Keywords:

Ontologies, Knowledge Acquisition, Web document classification, Multiple Classification Ripple Down Rules, Formal Concept Analysis

1. INTRODUCTION

Nowadays, almost all information is Web-accessible due to the remarkable evolution of information technology. The main purpose of the Web is to provide information. The Internet is the medium that provides at minimal cost a variety of information and does so more quickly than other mediums. Keeping pace with the growth of the computer industry is the growth in information posted on the Internet by both organizations and individual users. The rate of information creation on the Internet is around two billion items each day (Choi *et al*. 2004). Currently, this information exists on the Internet in the form of Web-based documents using the HTML format which are classified by document classification engines.

The purpose of this study is to develop an improved technique and a supporting demonstration system to provide more accurate document classification. We have developed the Web document classification system using the AMF (Admixture of MCRDR-FCA) algorithm which combines techniques from MCRDR (Kang 1995) and FCA (Wille 1982). In short, our system first deletes stop-words from Web documents. The remaining terms are stemmed using Porter's (1980) Stemming algorithm. Next, we extract word features from the Web documents by using Information Gain (Shannon 1948, Yang, Y. and Pedersen 1997). We use the MCRDR algorithm and the FCA algorithm individually to create an initial temporary knowledge base which is used to filter and generate a new knowledge base allowing final classification. This combining process is known as the AMF algorithm. To evaluate the AMF algorithm, we have developed a demonstration system that permits easy comparison with a number of well-known document classification techniques: naïve Bayesian. Naïve Bayesian with Threshold, Term Frequency-Inverse Document Frequency (TFIDF) and two Support Vector Machine techniques. The Naïve Bayesian classification algorithm (Mitchell 1997) uses probability based on Bayes theorem. The Naïve Bayesian with Threshold has been developed to overcome the lower classification precision of documents with low conditional probability that results due to the use of a fixed threshold in the naïve Bayesian algorithm. We compare our results with term frequency-inverse document frequency (TFIDF) (Ricardo and Berthier 1999), which works by expressing a weight vector based on word frequency of the given document 'd ', due to its traditional use in information retrieval. Finally, Support Vector Machines have been found to produce good results, so we compare our approach with the two main versions Support Vector Machines Pairwise (SVMPR) (Liau and Noble 2003) and Support Vector Machines Winner-Take-All (SVMWTA) (Aiolli and Sperduti 2005).

The paper is organised as follows. In Section 2, we introduce the concepts behind MCRDR and FCA and then describe the AMF algorithm for Web document classification. The subsequent section introduces the demonstration system, describes the experiments and discusses the corresponding results. The last section concludes this paper and suggests some paths for further work.

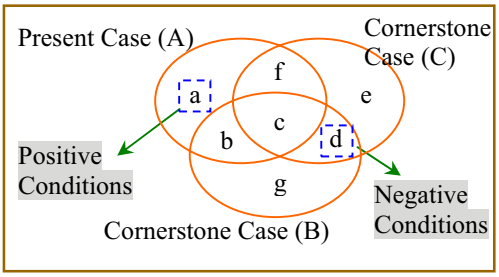
2. DOCUMENT CLASSIFICATION ALGORITHMS

In this section we briefly introduce the key concepts underlying the AMF (Admixture of MCRDR and FCA) algorithm. The final subsection describes how we have combined the latter two algorithms.

2.1. MCRDR (Multiple Classification Ripple Down Rules)

Kang (1995) developed the MCRDR algorithm. 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 (Kang 1995). The tree consists of a set of production rules in the form "*If Condition Then Conclusion*".

2.1.1. Creation of new rule



Formula (1):

Positive Condition

Present Case (A)-(Cornerstone Case (B))∪(Cornerstone Case C))

Negative Condition

(Cornerstone Case (B)∩(Cornerstone Case-Present Case (A)

Figure1. *Difference List {a, not d} are found distinguish the Present Case (A) from two Cornerstones Cases (B) and (C)*

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 to be originally added. That is, the present case has been run and a rule has fired but the domain expert does not agree with the conclusion given. There must be some features in the present case which are different to the cornerstone case which merit a different conclusion. 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 (1)).

2.1.2 Inference

The inference process of MCRDR allows for multiple independent conclusions with the validation and verification of multiple paths (Kang 1995). 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. 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.

