Pattern Extraction Method for Text Classification

Hung Son Nguyen[1] **and Hui Wang**[2]

[1] Institute of Mathematics, Warsaw University, Banacha 2, Warsaw 02095, Poland
E-mail: son@mimuw.edu.pl
[2] School of Information and Software Engineering University of Ulster at Jordanstown N Ireland, BT37 0QB. Email: h.wang@ulst.ac.uk

Abstract. The quality of classification can be increased by using some feature extraction algorithm, i.e. the algorithm that finds new and more relevant features, before application of learning procedure. In this paper, we investigate a novel feature extraction method for textual data. Usually, texts (documents) are represented as collections of words or keywords. We present a method for finding new numerical attributes that improve the quality of classification. New features are based on a set of words (text pattern) and are defined as number of words occurring in both text pattern and the considered document. Our approach is based on Rough set methods and Lattice Machine theory. The experimental results show that the presented methods improve the classification quality on almost all textual data.

1 Introduction

Pattern extraction is one of most important tasks in Data Mining [3]. In previous papers we investigated some problems of pattern extraction from data and relevant feature selection by using Rough Set approach [4,7,8]. We considered the *generalized patterns* being conjunctions of descriptors of form $(attribute \in value\_set)$ (instead of ordinary descriptors of form $(attribute = value)$). We also presented an approach based on *Lattice Machine* [12] for text classification. In some sense, Lattice Machine technique joins data by using some "sum" operations (defined by users) to produce some generalized patterns. In this paper we propose the general scheme for classification systems, in which both feature extraction and feature selection processes are performed automatically from data. This scheme combines the pattern extraction methods as a feature extraction process and Rough Set methods as a feature selection process. We illustrate our proposition by example of text classification problem. Experiment results – performed by simplest application of our approach – show advantages of our proposition.

2 Preliminaries

An *information system* [9] is a pair $\mathbb{A} = (U, A)$, where U is a non-empty, finite set called the *universe* and A is a non-empty finite set of *attributes (or*

features), i.e. $a : U \to V_a$ for $a \in A$, where V_a is called *the value set of* a. Elements of U are called *objects or records*. Two objects $x, y \in U$ are said to be discernible by attributes from A if there exists an attribute $a \in A$ such that $a(x) \neq a(y)$. Any information system of the form $\mathbb{A} = (U, A \cup \{dec\})$ is called *decision table* where $dec \notin A$ is called *decision attribute* (or decision for short). Without loss of generality we assume that $V_{dec} = \{1, \ldots, d\}$. Then the set $DEC_k = \{x \in U : dec(x) = k\}$ will be called the k^{th} *decision class of* $\mathbb{A}$ for $1 \leq k \leq d$.

2.1 Templates as Patterns in Data

Let $\mathbb{A} = (U, A)$ be an information table. By *descriptors* (or simple descriptors) we mean terms of the form $(a = v)$, where $a \in A$ is an attribute and $v \in V_a$ is a value in the domain of a (see [7]). One can define *generalized descriptors* by expressions of form $(a \in S_a)$, where $S_a \subseteq V_a$ is a set of values. By a *template* we mean the conjunction of descriptors:

$$\mathbf{T} = D_1 \wedge D_2 \wedge ... \wedge D_m$$

where $D_1, ...D_m$ are either simple or generalized descriptors. We denote by $length(\mathbf{T})$ the number of descriptors being in $\mathbf{T}$.

For the given template with length m:

$$\mathbf{T} = (a_{i_1} \in S_1) \wedge ... \wedge (a_{i_m} = S_m)$$

the object $u \in U$ is said to satisfy the template $\mathbf{T}$ if and only if $\forall_j a_{i_j}(u) \in S_j$. We denote the set of objects satisfying the template $\mathbf{T}$ by $\mathbf{Supp}(\mathbf{T})$. In this way the template $\mathbf{T}$ describes the set of objects $\mathbf{Supp}(\mathbf{T})$ having the common property: "*values of attributes* $a_{i_1}, ..., a_{i_m}$ belong to $S_1, ..., S_m$, respectively". The template $\mathbf{T}$ is *consistent with a given decision table* $\mathbb{A} = (U, A \cup \{d\})$ (or simply: consistent) if all objects from $\mathbf{Supp}(\mathbf{T})$ belong to one decision class.

