Efficient Mining of Jumping Emerging Patterns with Occurrence Counts for Classification

Łukasz Kobyliński and Krzysztof Walczak

Institute of Computer Science, Warsaw University of Technology
ul. Nowowiejska 15/19, 00-665 Warszawa, Poland
{L.Kobylinski, K.Walczak}@ii.pw.edu.pl

Abstract. In this paper we propose an efficient method of discovering Jumping Emerging Patterns with Occurrence Counts for the use in classification of data with numeric or nominal attributes. This new extension of Jumping Emerging Patterns proved to perform well when classifying image data and here we experimentally compare it to other methods, by using generalized border-based pattern mining algorithm to build the classifier.

Keywords: data mining, emerging patterns, image representation, classification

1 Introduction

Recently there has been a strong progress in the area of rule- and pattern-based classification algorithms, following the very fruitful research in the area of association rules and emerging patterns. One of the most recent and promising methods is classification using jumping emerging patterns (JEPs). It is based on the idea that JEPs, as their support changes sharply from one dataset to another, carry highly discriminative information that allows creating classifiers, which associate previously unseen records of data to one of these datasets. As JEPs have been originally conceived for transaction databases, where each data record is a set of items, a JEP-based classifier is not usually directly applicable to relational databases, i.e. containing numeric or nominal attributes. In such case an additional discretization step is required to transform the available data to transactional form.

In this article we address the problem of efficiently discovering JEPs and using them directly for supervised learning in databases, where the data can be described as multi-sets of features. This is an enhancement of the transactional database representation, where instead of a binary relation between items and database records, an occurrence count is associated with every item in a set. Example real-world problems that could be approached in this way include market-basket analysis (quantities of bought products), as well as text and multimedia data mining (numbers of occurrences of particular features). We use a new type of JEPs to accomplish this task – the jumping emerging patterns with occurrence counts (occJEPs) – show both the original semi-naïve algorithm and a new border-based algorithm for finding occJEPs and compare their discriminative value with other recent classification methods.

The rest of the paper is organized as follows. Section 2 outlines previous work done in the field, while Section 3 gives an overview of the concept of emerging patterns in transaction databases. In Sections 4–7 we introduce jumping emerging patterns with occurrence counts (occJEPs), present their discovery algorithms and describe the chosen method of performing classification with a set of found occJEPs. Section 8 presents experimental results of classification and a comparison of some of the most current classifiers. Section 9 closes with a conclusion and discussion on possible future work.

2 Previous Work

The number of papers concerning emerging patterns and the more general concept known as contrast patterns is growing rapidly since the original introduction of EPs by Dong and Li in [1]. Emerging patterns have been defined there as itemsets, for which supports increase significantly from one dataset to another. This idea has become the subject of interest of many researchers, as the patterns proved to be a very accurate alternative to previously proposed rule- and tree-based classifiers. The first classification algorithm based on EP mining – Classification based on Aggregating Emerging Patterns (CAEP) – has been proposed in [2].

Efficient mining of emerging patterns has been studied in [3], while more accurate classification algorithm has been proposed in [4]. In [5] DeEPS, a new, lazy learning scheme for EP-based classification has been presented. A particularly good performance of classification has been achieved using only a subset of EPs, the jumping emerging patterns [6]. Mining efficiency of such patterns has been further improved in [7–9].

More recently, a rough set theory approach to pattern mining has been presented in [10] and a method based on the concept of equivalence classes in [11].

Many applications of EPs have been proposed to date, with a particularly fruitful research in the area of bioinformatics, specifically classification and finding relationships in gene data. The first EP-based algorithms concerning analysis of such data have been proposed in [12, 13].

The concept of recurrent items in transactional systems has been presented in the area of multimedia data analysis in [14] in the context of association rules, while general and efficient algorithms for discovering rules with recurrent items have been studied in [15] and [16]. The extension of the definition of jumping emerging patterns to include recurrent items and using them for building classifiers has been proposed in [17].

3 Emerging Patterns

Emerging patterns may be briefly described as patterns, which occur frequently in one set of data and seldom in another. We now give a formal definition of emerging patterns in transaction systems.

Let a transaction system be a pair $(\mathcal{D}, \mathcal{I})$, where $\mathcal{D}$ is a finite sequence of transactions $(T_1, \ldots, T_n)$ (database), such that $T_i \subseteq \mathcal{I}$ for $i = 1, \ldots, n$ and $\mathcal{I}$ is a non-empty set of items (itemspace). A support of an itemset $X \subset \mathcal{I}$ in a sequence $D = (T_i)_{i \in K \subseteq \{1,\ldots,n\}} \subseteq \mathcal{D}$ is defined as:

$$\mathrm{supp}_D(X) = \frac{|\{i \in K : \ X \subseteq T_i\}|}{|K|}. \tag{1}$$

Given two databases $D_1, D_2 \subseteq \mathcal{D}$ the growth rate of an itemset $X \subset \mathcal{I}$ from D_1 to D_2 is defined as:

$$GR_{D_1 \to D_2}(X) = \begin{cases} 0 & \text{if } \mathrm{supp}_{D_1}(X) = 0 \text{ and } \mathrm{supp}_{D_2}(X) = 0, \\ \infty & \text{if } \mathrm{supp}_{D_1}(X) = 0 \text{ and } \mathrm{supp}_{D_2}(X) \neq 0, \\ \frac{supp_{D_2}(X)}{supp_{D_1}(X)} & \text{otherwise.} \end{cases} \tag{2}$$