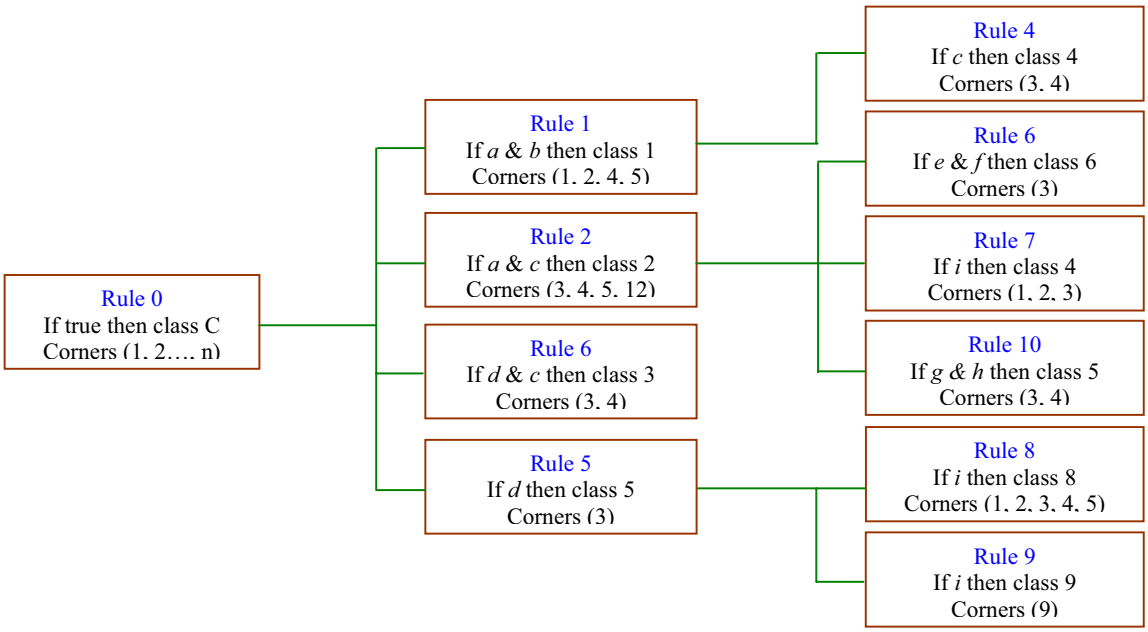


Figure 2. *Knowledge Base and Inference in MCRDR, Attributes are {a, c, d, e, f, h, k}*

2.2. FCA (Formal Concept Analysis)

Wille (1982) developed Formal Concept Analysis (FCA). FCA is based on the understanding of a concept as an entity of thought, consisting of an extension an intension, and representing this information using lattice theory (Birkhoff 1993, Ganter and Wille 1998). FCA provides an alternative graphical representation of tabular data that has been found to be instinctive to navigate and use (Ganter and Wille 1999).

2.2.1. Formal Context

The fundamental conceptual structure of FCA is the Formal Context (FCT). Definition of FCT follows:
Definition 1. A formal context is comprised from 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'. Therefore I $\subseteq$ G × M. We can define a formal context (FCT) as: FCT := (G, M, I). If an object '*g'* has an attributes '*m'* then $g \in$ G is related I to '*m'* which is indicated by the relationship $(g,m) \in$ I or *gIm*. There are mean that '*g*' includes '*m*'. In FCT := (G, M, I), A is the set of objects ($A \subseteq G$) and B is set of attributes ($B \subseteq M$). About the set '*A*' and the set '*B*' satisfied '$A \subseteq G$' and '$B \subseteq M$', its two operators are (see Formula (2)):

Formula (2):

$$\text{A} \subseteq \text{G}: \quad A' = \{m \in M \mid (g,m) \in I \quad \text{for all} \quad g \in A\}$$
$$\text{B} \subseteq \text{M}: \quad B' = \{g \in G \mid (g,m) \in I \quad \text{for all} \quad m \in B\}$$

Definition 2. These two operators are used to formalize the notion of a formal context. In the above operators, A' and B' are set of objects and attributes respectively. A' is included by all objects in A. B' is included by all attributes in B. By finding all intersections of the primitive concepts given in the formal context we can generate all concepts and using the subsumption operator $\subseteq$ we order all concepts to form a concept lattice of all concepts.

2.2.2. Formal Concept

Commonly, Formal Concepts (FC) represents a relationship between objects and attributes. The FC is defined as a pair (A, B). The pair (A, B) satisfies $A' = B$ and $B' = A$ when '$A \subseteq G$' and '$B \subseteq M$' in formal context (G, M, I). In FC, 'A' is the set of objects and called *extent*, and 'B' is the set of attributes and called *intent*.

Formula (3):

$$\text{(a): } A' = \bigcap_{g \in X} \{g\}' \quad , \quad \text{(b): } B' = \bigcap_{m \in Y} \{m\}'$$

We can construct all formal concepts of a formal context by the formulas (3(a)) and (3(b)). We can obtain all extents A′ by determining all row-intents $\{g\}'$ with $g \in G$ (see Formula (3(a))). Also, all intents B′ can be obtained by determining all column-extents $\{m\}'$ with $m \in M$ (see Formula (3(b))). Then, all their intersections are found. For example, *table 1* presents a cross-table of the formal context to which formula (2) has been applied. The *table 2* shows how formal concepts can be derived from the formal context (FCT := (G, M, I)) in *table 1* using formula (3).