The main purpose of pattern extraction methods is to find the minimal set of consistent templates $\mathbf{P} = \{\mathbf{T}_1, ..., \mathbf{T}_m\}$ that $\bigcup_{i=1}^{m} \mathbf{Supp}(\mathbf{T}_i) = U$. In this way, one can obtain more compact representation for the given decision table.

One can find in [7,6] many methods extracting templates from data. In this paper we present a method based on Lattice Machine [12].

2.2 Lattice Machine

Let us consider a given information table $\mathbb{A} = (U, A)$, where $A = \{a_1, ..., a_k\}$. Every object $u \in U$ is associated with a simple template of length k:

$$T_u = (a_1 = a_1(u)) \wedge ... \wedge (a_k = a_k(u))$$

In Lattice Machine [12], every simple template T_u is called *tuple*, and templates are called *hyper_tuples*. The original decision table is called *relation*,

ID	Attribute		Class
	A_1	A_2	
t_0	a	1	0
t_1	a	2	1
t_2	a	3	1
t_3	b	1	0
t_4	b	2	1
t_5	b	3	1
t_6	c	2	0
t_7	c	3	0

$\Longrightarrow$

ID	Attribute		Class
	A_1	A_2	
t'_0	$\{a, b\}$	$\{1\}$	0
t'_1	$\{a, b\}$	$\{2, 3\}$	1
t'_2	$\{c\}$	$\{2, 3\}$	0

Table 1. A simple (database) relation and a hyper relation obtained from the relation in Table 1 by the Lattice Machine.

and its compact version using templates is called *hyper_relation*. Examples of a relation and hyper relation are shown in Tables 1 and 1, respectively.

One can define a ordering relation $\leq$ over templates as follows:

$$\mathbf{T}_1 \leq \mathbf{T}_2 \Leftrightarrow_{def} \left| \begin{array}{l} \text{for any descriptor } D = (a \in S) \text{ from } \mathbf{T}_2 \text{ there exists} \\ \text{a descriptor } D' = (a \in S') \text{ from } \mathbf{T}_1 \text{ such that } S' \subseteq S \end{array} \right.$$

It has been shown [12] that an elegant structure is implied among generalized templates – the collection of all hyper tuples in a given domain is a semilattice under the following ordering relation $\leq$. We call this *domain lattice*. The templates T_u for $u \in U$ are minimal elements in the *domain lattice*.

In a domain lattice, a labelling of objects by decision attribute induces a labelling of the minimal elements of lattice. For example, the following decision table corresponds to the labelled lattice in Figure 1.

E: {Large, Red, Triangle} $\rightarrow \oplus$
G: {Large, Blue, Circle} $\rightarrow \oplus$
J: {Small, Blue, Triangle} $\rightarrow \ominus$
L: {Small, Green, Rectangular} $\rightarrow \ominus$

Given a labelled lattice, a straightforward representation of the labelling is by sets – each set consisting of all data units with the same label, and a simple classification is by enumeration – if a new data unit is the same as one element in the set representation, then it is classified by the label of this element; otherwise, it is marked as unknown. This is in fact the idea of rote learning. Clearly there is no generalisation in this kind of learning. The nearest neighbour method goes one step further. In its basic form, the representation is also by sets, but the classification is based on some distance measure. There is generalisation in this kind of learning, but distance measures can be troublesome in cases where discrete attributes are involved.

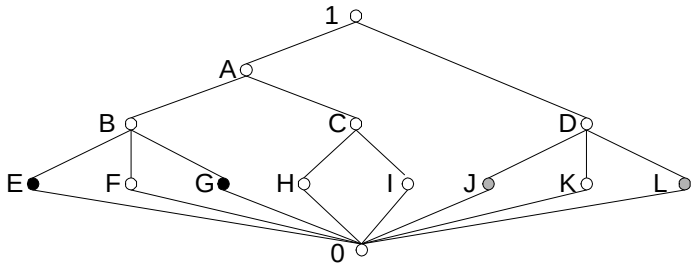

Fig. 1. *A labelled lattice.*

The Lattice Machine takes a different approach to this problem. It represents the labelling by a subset of the elements in the lattice – usually much less than the number of labelled elements. These elements have the property that all elements below them have the same label or are unlabelled, and we called them *equilabelled*. The classification is governed by the following simple rule:

> family given an equilabelled element, any elements below it will have the same label as the equilabelled element (if any).