Given a minimum growth rate ρ, we define an itemset $X \subset \mathcal{I}$ to be a ρ-emerging pattern (ρ-EP) from D_1 to D_2 if $GR_{D_1 \to D_2}(X) > \rho$. Furthermore, we say that an itemset X is a jumping emerging pattern (JEP), when its growth rate is infinite, that is $GR_{D_1 \to D_2}(X) = \infty$. Having a minimum support threshold ξ, we define a strong ξ-jumping emerging pattern to be a JEP from D_1 to D_2 for which $\mathrm{supp}_{D_1}(X) = 0$ and $\mathrm{supp}_{D_2}(X) > \xi$. A set of all JEPs from D_1 to D_2 is called a JEP space and denoted by $JEP(D_1, D_2)$.

Example 1. For the example database given by Table 1, having $D_1 = (T_{1,2,3})$, $D_2 = (T_{4,5})$, the set of minimal JEPs from D_1 to D_2 is equal to: $\{\{black\}, \{brown\}, \{red, white\}\}$. A minimal JEP is a jumping emerging pattern X, such that no proper subset of X is a JEP, e.g. in the case of $\{red, white\}$ neither $\{red\}$ nor $\{white\}$ are JEPs.

Table 1. Transaction database example. T_i – transactions with binary items, T_i^{r} – transactions with recurrent items.

	i	T_i	T_i^{r}
D_1	1	*blue*, *green*, *white*, *yellow*	$8 \cdot blue,\ 4 \cdot green,\ 3 \cdot white,\ 1 \cdot yellow$
	2	*beige*, *red*, *yellow*	$10 \cdot beige,\ 3 \cdot red,\ 3 \cdot yellow$
	3	*white*, *magenta*	$12 \cdot white,\ 4 \cdot magenta$
D_2	4	*blue*, *brown*, *white*	$6 \cdot blue,\ 2 \cdot brown,\ 8 \cdot white$
	5	*black*, *white*, *red*, *yellow*	$9 \cdot black,\ 2 \cdot white,\ 3 \cdot red,\ 2 \cdot yellow$

4 Jumping Emerging Patterns with Occurrence Counts

Let a transaction system with recurrent items be a pair $(\mathcal{D}^{\mathrm{r}}, \mathcal{I})$, where $\mathcal{D}^{\mathrm{r}}$ is a database and $\mathcal{I}$ is an itemspace (the definition of itemspace remains unchanged). We define database $\mathcal{D}^{\mathrm{r}}$ as a finite sequence of transactions $(T_1^{\mathrm{r}}, \dots, T_n^{\mathrm{r}})$ for $i = 1, \dots, n$. Each transaction is a set of pairs $T_i^{\mathrm{r}} = \{(t_i,\ q_i);\ t_i \in \mathcal{I}\}$, where $q_i : \mathcal{I} \to \mathbb{N}$ is a function, which assigns the number of occurrences to each item of the transaction. Similarly, a multiset of items X^{r} is defined as a set of pairs $\{(x,\ p);\ x \in \mathcal{I}\}$, where $p : \mathcal{I} \to \mathbb{N}$. We say that $x \in X^{\mathrm{r}} \iff p(x) \geq 1$ and define $X = \{x : x \in X^{\mathrm{r}}\}$. We will write $X^{\mathrm{r}} = (X,\ P)$ to distinguish X as the set of items contained in a multiset X^{r} and P as the set of functions, which assign occurrence counts to particular items.

The support of a multiset of items X^{r} in a sequence $D^{\mathrm{r}} = (T_i^{\mathrm{r}})_{i \in K \subseteq \{1,\dots,n\}} \subseteq \mathcal{D}^{\mathrm{r}}$ is defined as:

$$\mathrm{supp}_D(X^{\mathrm{r}}, \theta) = \frac{|\{i \in K :\ X^{\mathrm{r}} \overset{\theta}{\subseteq} T_i^{\mathrm{r}}\}|}{|K|}, \tag{3}$$

where $\overset{\theta}{\subseteq}$ is an inclusion relation between a multiset $X^{\mathrm{r}} = (X,\ P)$ and a transaction $T^{\mathrm{r}} = (T,\ Q)$ with an occurrence threshold $\theta \geq 1$:

$$X^{\mathrm{r}} \overset{\theta}{\subseteq} T^{\mathrm{r}} \iff \forall_{x \in \mathcal{I}}\ q(x) \geq \theta \cdot p(x). \tag{4}$$

We will assume that the relation $\subseteq$ is equivalent to $\overset{1}{\subseteq}$ in the context of two multisets.