Table 1. *A cross-table shoes a formal context for a part of "Animal Kingdom". The column A1-A6 represents has-four legs, has-wings, has-long nose, has-hair, has-scales and has-webfoot. An 'X' indicates that the object has the corresponding attribute.*

	A1	A2	A3	A4	A5	A6
Anteater	X		X	X		
Elephant	X		X			
Beaver	X			X		X
Duck		X				X
Flying Fish		X			X	

Table 2. *This table shows process of finding formal concepts for example of the table 1. The table 2(a) shows results of process 1, 2 and 3. The table 2(b) presents result of process 4.*

Step	Intend	Extend
1		{1, 2, 3, 4, 5}
2	A1	{1, 2, 3}
3	A2	{4,5} { }
4	A3	{1, 2}
5	A4	{1, 3} {1}
6	A5	{5}
7	A6	{3, 4} {3} {4}

(a)

Step	Intend	Extend
1	{ }	{1, 2, 3, 4, 5}
2	{A1}	{1, 2, 3}
3	{A2} {A1, A2, A3, A4, A5, A6}	{4,5} { }
4	{A1, A3}	{1, 2}
5	{A1, A4} {A1, A3, A4}	{1, 3} {1}
6	{A2, A5}	{5}
7	{A6} {A1, A4, A6} {A2, A6}	{3, 4} {3} {4}

(b)

2.3. AMF (Admixture of MCRDR and FCA)

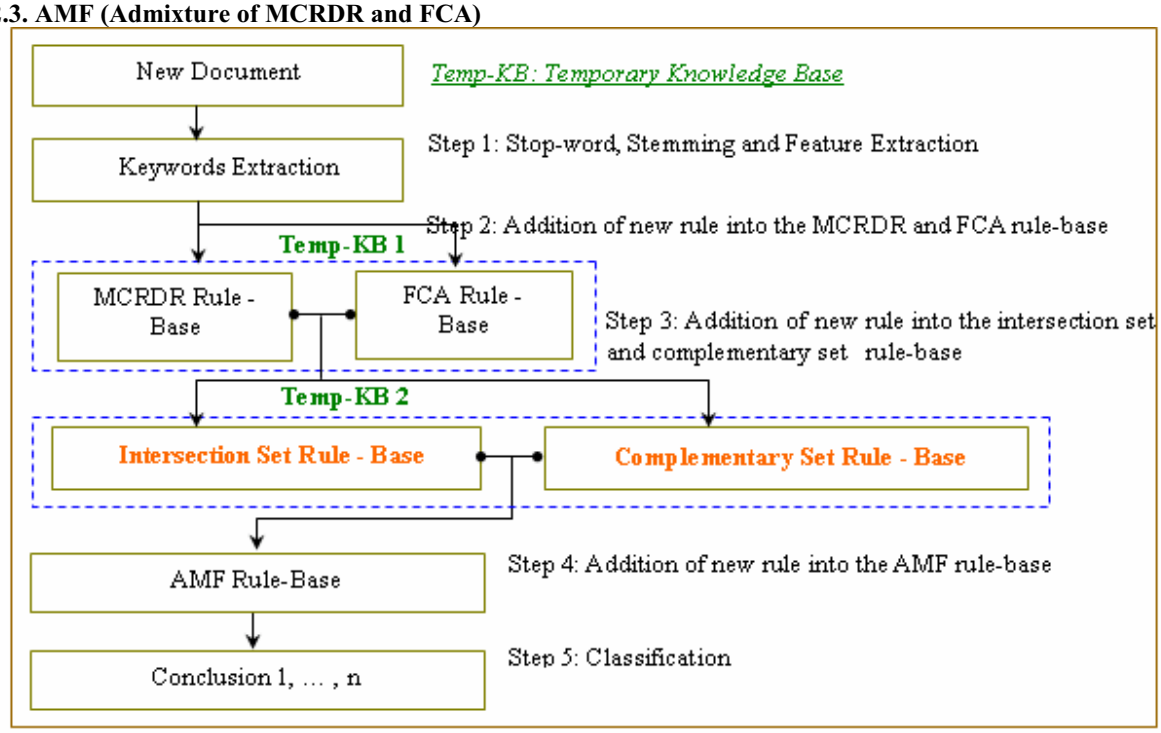


Figure 3. *Diagram of AMF Algorithm for document classification*

The AMF algorithm combines the merits of both the MCRDR and FCA algorithms, as shown in Figure 3. The AMF algorithm obtains new rules for accurate classification based on the feature keywords in the document. The feature keywords are independent of one another in the AMF algorithm. A prominent characteristic of the AMF algorithm is the ability to represent the semantic relationships between feature keywords and to allow conclusion (a Class) to have several rules.