The hyper relation in Table 1 is in fact the set of equilabelled elements found by the Lattice Machine from the decision table in Table 1.

The core of the Lattice Machine is the following algorithm [12]. Let $\mathbb{A} = (U, A \cup \{dec\})$ be a decision table and let $DEC_k = \{x \in U : dec(x) = k\}$ the k^{th} *decision class of* $\mathbb{A}$ induced by dec for $1 \leq k \leq d$. For any decision class DEC_k, the algorithm finds a set H_k of equilabelled maximal elements (in the sense that the sum of any pair of them is not equilabelled any more). This algorithm is based on the lattice operation *sum* $(+)$ for finding the unique least upper bound of a set of elements.

Let $\mathcal{E}$ be the set of all (possible) equilabelled elements in a labelled lattice.

$C_1 \quad :=_{df} \; DEC_k.$

$C_{n+1} :=_{df}$ The set of maximal elements of $[\downarrow (C_n + DEC_i)] \cap \mathcal{E}.$

Note that, for a lattice L and $e \in L$, $\downarrow e = \{y \in L : y \leq e\}$. It has been shown [12] that there is some n such that $C_n = C_{n+1}$, and therefore $C_n = C_r$ for all $r \geq n$. It has also been proved that $C_n = H_k$. This H_k is called *interior* of class DEC_k.

One of the advantage of Lattice Machine is possibility of its implementation on parallel computers where the lattice sum operations (in Step 2) can be done in short time.

3 Feature extraction and selection algorithms for textual data

In this section we describe some pattern extraction methods for textual data. On the simplest text classification model, one can assume that every textual object (document) is represented by a set of words. The training data set consists of labelled documents. The task is to label new documents by using training data.

One of the most important notions of Rough set theory is: "*indiscernibility relation*". We say that two objects x and y from the domain are indiscernible by the set of attributes $A = \{a_1, ..., a_k\}$ if $\forall_{a \in A} a(x) = a(y)$. In this situation we say that objects x and y are in the relation IND_A and denote by $x\ IND_A\ y$. It is easy to see that the defined above IND_A is equivalent relation, hence it produces a partition of the set of objects into "*equivalent classes*".

In the feature selection process we would like to generate such a set of attributes A that the partition of objects defined by IND_A is consistent with the partition of objects into (decision) classes. In other words, the partition defined by IND_A must be more detail than the partition defined by the decision attribute d. The *discernibility quality* of the set of attributes A can be measured by the number of pairs of different class objects discerned by A i.e.

$$Disc(A) = |\{x, y \in U : d(x) \neq d(y) \wedge \neg(x\ IND_A\ y)\}|$$

The (inside) conflict of the set of objects is defined by

$$conflict(S) = card\left(\{x, y \in S : d(x) \neq d(y)\}\right)$$

Hence, the discernibility quality of A can be computed by

$$Disc(A) = conflict(U) - \sum_i conflict(DEC_i)$$

where $C_i(A)$ are equivalent classes of IND_A.

In the feature selection method based on Rough Set Theory, the set of attributes is initiated by the empty set and in the consecutive steps it is increased by adding the new feature that best improves the discernibility quality of the whole set of attributes. This process is continued until the quality improvement of the next attribute is too small (e.g. when it is equal to 0).

This simple scheme has been adopted to solve many problems in data mining e.g.: attribute reduction, discretization of real value attributes, decision tree construction. The general scheme for solving decision problems based on Rough Set Theory can be presented in Figure 2.