Example 2. The support of a multiset of items $X^{\mathrm{r}} = \{1 \cdot \mathit{white},\ 2 \cdot \mathit{yellow}\}$ for threshold $\theta = 1$ in transaction sequence $D_1 = (T_{1,2,3}^{\mathrm{r}})$ of database given by Table 1 is equal to: $\mathrm{supp}_{D_1}(X^{\mathrm{r}}, 1) = 0$. Similarly, for $D_2 = (T_{4,5}^{\mathrm{r}})$, $\mathrm{supp}_{D_2}(X^{\mathrm{r}}, 1) = 1$. Having the threshold $\theta = 2$, $\mathrm{supp}_{D_1}(X^{\mathrm{r}}, 2) = 0$ and $\mathrm{supp}_{D_2}(X^{\mathrm{r}}, 2) = 0$.

Let a decision transaction system be a tuple $(\mathcal{D}^{\mathrm{r}}, \mathcal{I}, \mathcal{I}_{\mathrm{d}})$, where $(\mathcal{D}^{\mathrm{r}}, \mathcal{I} \cup \mathcal{I}_{\mathrm{d}})$ is a transaction system with recurrent items and $\forall_{T^{\mathrm{r}} \in \mathcal{D}^{\mathrm{r}}} |T \cap \mathcal{I}_{\mathrm{d}}| = 1$. Elements of $\mathcal{I}$ and $\mathcal{I}_{\mathrm{d}}$ are called condition and decision items, respectively. A support for a decision transaction system $(\mathcal{D}^{\mathrm{r}}, \mathcal{I}, \mathcal{I}_{\mathrm{d}})$ is understood as a support in the transaction system $(\mathcal{D}^{\mathrm{r}}, \mathcal{I} \cup \mathcal{I}_{\mathrm{d}})$.

For each decision item $c \in \mathcal{I}_{\mathrm{d}}$ we define a decision class sequence $C_c = (T_i^{\mathrm{r}})_{i \in K}$, where $K = \{k \in \{1, \dots, n\} : c \in T_k\}$. Notice that each of the transactions from $\mathcal{D}^{\mathrm{r}}$ belongs to exactly one class sequence. In addition, for a database $D = (T_i^{\mathrm{r}})_{i \in K \subseteq \{1,\dots,n\}} \subseteq \mathcal{D}^{\mathrm{r}}$, we define a complement database $D' = (T_i^{\mathrm{r}})_{i \in \{1,\dots,n\} - K}$.

Given two databases $D_1, D_2 \subseteq \mathcal{D}^{\mathrm{r}}$ we call a multiset of items X^{r} a jumping emerging pattern with occurrence counts (occJEP) from D_1 to D_2, if $\mathrm{supp}_{D_1}(X^{\mathrm{r}}, 1) = 0\ \wedge\ \mathrm{supp}_{D_2}(X^{\mathrm{r}}, \theta) > 0$, where θ is the occurrence threshold. A set of all occJEPs with a threshold θ from D_1 to D_2 is called an occJEP space and denoted by

$occJEP(D_1, D_2, \theta)$. We distinguish the set of all minimal occJEPs as $occJEP_\mathrm{m}$, $occJEP_\mathrm{m}(D_1, D_2, \theta) \subseteq occJEP(D_1, D_2, \theta)$. Notice also that $occJEP(D_1, D_2, \theta) \subseteq occJEP(D_1, D_2, \theta - 1)$ for $\theta \geq 2$. In the rest of the paper we will refer to multisets of items as itemsets and use the symbol X^r to avoid confusion.

Example 3. Taking into consideration $D_1 = (T^\mathrm{r}_{1,2,3})$ and $D_2 = (T^\mathrm{r}_{4,5})$ from Table 1, the set of minimal occJEPs from D_1 to D_2 with threshold $\theta = 1$ is equal to: $\{\{1{\cdot}\mathit{black}\}, \{1{\cdot}\mathit{brown}\}, \{1{\cdot}\mathit{blue}, 4{\cdot}\mathit{white}\}, \{1{\cdot}\mathit{red}, 1{\cdot}\mathit{white}\}, \{1{\cdot}\mathit{white}, 2{\cdot}\mathit{yellow}\}\}$. Changing the threshold to $\theta = 2$ results in reducing the set of patterns to: $\{\{1 \cdot \mathit{black}\}, \{1 \cdot \mathit{brown}\}, \{1 \cdot \mathit{blue}, 4 \cdot \mathit{white}\}, \{1 \cdot \mathit{red}, 1 \cdot \mathit{white}\}\}$. This is because $\mathrm{supp}_{D_1}(\{1 \cdot \mathit{white}, 2 \cdot \mathit{yellow}\}, 1) = 0$ and $\mathrm{supp}_{D_2}(\{1 \cdot \mathit{white}, 2 \cdot \mathit{yellow}\}, 1) > 1$, but $\mathrm{supp}_{D_2}(\{1 \cdot \mathit{white}, 2 \cdot \mathit{yellow}\}, 2) = 0$.

Occurrence Threshold The introduction of an occurrence threshold θ allows for differentiating transactions containing the same sets of items with a specified tolerance margin of occurrence counts. It is thus possible to define a difference in the number of occurrences, which is necessary to consider such a pair of transactions as distinct sets of items.