2.3.1. Procedure for Creation New Rule

A process of document classification using the AMF algorithm, shown in Figure 3, is described as follows:

Step 1: As part of our data preprocessing and in order to increase the accuracy of document classification we perform typical stop-word removal, stemming and feature extraction using Information Gain for each document to produce a set of refined keywords.

Step 2: A Temporary Knowledge Base 1 (Temp-KB1) is created which comprises both an MCRDR Rule-Base and an FCA Rule-Base. This is achieved using the refined keywords from step 1 and applying separately the MCRDR algorithm (formula 1) and FCA algorithm (formulas 2 & 3). The algorithms can be applied automatically or manually. For instance in the case of MCRDR the system can select some feature that distinguishes the current case from the cornerstone cases or the rules can be specified by a domain expert, as is more common. The formal context from which the FCA rules/concepts are generated can be specified by a human or automatically populated from cases.

Step 3: The Temporary Knowledge Base 2 (Temp-KB2) contains two Rule-Bases which are the Intersection Set Rule-Base and the Complementary Set Rule-Base. Its two operations are:

Formula (4):

$$\text{Intersection Set} = \{ MCRDR \cap FCA \}$$

$$\text{Complementary Set} = \{ (MCRDR \cap FCA)^{C} \}$$

Using these operators we find the intersection and complementary sets between the rules in the MCRDR Rule-Base and the rules in the FCA Rule-Base in Temp-KB1. Then, the results (new rules) are added to the Intersection Set Rule-Base and Complementary Set Rule-Base in Temp-KB2 respectively.

Step 4: The AMF Rule-Base inherits all intersection rules from the Intersection Set Rule-Base in Temp-KB2. Next, the sets of rules in the Intersection and Complementary Set Rule-Bases are combined using the MCRDR algorithm, therefore creating new rules. Then, the new rules are validated using the FCA algorithm and added to the AMF Rule-Base. The new rules form a conceptual hierarchy of formal abstractions. If any rules could not be validated by the FCA algorithm, a 'stopping-rule' is added by applying the MCRDR algorithm. This means that the rule will not be used anymore.

Step 5: The document is classified into some class according to the inference process described above (see Section 2.4.2).

For example, Figure 4 shows the progress of Web document classification. We assume the keywords (called 'a', 'b', 'c', 'd', 'e') have been extracted from a Web document through applying Step 1. The keywords {a, b, c, d, e} form the new rules {(a&b), (a&c), (d&e)} and {(a&c), (c&d), (b&e)} by applying the MCRDR algorithm and FCA algorithm, respectively. The new rules are saved to the MCRDR Rule-Base and the FCA Rule-Base in the Temp-KB1 separately (Step 2). Next, we find the set of intersecting rule/s {(a&c)} and complementary rule/s {(a&b), (d&e), (c&d), (b&e)} from MCRDR Rule-Base and FCA Rule-Base in the Temp-KB1. Thereafter, the intersection set rule {(a&c)} is added to the Intersection Set Rule-Base and the complementary set rules {(a&b), (d&e), (c&d), (b&e)} are added to the Complementary Set Rule Base in Temp-KB2 (Step 3). The intersection {(a&c)} in the Intersection Set Rule-Base is added to the AMF Rule-Base through inheritance. The Intersection (a&c) is combined with the complementary set rules {(a&b), (d&e), (c&d), (b&e)} in the Complementary Set Rule-Base to create new rules {(a&b&c), (a&b&e), (a&c&d), (a&d&e)} using MCRDR algorithm. Next, the combined new rules {(a&b&c), (a&b&e), (a&c&d), (a&d&e)} are verified by FCA algorithm. Then, the rules separate veridical rule (correct rule) {(a&b&c), (a&c&d)}, and stop-rule {(a&b&e), (a&b&c)} and the rules {(a&b&c), (a&c&d)} are moved to the AMF Rule-Base (Step 4).