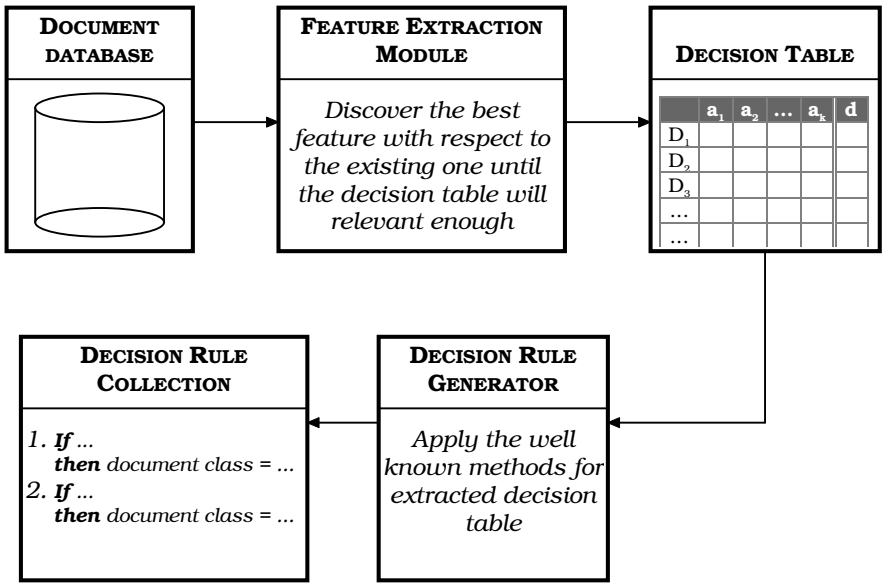

Fig. 2. The general scheme for solving decision problems based on Rough Sets theory

In the next section we describe several methods of searching for *pattern texts* that allow to discern the particular set of documents from another. We define the feature $f_{\mathbf{T}}$ associated to the given pattern text $\mathbf{T}$ as follows:

$$f_{\mathbf{T}}(D) = card(\mathbf{T} \cup D)$$

4 Pattern Text extraction Methods

One consider two possible tasks in the successive step of the feature extraction process for document classification (i.e. some set of attributes has been found). The *global problem* can be formulated as follows:

Given the set of attributes A find a new feature which best improve the quality of A.

The *local problem* can be formulated as follows:

Given the set of attributes A:

- *find the set $\mathbf{S}$ of indiscernible objects by A with maximal value of $conflict(\mathbf{S})$.*
- *find a new feature which discern $\mathbf{S}$ best.*

Let $\mathbf{S}_1$ and $\mathbf{S}_2$ be given two sets of documents. We say that the pattern text $\mathbf{T}$ and the integer k discerns $\mathbf{S}_1$ from $\mathbf{S}_2$ if one of the next conditions satisfies:

$$\forall_{D\in\mathbf{S}_1} f_{\mathbf{T}}(D) > k \tag{1}$$

$$\forall_{D\in\mathbf{S}_2} f_{\mathbf{T}}(D) \leq k \tag{2}$$

Any such pair $(\mathbf{T}, k)$ will be called "*the discriminator for* $\mathbf{S}_1$ *and* $\mathbf{S}_2$" and denoted by $a_{\mathbf{T},k}$. The *discernibility power* of the discriminator $f_{\mathbf{T},k}$ over $\mathbf{S}_1$ and $\mathbf{S}_2$ is defined by:

$Disc_{\mathbf{T},k}(\mathbf{S}_1;\mathbf{S}_2) = |\{D \in \mathbf{S}_1 : f_{\mathbf{T}}(D) > k\}| \times |\{D \in \mathbf{S}_2 : f_{\mathbf{T}}(D) \leq k\}|$

The problem is to find the smallest pattern text $\mathbf{T}$ and the corresponding integer k such that for given two sets of documents $\mathbf{S}_1$ and $\mathbf{S}_2$ the value $Disc_{\mathbf{T},k}(\mathbf{S}_1;\mathbf{S}_2)$ is maximal.

Our methods are based on the Lattice Machine and Boolean reasoning approach. We will describe some basic technique of searching for pattern text in the simplest situations. These techniques will be used to describe the pattern text extraction method.