For the example image database given by Table 1 we can see that the differences between counts of such items as *white* and *yellow* may be too small to assume they represent a general pattern present in the database that would allow building a classifier. Setting the threshold to a higher value results in a smaller number of patterns, but the discovered ones have a greater confidence.

5 A Semi-Naïve Mining Algorithm

Our previous method of discovering occJEPs, introduced in [17], is based on the observation that only minimal patterns need to be found to perform classification. Furthermore, it is usually not necessary to mine patterns longer than a few items, as their support is very low and thus their impact on classification accuracy is negligible. This way we can reduce the problem to: (a) finding only such occJEPs, for which no patterns with a lesser number of items and the same or lower number of item occurrences exist; (b) discovering patterns of less than δ items.

Let C_c be a decision class sequence of a database $\mathcal{D}^\mathrm{r}$ for a given decision item c and C'_c a complement sequence to C_c. We define $D_1 = C'_c$, $D_2 = C_c$ and the aim of the algorithm to discover $occJEP_\mathrm{m}(D_1, D_2, \theta)$. We begin by finding the patterns, which are not supported in D_1, as possible candidates for occJEPs. In case of multi-item patterns at least one of the item counts of the candidate pattern has to be larger than the corresponding item count in the database. We can write this as:

$$X^\mathrm{r} = (X, P) \text{ is an occJEP candidate } \iff \forall_{T^\mathrm{r}=(T,Q)\in D_1} \exists_{x\in X}\ p(x) > q(x).$$

Table 2 shows an example set of conditions for single and multi-item occJEP candidates.

Table 2. Finding occJEPs in a transaction database with recurrent items. Example conditions for single-item patterns and patterns consisting of two items.

D_1	$q(i_1)$	$q(i_2)$	$q(i_3)$
T_1^r	3	12	3
T_2^r	0	16	11
T_3^r	5	19	4
T_4^r	2	14	13

$cond(p(i_1))$	$cond(p(i_2), p(i_3))$
$p(i_1) > 3$	$p(i_2) > 12 \vee p(i_3) > 3$
$p(i_1) > 0$	$p(i_2) > 16 \vee p(i_3) > 11$
$p(i_1) > 5$	$p(i_2) > 19 \vee p(i_3) > 4$
$p(i_1) > 2$	$p(i_2) > 14 \vee p(i_3) > 13$

The first step of the algorithm is then to create a set of conditions in the form of $[p(i_j) > q_1(i_j) \vee \ldots \vee p(i_k) > q_1(i_k)] \wedge \ldots \wedge [p(i_j) > q_n(i_j) \vee \ldots \vee p(i_k) > q_n(i_k)]$ for each of the candidate itemsets $X^r = (X, P),\ X \subseteq 2^{\mathcal{I}}$, where j and k are subscripts of items appearing in a particular X^r and n is the number of transactions in D_1. Solving this set of inequalities results in its transformation to the form of $[p(i_j) > r_j \wedge \ldots \wedge p(i_k) > r_k] \vee \ldots \vee [p(i_j) > s_j \wedge \ldots \wedge p(i_k) > s_k]$, where r and s are the occurrence counts of respective items. The counts have to be incremented by 1, to fulfill the condition of $\mathrm{supp}_{D_1}(X^r, \theta) = 0$.

Example 4. For the previously introduced example of D_1 from Table 2, we can see that $cond(p(i_1))$ resolves to $p(i_1) > 5$ and $cond(p(i_1), p(i_2))$ to $p(i_2) > 19 \ \vee\ p(i_3) > 13 \ \vee\ (p(i_2) > 14 \ \wedge\ p(i_3) > 11) \ \vee\ (p(i_2) > 16 \ \wedge\ p(i_3) > 4)$. Notice that resolved conditions for pattern length l also contain all conditions for $l - 1$. The resulting candidates for minimal patterns, after incrementing the occurrence counts, are thus the following: $X_1^r = \{6 \cdot i_1\}$, $X_2^r = \{20 \cdot i_2\}$, $X_3^r = \{14 \cdot i_3\}$, $X_4^r = \{15 \cdot i_2, 12 \cdot i_3\}$, $X_5^r = \{17 \cdot i_2, 5 \cdot i_3\}$.

Having found the minimum occurrence counts of items in the candidate itemsets, we then calculate the support of each of the itemsets in D_2 with a threshold θ. The candidates, for which $\mathrm{supp}_{D_2}(X, \theta) > 0$ are the minimal $occJEPs(D_1, D_2, \theta)$.

Example 5. Continuing the above example, we see from Table 3 that the support of candidate patterns $\mathrm{supp}_{D_2}(X_1^r, 3) > 0$, $\mathrm{supp}_{D_2}(X_3^r, 2) = 0$ and $\mathrm{supp}_{D_2}(X_5^r, 1) > 0$. X_1^r and X_5^r are thus minimal occJEPs with threshold values $\theta = 3$ and $\theta = 1$ respectively. By the definition of occJEPs, X_1^r is also an occJEP for $\theta \in [1,\ 3]$. Other minimal occJEPs are X_3^r and X_4^r for $\theta = 1$, as their respective supports in D_2 are equal to 1/4.