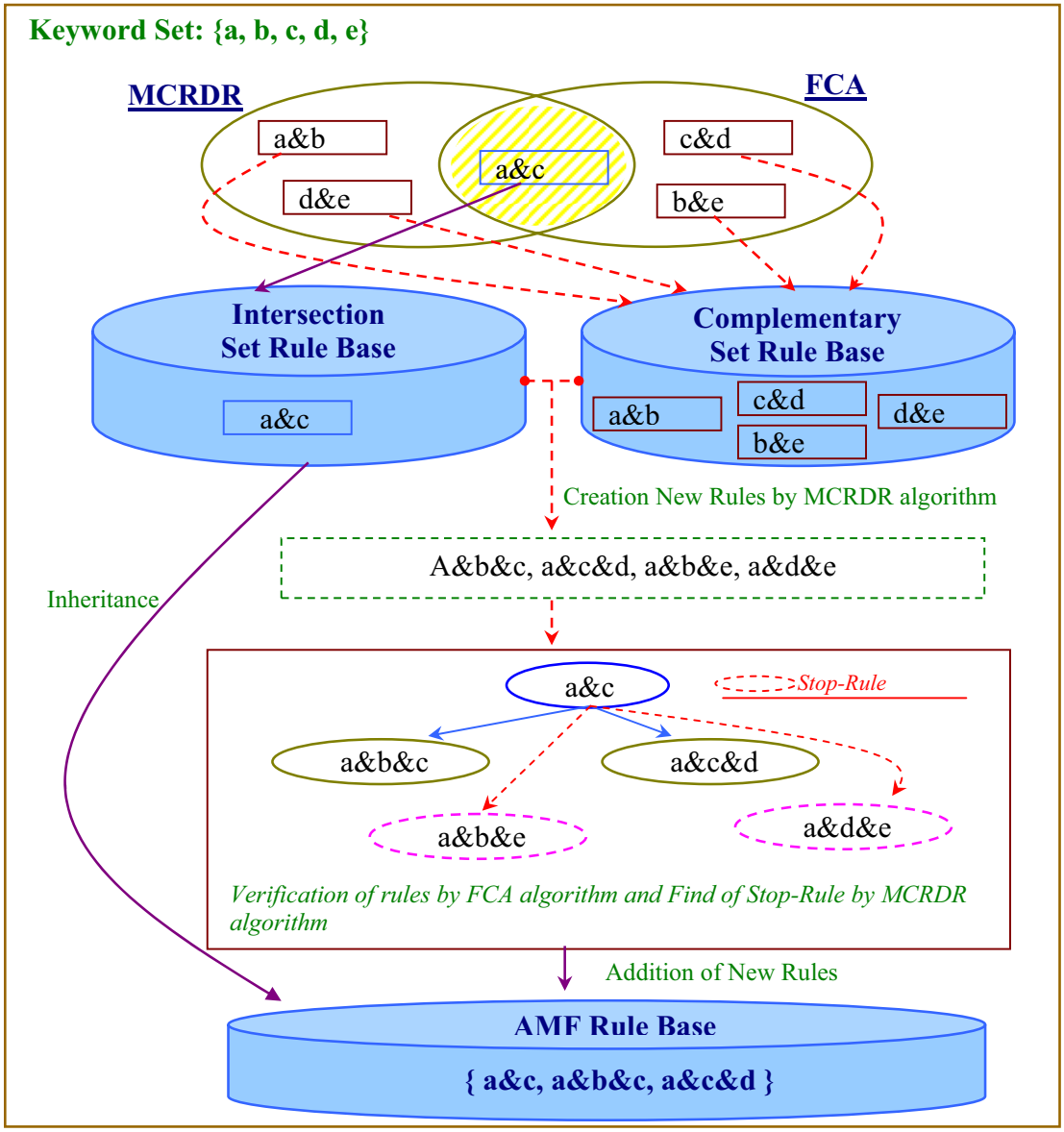


Figure 4. *Exemplifying the process of creating new rules using the AMF algorithm*

3. EXPERIMENT

3.1. Aims and Data Set

We developed the AMF algorithm for Web document classification. In order to validate the performance of the AMF technique, we compared its precision with a number of classification techniques. For the experiments, we used a data set of Web documents provided by Sinka and Corne (2002) which has been used for validating clustering of Web document algorithms. Sinka and Corne (2002) provide four themes, namely, “Banking & Finance”, “Programming & Language”, “Science” and “Sport”. The “Banking & Finance” and “Programming & Language” themes each contain three sub-categories, and “Science” and “Sport” themes each contain two sub-categories (see Table 3).

We extracted data sets containing 100 experimental data elements in each of the ten sub-classes, comprising a total of 1000 elements of experimental data. Each set of 100 was separated into a learning data set (60) and an evaluation data set

(40). We conducted our experiment three times. Initially we trained with 20 elements, each time increasing the input learning data by 20 data until the third run contained 60 elements. The same evaluation data set was used each time (see Table 4).

Table 3. Four themes with ten sub-categories

Theme	Sub-Category
Banking & Finance	*Commercial Banks*
	Building Societies
	Insurance Agencies
Programming Language	*Java*
	C / C++
	Visual Basic
Science	*Astronomy*
	Biology
Sport	*Soccer*

Table 4. Data Set for Experiment

Each of a Sub-Class		
Experiment Frequency	**Learning Data Set**	**Evaluation Data Set**
1 time	20	40
2 time	+ 20	= 40
3 time	+ 20	= 40
Total	**60**	**40**
All Sub-Classes		
Amount	**600**	**400**

3.2 Development System

The screen dump in Figure 5 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 allows us to choose which classification rule to apply to the set of Web documents, the second panel allows selection of the 10 classes (Commercial Banks, Java, Biology, Soccer and so on) of the data and whether training (learning) or testing (evaluation) 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.