4.1 Boolean reasoning based method

Let us consider the simplest situation when we would like to discern the only one text document D from the set of documents $\mathbf{S} = \{D_1, ..., D_n\}$. One consider the *Minimal Hitting Set* problem of the family of sets. This problem is NP-complete, but one can apply the greedy heuristics to determine the semi-minimal hitting set of words T for $D - D_1, ..., D - D_n$. as a pattern text. Then the pair $(T, |T|)$ is a good classifier. For any word $w \in D$, we denote by $n_{\mathbf{S}}(w)$ the number of documents $D_i \in \mathbf{S}$ containing the word w. The simplest version of greedy heuristic can be described as follows:

ALGORITHM `Basic algorithm`
Input: `the document` D `and the set of documents`
$\mathbf{S} = \{D_1, ..., D_n\}$ `to be distinguished from` D.
Output: `The minimal hitting set of words` $T \subset D$ `for`
$D \backslash D_1, ..., D \backslash D_n$.

1. $T := \emptyset$;
2. while $\mathbf{S} \neq \emptyset$ **do**
`Search for the word` $w \in D$ `that`
$n_{\mathbf{S}}(w)$ `is minimal;`
`Reduce` $\mathbf{S}$ `to the set of documents`
`containing` w `i.e.`
$\mathbf{S} := \{D : w \in D\}$
endwhile

We assume that $\mathbf{S}_1 = \{D_1, D_2, ..., D_n\}$ and $\mathbf{S}_2 = \{D_{n+1}, ..., D_{n+m}\}$. Using Basic Algorithm, one can find the semi-optimal pattern texts $T_1, ...T_n$ for $D_1, D_2, ..., D_n$, respectively, with regards to $\mathbf{S}_2$. Then we set $T := T_1 \cup ... \cup T_n$. The parameter k can be determined by applying MD algorithm for discretization problem (see [4]).

4.2 Lattice Machine based method

Lattice Machine is a general framework for supervised learning. The basic elements can be re–configured to suit different learning problem. In this section we are going to show how to re-configure the Lattice Machine for the task of text classification.

Representation Lattice Machine was originally proposed to work on structured data. Text documents, however, are unstructured data. A common representation of text document is by boolean feature vector. For example, one could construct a feature for each word appearing in the corpus – a set of documents. A corpus with m documents and n words would thus be represented as a $m \times n$ matrix. This representation in fact turns unstructured datasets into structured. However, this representation is inefficient in its usage of space, since even a moderate sized corpus usually contains many different words. Lattice Machine has a natural solution to this problem.

The ordering relation In the context of text classification, if the documents are relatively large hence different documents have large number of common elements, it would be more efficient to use $\supseteq$ rather than $\subseteq$ to define the ordering relation. This amounts to classifying a document using a subset of the words in the document either in a fixed order or in any order. If the documents are relatively small, however, it would be more effective to use $\subseteq$ instead. This amounts to classifying a document according to whether it is a subset of an already classified document. In what follows, our presentation is mainly for $\supseteq$. Where a result doesn't apply to $\subseteq$, we will clearly spell it out, and an alternative solution will be provided accordingly.

Since we choose to represent a document by a set of words, the ordering relation between documents, denoted by $doc_1 \leq doc_2$, can be specified by:

$$set\_of\_words(doc_1) \supseteq set\_of\_words(doc_2),$$

where $set\_of\_words(doc_i)$ is the set of words in doc_i.

The sum operation Now that documents are represented by sets (word list) and the ordering relation is (inverse) set inclusion, the sum operation (denoted by $+$) is simply set intersection.

ID	SET OF WORDS	CLASS
[1]	97 doc code r area 02 18 n america by state lewen	commun
[2]	97 j code text area 02 18 n america 2 memo by state megibow	commun
[3]	97 doc at t answer machin code 10 15 command for 1343 dave diehl	commun
[4]	97 doc at t answer machin control code 10 15 1545 r frisbi	commun
[5]	97 doc 11 at t answer machin 1339 control code text michael choi	commun
[6]	97 at code text r 02 18 2 hard drive rev sutherland	comput
[7]	97 code text r 02 18 2 xt hard drive rev sutherland	comput
[8]	97 code text 02 18 ibm irq and pc diagnost w taucher	comput
[9]	97 doc j at command 04 09 modem karam	comput
[10]	97 doc j code 02 18 7 error system mac 20k simpson	comput

Table 2. A sample of the MEMO data.