6 Border-Based Mining Algorithm

The border-based occJEP discovery algorithm is an extension of the EP-mining method described in [3]. Similarly, as proved in [7] for regular emerging patterns, we can use the concept of borders to represent a collection of occJEPs. This is because the occJEP space S is convex, that is it follows: $\forall X^r, Z^r \in S^r\ \forall Y^r \in 2^{S^r}\ X^r \subseteq Y^r \subseteq Z^r \Rightarrow Y^r \in S^r$. For the sake of readability we will now onward

Table 3. Finding occJEPs in a transaction database with recurrent items. Calculating the support count of candidate itemsets in complementary database.

D_2	$p(i_1)$	$p(i_2)$	$p(i_3)$
T_1^{r}	11	15	4
T_2^{r}	18	16	12
T_3^{r}	12	17	5
T_4^{r}	23	14	14

$\phi(X_1^{\mathrm{r}}, T^{\mathrm{r}}, 3)$	$\phi(X_3^{\mathrm{r}}, T^{\mathrm{r}}, 2)$	$\phi(X_5^{\mathrm{r}}, T^{\mathrm{r}}, 1)$
0	0	0
1	0	0
0	0	1
1	0	0
$\mathrm{supp}_{D_2} = 1/2$	$\mathrm{supp}_{D_2} = 0$	$\mathrm{supp}_{D_2} = 1/4$

denote particular items with consecutive alphabet letters, with an index indicating the occurrence count, and skip individual brackets, e.g. $\{a_1b_2, c_3\}$ instead of $\{\{1 \cdot i_1, 2 \cdot i_2\}, \{3 \cdot i_3\}\}$.

Example 6. $\mathcal{S} = \{a_1, a_1b_1, a_1b_2, a_1c_1, a_1b_1c_1, a_1b_2c_1\}$ is a convex collection of sets, but $\mathcal{S}' = \{a_1, a_1b_1, a_1c_1, a_1b_1c_1, a_1b_2c_1\}$ is not convex. We can partition it into two convex collections $\mathcal{S}_1' = \{a_1, a_1b_1\}$ and $\mathcal{S}_2' = \{a_1c_1, a_1b_1c_1, a_1b_2c_1\}$.

A border is an ordered pair $< \mathcal{L}, \mathcal{R} >$ such that $\mathcal{L}$ and $\mathcal{R}$ are antichains, $\forall X^{\mathrm{r}} \in \mathcal{L}\ \exists Y^{\mathrm{r}} \in \mathcal{R}\ X^{\mathrm{r}} \subseteq Y^{\mathrm{r}}$ and $\forall X^{\mathrm{r}} \in \mathcal{R}\ \exists Y^{\mathrm{r}} \in \mathcal{L}\ Y^{\mathrm{r}} \subseteq X^{\mathrm{r}}$. The collection of sets represented by a border $< \mathcal{L}, \mathcal{R} >$ is equal to:

$$[\mathcal{L}, \mathcal{R}] = \{Y^{\mathrm{r}} : \exists X^{\mathrm{r}} \in \mathcal{L}, \exists Z^{\mathrm{r}} \in \mathcal{R} \text{ such that } X^{\mathrm{r}} \subseteq Y^{\mathrm{r}} \subseteq Z^{\mathrm{r}}\}. \tag{5}$$

Example 7. The border of collection $\mathcal{S}$, introduced in earlier example, is equal to $[\mathcal{L}, \mathcal{R}] = [\{a_1\}, \{a_1b_2c_1\}]$.

The most basic operation involving borders is a border differential, defined as:

$$< \mathcal{L}, \mathcal{R} > = < \{\emptyset\}, \mathcal{R}_1 > - < \{\emptyset\}, \mathcal{R}_2 > . \tag{6}$$

As proven in [7] this operation may be reduced to a series of simpler operations. For $\mathcal{R}_1 = \{U_1, \ldots, U_m\}$:

$$< \mathcal{L}_i, \mathcal{R}_i > = < \{\emptyset\}, \{U_i^{\mathrm{r}}\} > - < \{\emptyset\}, \mathcal{R}_2 > . \tag{7}$$

$$< \mathcal{L}, \mathcal{R} > = < \bigcup_{i=1}^{m} \mathcal{L}_i, \bigcup_{i=1}^{m} \mathcal{R}_i > . \tag{8}$$

A direct approach to calculating the border differential would be to expand the borders and compute set differences.