In order to improve the performance of the experiment, we performed pre-processing on the data. Data preparation is a key step in the data mining process. As described previously, for us this step involved deletion of stop-words, stemming and feature extraction (see Figure 6) in this system.
The meaning of 'Stop-words' 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 'Stop-words' is to make a 'Stop-words' dictionary in the beginning of indexation and to get rid of those words. This system follows that technique.

For stemming, we used the Porter Stemming algorithm (Porter 1980). Porter's Stemming algorithm is currently the most popular technique for this purpose. The Porter's stemmer does not remove prefixes. 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 which allow the suffix to be replaced by the specified replacement string.

For feature extraction, our system uses the well-known Information Gain approach (Shannon 1948, Yang and Pedersen 1997) that selects words that have a large entropy difference as word features based on information theory. When the complete set of vocabulary (V) consists of rules (formula 5(a)) and n words, formula 5(b) 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 5(c).

Formula (5):

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

(b) $InforGain(w_k) = P(w_k) \sum_i P(c_i \mid w_k) \log \frac{P(c_i \mid w_k)}{P(c_i)}$

$$+ P(\overline{w_k}) \sum_i P(c_i \mid w_k) \log \frac{P(c_i \mid \overline{w_k})}{P(c_i)}$$

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

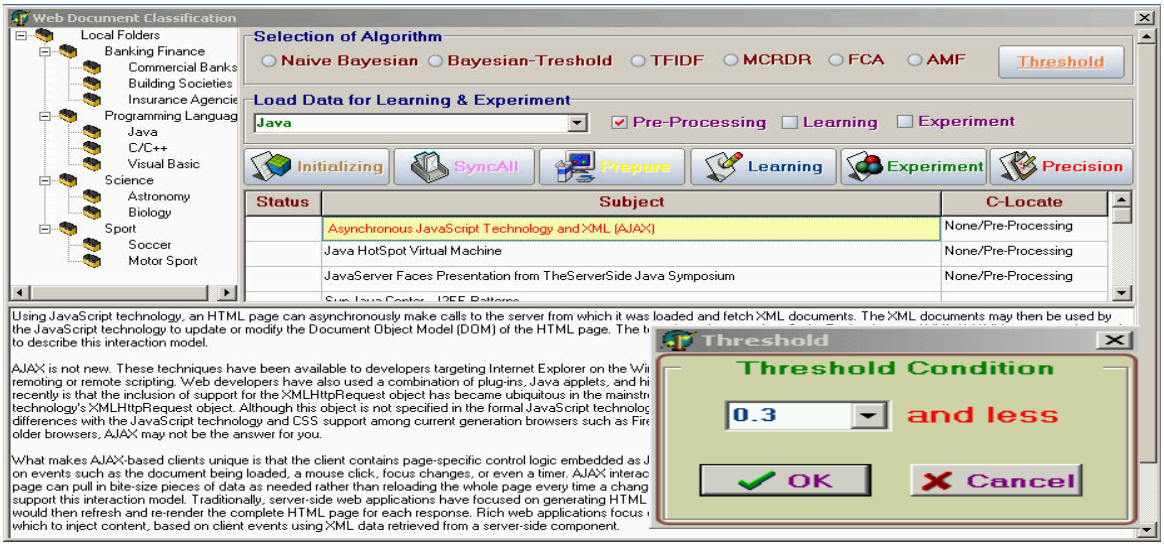


Figure 5. *Web document Classification system for experiment and control of Threshold value*

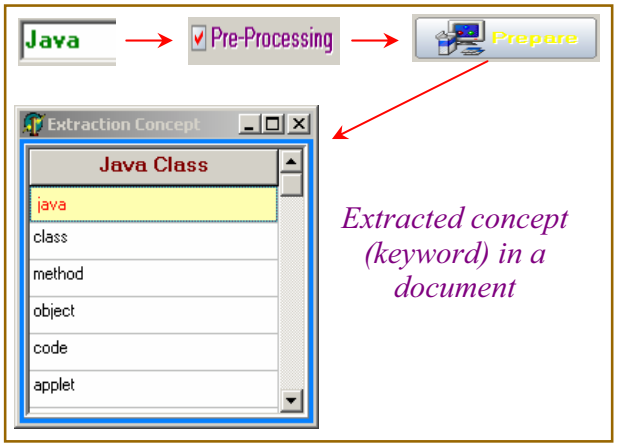


Figure 6. *Example, progress of extracted concepts (keywords) about a document in the system*