4.3 Example of Lattice Machine Approach

Now we use a sample of the MEMO data shown in Table 2 to illustrate our solution to the text classification problem. This dataset was used in [2]. It was 11 classes, but we use only samples from two classes. Each line corresponds to one document, and it is a list of words in the document (stemmed with the Porter Stemmer algorithm [10,14]). Since the documents have been processed, they may not be readable to humans any more. So are the interiors to be found later.

[1+1]: 97 doc code r area 02 18 n america by state lewen
[1+2]: 97 code area 02 18 n america by state
[1+3]: 97 doc code
[1+4]: 97 doc code r
[1+5]: 97 doc code
[2+2]: 97 j code text area 02 18 n america 2 memo by state megibow
[2+3]: 97 code
[2+4]: 97 code
[2+5]: 97 code text
[3+3]: 97 doc at t answer machin code 10 15 command for 1343 dave diehl
[3+4]: 97 doc at t answer machin code 10 15
[3+5]: 97 doc at t answer machin code
[4+4]: 97 doc at t answer machin control code 10 15 1545 r frisbi
[4+5]: 97 doc at t answer machin control code
[5+5]: 97 doc 11 at t answer machin 1339 control code text michael choi

Table 3. The sum of all pairs of tuples from class "commun" in Table 2

For the "commun" class, we sum all pairs of documents using $\supseteq$ and we get the hyper tuples in Table 3. The set of maximal equilabelled hyper tuples

[6+6]: 97 at code text r 02 18 2 hard drive rev sutherland
[6+7]: 97 code text r 02 18 2 hard drive rev sutherland
[6+8]: 97 code text 02 18
[6+9]: 97 at
[6+10]: 97 code 02 18
[7+7]: 97 code text r 02 18 2 xt hard drive rev sutherland
[7+8]: 97 code text 02 18
[7+9]: 97
[7+10]: 97 code 02 18
[8+8]: 97 code text 02 18 ibm irq and pc diagnost w taucher
[8+9]: 97
[8+10]: 97 code 02 18
[9+9]: 97 doc j at command 04 09 modem karam
[9+10]: 97 doc j
[10+10]: 97 doc j code 02 18 7 error system mac 20k simpson

Table 4. The sum of all pairs of tuples from class "comput" in Table 2

[1+2]: 97 code area 02 18 n america by state	commun
[1+4]: 97 doc code r	commun
[3+5]: 97 doc at t answer machin code	commun
[6+7]: 97 code text r 02 18 2 hard drive rev sutherland	comput
[8+8]: 97 code text 02 18 ibm irq and pc diagnost w taucher	comput
[9+10]: 97 doc j	comput

Table 5. Maximal equilabelled hyper tuples for class "commun" and "comput"

is shown in Table 5. The same operation is applied to this table, and it is clear that the result is the same as this table. Therefore this is the interior for the "commun" class. The same is done for the "comput" class, and the interior for this class is shown in Table 4.

5 Experimental results

To evaluate our approach as a tool for text classification, we applied it to nine benchmark problems which were used in [2]. Table 6 lists the problems and gives a summary description of each domain, the number of classes and number of terms found in each problem, and the number of training and testing examples used in the experiment. On smaller problems 10-fold cross-validation was used, and on larger problems a single hold out set of the specified size was used.

We applied the pattern text extraction method presented in Section 4 for these data tables. As a result, we obtained a decision table, where the founded patterns are attributes and documents are objects (see Table 2).

Datasets	#Train	#Test	#Classes	#Terms	Text-valued field	Label
memos	334	10cv	11	1014	document title	category
cdroms	798	10cv	6	1133	CD-Rom game name	category
birdcom	914	10cv	22	674	common name of bird	phylogenic order
birdsci	914	10cv	22	1738	common scientific name of bird	phylogenic order
hcoarse	1875	600	126	2098	company name	industry (coarse grain)
hfine	1875	600	228	2098	company name	industry (fine grain)
books	3501	1800	63	7019	book title	subject heading
species	3119	1600	6	7231	animal name	phylum
netvet	3596	2000	14	5460	URL title	category

Table 6. Description of benchmark problems.