Example 8. The border differential between $[\{\emptyset\}, \{a_1b_2c_1\}]$ and $[\{\emptyset\}, \{a_1c_1\}]$ is equal to $[\{b_1\}, \{a_1b_2c_1\}]$. This is because:

$$[\{\emptyset\}, \{a_1b_2c_1\}] = \{a_1, b_1, b_2, c_1, a_1b_1, a_1b_2, a_1c_1, b_1c_1, b_2c_1, a_1b_1c_1, a_1b_2c_1\}$$
$$[\{\emptyset\}, \{a_1c_1\}] = \{a_1, c_1, a_1c_1\}$$
$$[\{\emptyset\}, \{a_1b_2c_1\}] - [\{\emptyset\}, \{a_1c_1\}] = \{b_1, b_2, a_1b_1, a_1b_2, b_1c_1, b_2c_1, a_1b_1c_1, a_1b_2c_1\}$$

6.1 Algorithm optimizations

On the basis of optimizations proposed in [3], we now show the extensions necessary for discovering emerging patterns with occurrence counts. All of the ideas presented there for reducing the number of operations described in the context of regular EPs are also applicable for recurrent patterns. The first idea allows avoiding the expansion of borders when calculating the collection of minimal itemsets $\mathrm{Min}(\mathcal{S})$ in a border differential $\mathcal{S} = [\{\emptyset\}, \{U^{\mathrm{r}}\}] - [\{\emptyset\}, \{S_1^{\mathrm{r}}, \ldots, S_k^{\mathrm{r}}\}]$. It has been proven in [3] that $\mathrm{Min}(\mathcal{S})$ is equivalent to:

$$\mathrm{Min}(\mathcal{S}) = \mathrm{Min}(\{\bigcup\{s_1, \ldots, s_k\} : s_i \in U^{\mathrm{r}} - S_i^{\mathrm{r}}, 1 \leq i \leq k\}). \tag{9}$$

In the case of emerging patterns with occurrence counts we need to define the left-bound union and set theoretic difference operations between multisets of items $X^{\mathrm{r}} = (X,\ P)$ and $Y^{\mathrm{r}} = (Y,\ Q)$. These operations guarantee that the resulting patterns are still minimal.

Definition 1. *The left-bound union of multisets* $X^{\mathrm{r}} \cup Y^{\mathrm{r}} = Z^{\mathrm{r}}$. $Z^{\mathrm{r}} = (Z, R)$, *where:* $Z = \{z : z \in X \vee z \in Y\}$ *and* $R = \{r(z) = max(p(z), q(z))\}$.

Definition 2. *The left-bound set theoretic difference of multisets* $X^{\mathrm{r}} - Y^{\mathrm{r}} = Z^{\mathrm{r}}$. $Z^{\mathrm{r}} = (Z, R)$, *where:* $Z = \{z : z \in X \wedge p(z) > q(z)\}$ *and* $R = \{r(z) = q(z) + 1\}$.

Example 9. For the differential: $[\{\emptyset\}, \{a_1b_3c_1d_1\}] - [\{\emptyset\}, \{b_1c_1\}, \{b_3d_1\}, \{c_1d_1\}]$. $U = \{a_1b_3c_1d_1\}$, $S_1 = \{b_1c_1\}$, $S_2 = \{b_3d_1\}$, $S_3 = \{c_1d_1\}$. $U - S_1 = \{a_1b_2d_1\}$, $U - S_2 = \{a_1c_1\}$, $U - S_3 = \{a_1b_1\}$. Calculating the Min function:

$$\begin{aligned}
&\mathrm{Min}([\{\emptyset\}, \{a_1b_3c_1d_1\}] - [\{\emptyset\}, \{b_1c_1\}, \{b_3d_1\}, \{c_1d_1\}]) = \\
&= \mathrm{Min}(\{a_1a_1a_1, a_1a_1b_1, a_1c_1a_1, a_1c_1b_1, b_2a_1a_1, \\
&\quad b_2a_1b_1, b_2c_1b_1, d_1a_1a_1, d_1a_1b_1, d_1c_1a_1, d_1c_1b_1\}) = \\
&= \mathrm{Min}(\{a_1, a_1b_1, a_1c_1, a_1b_1c_1, a_1b_2, a_1b_2, b_2c_1, a_1d_1, a_1b_1d_1, a_1c_1d_1, b_1c_1d_1\}) = \\
&= \{a_1, b_2c_1, b_1c_1d_1\}\ .
\end{aligned}$$

Similar changes are necessary when performing the border expansion in an incremental manner, which has been proposed as the second possible algorithm optimization. The union and difference operations in the following steps need to be conducted according to Definitions 1 and 2 above, see Algorithm 1.

Lastly, a few points need to be considered when performing the third optimization, namely avoiding generating nonminimal itemsets. Originally, the idea was to avoid expanding such itemsets during incremental processing, which are known to be minimal beforehand. This is the case when the same item is present both in an itemset in the old $\mathcal{L}$ and in the set difference $U - S_i$ (line 3 of the incremental expansion algorithm above). In case of recurrent patterns this condition is too weak to guarantee that all patterns are still going to be generated, as we have to deal with differences in the number of item occurrences. The modified conditions of itemset removal are thus as follows:

Algorithm 1: Incremental expansion

```
Input  : $U^{\mathrm{r}}$, $S^{\mathrm{r}}_i$
Output: $\mathcal{L}$
1 $\mathcal{L} \longleftarrow \{\{x\} : x \in U^{\mathrm{r}} - S^{\mathrm{r}}_1\}$
2 for $i = 2$ to $k$ do
3   $\mathcal{L} \longleftarrow \mathrm{Min}\{X^{\mathrm{r}} \cup \{x\} : X^{\mathrm{r}} \in \mathcal{L},\ x \in U^{\mathrm{r}} - S^{\mathrm{r}}_i\}$
4 end
```

1. If an itemset X^{r} in the old $\mathcal{L}$ contains an item x from $T^{\mathrm{r}}_i = U^{\mathrm{r}} - S^{\mathrm{r}}_i$ and its occurrence count is equal or greater than the one in T^{r}_i, then move X^{r} from $\mathcal{L}$ to $NewL$.
2. If the moved X^{r} is a singleton set $\{(x, p(x))\}$ and its occurrence count is the same in $\mathcal{L}$ and T^{r}_i, then remove x from T^{r}_i.