3.3. Results

Figure 7 displays the variation across algorithms and datasets for the experimental data. In Figures 7 and 8 and Table 5, columns A1-A8 represent the algorithms: *naïve Bayesian (A1), naïve Bayesian with Threshold (A2), TFIDF (A3), MCRDR (A4), FCA (A5), AMF (A6), SVMPR (A7) and SVMWTA (A8). The* rows C1-C10 represent the data sub-classes: *Commercial Banks (C1), Building Societies (C2), Insurance Agencies (C3), Java (C4), C/C++ (C5), Visual Basic (C6), Astronomy (C7), Biology (C8), Soccer (C9)* and *Motor Sport (C10).* We can see that AMF performs significantly better than all the other algorithms on the Visual Basic dataset, but is outperformed by the SVMWTA on some datasets. Figure 8 aggregates the results to reveal that AMF performs the best overall, even though it is only marginally better than SVMWTA.

Looking at the actual numbers, Table 5 show high overall precision 81% - 89% for all algorithms. As expected, the more documents used in training the higher the classification accuracy. There are clear differences in the classification accuracy of the methods. The AMF algorithm shows the highest precision 91.7% and average precision is 88.61% in the system. On the contrary, TFIDF shows the lowest average precision of 80.74%. Individually, the MCRDR and FCA algorithm achieve similar precision 85.86% and 86.12% respectively in the system. These results provide an initial benchmark and encouragement to continue our investigation in this direction. Future research should investigate and maximize the differences between Web documents and general text documents. In particular the structure offered by HTML and XML offer possibilities for incorporating alternative and multiple methods to achieve improved document classification.

Table 5. *Results of average precision of algorithms for sub-classes in Table 3*

	A1	A2	A3	A4	A5	A6	A7	A8
C1	80.3	80.8	78.3	84.2	85	85.8	85.7	86.3
C2	78.3	81.7	80	85	86.7	86.7	86	89.2
C3	83.3	84.8	81.7	86.7	85.8	87.5	82.8	87.6
C4	79.2	83.3	81.7	85.8	84.2	89.2	85.9	86.7
C5	78.3	82.5	79.2	82.6	85.5	87.7	89.6	91.5
C6	80.8	82.5	80	86.7	87.5	91.7	82.5	84.6
C7	81.7	85.2	81.5	85.4	86.2	87.5	86.6	88.9
C8	79.8	85	84.2	87.5	85.3	90	90.2	91.2
C9	85	81.7	78.3	88.3	86.5	89.2	86.4	87.8
C10	84.2	86.4	82.5	86.4	88.5	90.8	89.9	91.8
Ave	**82.39**	**83.98**	**80.74**	**85.86**	**86.12**	**88.61**	**86.56**	**88.56**

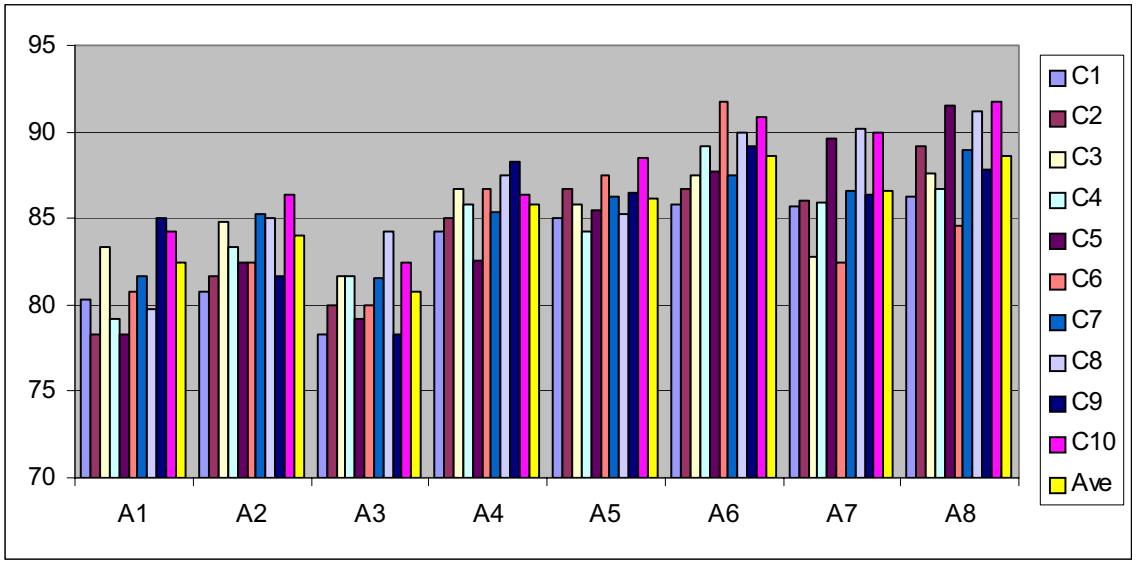


Figure 7. Results of average precision of algorithms for each sub-class in Table 5