Afterward, we applied a Rough Set based decision tree construction method for classification (see [4,5]).

Dataset	default	RIPPER	C4.5	LM/Text	Combined
memos	19.8	50.9	57.5	59.8	59.6
cdroms	26.4	38.3	39.2	40.0	40.5
birdcom	42.8	88.8	79.6	90.4	88.9
birdsci	42.8	91.0	83.3	92.3	92.4
hcoarse	11.3	28.0	30.2	20.2	46.1
hfine	4.4	16.5	17.2	13.8	15.7
books	5.7	42.3	52.2	45.3	60.2
species	51.8	90.6	89.4	89.4	89.6
netvet	22.4	67.1	68.8	67.8	68.9
Average	31.6	57.1	57.5	57.7	58.6

Table 7. Experimental results of the Lattice Machine, along with those of C4.5 and RIPPER cited from [2].

The experimental results are listed in Table 7. As a comparison we cite from [2] experimental results on the same set of problems using two state-of-the-art learning algorithms – C4.5 [11], RIPPER [1], Lattice Machine [13] and the proposed method called *Combined method*.

Acknowledgement

This paper has been partially supported by Polish State Committee of Research (KBN) grant No 8T11C02519, grant of the Wallenberg Foundation and

British Council/KBN grant "Uncertainty Management in Information Systems: Foundations and Applications of Non-Invasive Methods (1999-2001)".

References

1. W. W. Cohen. Fast effective rule induction. In *Machine Learning: Proceedings of the Twelfth International Conference*. Morgan Kaufmann, 1995.
2. William W. Cohen and Haym Hirsh. Joins that generalize: Text classification using whirl. In *Proc. KDD-98, New York*, 1998. `http://www.research.att.com/~wcohen/`.
3. V.M. Fayad, G.Piatetsky Shapiro, P. Smyth, R. Uthurusamy (eds): *Advanced in Knowledge Discovery and Data Mining*, AAAI/MIT Press 1996.
4. Nguyen H.Son, Skowron A., 1997. Boolean reasoning for feature extraction problems. In: Z.W. Raś and A.Skowron (Eds.): Proceedings of Tenth International Symposium on Foundation of Intelligent Systems, ISMIS'97, Oct. 1997, NC, USA, *Foundation of Intelligent Systems* **LNAI 1325**, Springer Verlag, pp. 117-126.
5. H.S. Nguyen and S.H. Nguyen. Pattern extraction from data, *Fundamenta Informaticae* **34** (1998) 129–144.
6. Nguyen H. Son, Nguyen S. Hoa (1999). Rough Sets and Association rule Generation. *Fundamenta Informaticae* **40**, pp. 383-405.
7. Nguyen S. Hoa, A. Skowron, P. Synak, 1998. Discovery of data pattern with applications to decomposition and classification problems. In L. Polkowski, A. Skowron (eds.): *Rough Sets in Knowledge Discovery* **2**. Physica-Verlag, Heidelberg, pp. 55–97.
8. Nguyen S.Hoa, 1999. Discovery of Generalized Patterns. In Z.W. Raś and A.Skowron (Eds.): Proceedings of 11^{th} International Symposium on Foundation of Intelligent Systems, ISMIS'99, *Foundation of Intelligent Systems* **LNAI 1609**, pp. 574–582.
9. Pawlak Z., 1991. Rough Sets. Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers, Dordrecht.
10. M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
11. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.
12. Hui Wang, Ivo Düntsch, and David Bell. Data reduction based on hyper relations. In *Proceedings of KDD98, New York*, pages 349–353, 1998.
13. Hui Wang, Son Nguyen. Text classification using Lattice Machine. In *Proceedings of ISMIS'99*, Springer–Verlag, Warsaw, pages 349–353, 1999.
14. Jinxi Xu and W.B. Croft. Corpus-based stemming using co-occurrence of word variants. *ACM TOIS*, 16(1):61–81, Jan. 1998.