Example 10. Let $U^{\mathrm{r}} = \{a_1 b_2\}$, $S^{\mathrm{r}}_1 = \{a_1\}$, $S^{\mathrm{r}}_2 = \{b_1\}$. Then $T^{\mathrm{r}}_1 = U^{\mathrm{r}} - S^{\mathrm{r}}_1 = \{b_1\}$ and $T^{\mathrm{r}}_2 = U^{\mathrm{r}} - S^{\mathrm{r}}_2 = \{a_1 b_2\}$. We initialize $\mathcal{L} = \{b_1\}$ and check it against T^{r}_2. While T^{r}_2 contains $\{b_2\}$, $\{b_1\}$ may not be moved directly to $NewL$, as this would falsely result in returning $\{b_1\}$ as the only minimal itemset, instead of $\{a_1 b_1, b_2\}$. Suppose $S^{\mathrm{r}}_1 = \{a_1 b_1\}$, then initial $\mathcal{L} = \{b_2\}$ and this time we can see that $\{b_2\}$ does not have to be expanded, as the same item with at least equal occurrence count is present in T^{r}_2. Thus, $\{b_2\}$ is moved directly to $NewL$, removed from T^{r}_2 and returned as a minimal itemset.

The final algorithm, consisting of all proposed modifications, is presented below as Algorithm 2.

6.2 Discovering occJEPs

Creating an occJEP-based classifier involves discovering all minimal occJEPs to each of the classes present in a particular decision system. We can formally define the set of patterns in a classifier $occJEP^{\theta}_{\mathrm{C}}$ for a given occurrence threshold θ as: $occJEP^{\theta}_{\mathrm{C}} = \bigcup_{c \in \mathcal{I}_{\mathrm{d}}} occJEP_{\mathrm{m}}(C'_c, C_c, \theta)$, where $C_c \subseteq \mathcal{D}^{\mathrm{r}}_L$ is a decision class sequence for decision item c and C'_c is a complementary sequence in a learning database $\mathcal{D}^{\mathrm{r}}_L$.

To discover patterns between two dataset pairs, we first need to remove non-maximal itemsets from each them. Next, we multiply the occurrence counts of itemsets in the background dataset by the user-specified threshold. Finally, we need to iteratively call the Border-differential function and create a union of the results to find the set of all minimal jumping emerging patterns with occurrence counts from C'_c to C_c (see Algorithm 3).

Example 11. Consider a learning database $\mathcal{D}^{\mathrm{r}}_L$ containing transactions of three distinct classes: $C_1, C_2, C_3 \subset \mathcal{D}^{\mathrm{r}}_L$. $C_1 = \{b_2, a_1 c_1\}$, $C_2 = \{a_1 b_1, c_3 d_1\}$ and $C_3 = \{a_3, b_1 c_1 d_1\}$. We need to discover occJEPs to each of the decision class sequences: $occJEP_{\mathrm{m}}(C_2 \cup C_3, C_1, \theta)$, $occJEP_{\mathrm{m}}(C_1 \cup C_3, C_2, \theta)$ and $occJEP_{\mathrm{m}}(C_1 \cup C_2, C_3, \theta)$.

Algorithm 2: Border differential

```
Input  : $< \{\emptyset\}, \{U^{\mathrm{r}}\} >, < \{\emptyset\}, \{S^{\mathrm{r}}_1, \ldots, S^{\mathrm{r}}_k\} >$
Output: $\mathcal{L}$
1  $T^{\mathrm{r}}_i \longleftarrow U^{\mathrm{r}} - S^{\mathrm{r}}_i$ for $1 \leq i \leq k$
2  if $\exists T^{\mathrm{r}}_i = \{\emptyset\}$ then
3      return $< \{\}, \{\} >$
4  end
5  $\mathcal{L} \longleftarrow \{\{x\} : x \in T^{\mathrm{r}}_1\}$
6  for $i = 2$ to $k$ do
7      $NewL \longleftarrow \{X^{\mathrm{r}} = (X, P(X)) \in \mathcal{L} : X \cap T_i \neq \emptyset \wedge \forall x \in (X \cap T_i)\ p(x) \geq t(x)\}$
8      $\mathcal{L} \longleftarrow \mathcal{L} - NewL$
9      $T^{\mathrm{r}}_i \longleftarrow T^{\mathrm{r}}_i - \{x : \{(x, p(x))\} \in NewL\}$
10     foreach $X^{\mathrm{r}} \in \mathcal{L}$ sorted according to increasing cardinality do
11         foreach $x \in T_i$ do
12             if $\forall Z^{\mathrm{r}} \in NewL$ $\mathrm{supp}_{Z^{\mathrm{r}}}(X^{\mathrm{r}} \cup \{x\}, 1) = 0$ then
13                 $NewL \longleftarrow NewL \cup (X^{\mathrm{r}} \cup \{x\})$
14             end
15         end
16     end
17     $\mathcal{L} \longleftarrow NewL$
18 end
```