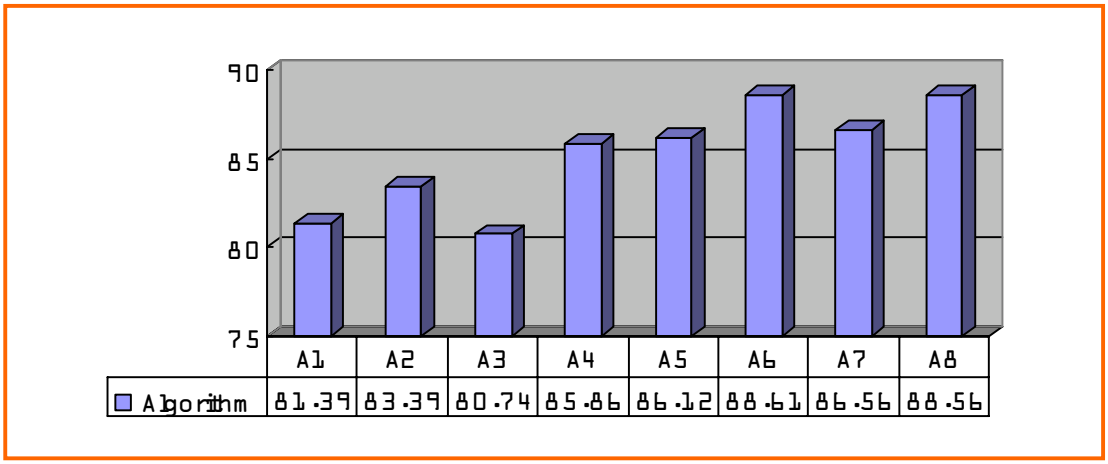


Figure 8. Results of average precision of algorithms in Table 5

4. CONCLUSIONS AND FUTURE WORK

As presented in the paper, we have achieved higher accuracy by using the AMF algorithm than existing classification algorithm like naïve Bayesian, TFIDF, SVMPR, SVMWTA, MCRDR and FCA classification algorithm. Future research should investigate and maximize the differences between Web documents and general text documents. In particular the structure offered by HTML and XML offer possibilities for incorporating alternative and multiple methods to achieve improved document classification. As supported by the results given in this paper, better accuracy in document classification can be achieved by improvements to the pre-processing techniques applied to document classification. The techniques employed for preprocessing, that is stop word removal, stemming and feature extraction, are common and were applied before all the algorithms. We propose that the use of a concept hierarchy provided the necessary structure to focus and organize the data better to achieve higher precision. In this paper we have focused on precision (usefulness of a hitlist) rather than the recall (completeness of the hitlist) as our goal was not to receive the most complete list of relevant web documents, which would take too long to read through, but that the ones we do retrieve are useful.

References

Birkhoff, G., (1993) *Lattice Theory 3rd edition*, American Mathematical Society, Incremental Clustering for Dynamic Information Processing, ACM Trans. on Information Processing Systems, pp. 143-164.

Choi, J.H., Seo, H.S., Noh, S.G., Choi, K.H. and Jung G.h, (2004) "Web Page Classification System based upon Ontology", *KIPS-2004*, Vol. 11, No. 06, South Korea, pp. 723-734.

Ganter, B. and Wille, R., (1998) *General lattice theory 2nd edition*. Birkhauser Verlag, Basel, pp. 591-605.

Ganter, B. and Wille, R., (1999) "Formal Concept Analysis – Mathematical Foundations", Springer-Verlag, Berlin

Kang, B.H., (1995) *Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules*, PhD dissertation, Computer Science, University of New South Wales.

Porter, M.F., (1980) *An algorithm for suffix stripping*, Program 14 (3), pp.130-137.

Shannon, C.E., (1948) “A mathematical theory of communication”, *Bell System Technical Journal*, Vol. 27, pp.379-423 and pp.623-656.

Sinka, M.P. and Corne, D.W., (2002 “A large benchmark dataset for web document clustering”, *Soft computing Systems: Design Management and Applications, Frontiers in Artificial Intelligence and Application*, Vol.87, pp.881-890.

Yang, Y. and Pedersen, J.O., (1997) “A Comparative Study on Feature Selection in Text Categorization”, *14th International Conference on Machine Learning*, pp.412-420.

Wille, R., (1982) *Restructuring lattice theory: an approach based on hierarchies of concepts*, in: Ivan Rival(ed.), Ordered sets, Reidel, Dordrecht-Boston, pp.445-470.