Suppose $\theta = 2$. Calculating the set of all minimal patterns involves invoking the Discover-minimal-occJEPs function three times, in which the base Border-differential function is called twice each time and the resulting occJEPs are as follows: $\{a_1c_1\}$ to class 1, $\{c_3, a_1b_1\}$ to class 2 and $\{a_3, b_1c_1, b_1d_1\}$ to class 3.

7 Performing Classification

Classification of a particular transaction in the testing database $\mathcal{D}^{\mathrm{r}}_{\mathrm{T}}$ is performed by aggregating all minimal occJEPs, which are supported by it [9]. A scoring function is calculated and a category label is chosen by finding the class with the maximum score:

$$\mathrm{score}(T^{\mathrm{r}}, c) = \sum_{X^{\mathrm{r}}} \mathrm{supp}_{C_c}(X^{\mathrm{r}}), \tag{10}$$

where $C_c \subseteq \mathcal{D}^{\mathrm{r}}_{\mathrm{T}}$ and $X^{\mathrm{r}} \in occJEP_{\mathrm{m}}(C'_c, C_c)$, such that $X^{\mathrm{r}} \subseteq T^{\mathrm{r}}$. It is possible to normalize the score to reduce the bias induced by unequal sizes of particular decision sequences. This is performed by dividing the calculated score by a normalization factor: norm-score(T^{r}, c) = score(T^{r}, c)/base-score(c), where base-score is the median of scores of all transactions with decision item c in the learning database: base-score$(c) = \mathrm{median}\{\mathrm{score}(T^{\mathrm{r}}, c),$ for each $T^{\mathrm{r}} \in C_c \subseteq \mathcal{D}^{\mathrm{r}}_{\mathrm{L}}\}$.

Algorithm 3: Discover minimal occJEPs

```
  Input  : $C'_c$, $C_c$, $\theta$
  Output: $\mathcal{J}$
1 for $S^{\mathrm{r}}_i \in \mathcal{R}$ do
2     $S^{\mathrm{r}}_i \longleftarrow (S_i, s(x) \cdot \theta)$
3 end
4 $\mathcal{J} \longleftarrow \{\emptyset\}$
5 for $L^{\mathrm{r}}_i \in \mathcal{L}$ do
6     $\mathcal{J} \longleftarrow \mathcal{J} \cup$ Border-differential$(< \{\emptyset\}, \{L^{\mathrm{r}}_i\} >, < \{\emptyset\}, \{S^{\mathrm{r}}_1, \ldots, S^{\mathrm{r}}_k\} >)$
7 end
```

8 Experimental Results

We have used two types of data with recurrent items to assess the performance of the proposed classifier. The first is a dataset used previously in [17], which consists of images, represented by texture and color features, classified into four categories: *flower*, *food*, *mountain* and *elephant*. The data contains ca. 400 instances and 16 recurrent attributes, where each instance is an image represented by 8 types of texture and 8 types of color features, possibly occurring multiple times on a single image. The second dataset used for experiments represents the problem of text classification and has been generated on the basis of the Reuters-21578 collection of documents.

8.1 Image Dataset

The image dataset is a collection of images made available by the authors of the SIMPLIcity CBIR system [18], consisting of 1 000 photographs, which are JPEG color image files, having a resolution of 384×256 pixels. An example selection of photographs is presented on Figure 1.

Fig. 1. Example images from the SIMPLIcity test database

Feature Representation We have used a tile-based, symbolic representation of photographs to enable using classification methods developed for transactional data in the domain of images. The images are uniformly divided into a grid of $x \times y$ tiles, where x is the number of rows and y is the number of columns, and for each of the tiles the color and texture features are calculated.

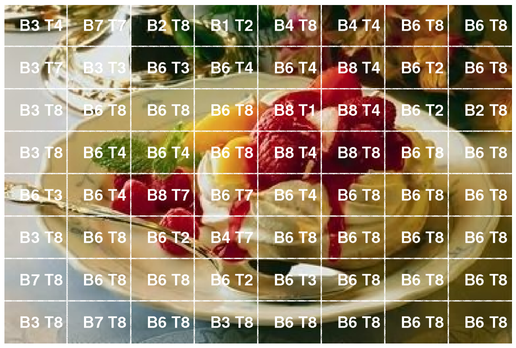
Color features are represented by a histogram calculated in the HSV color space, with the hue channel quantized to h discrete ranges, while saturation and value channels to s and v ranges respectively. In effect, the representation takes the form of a $h \times s \times v$ element vector of real values between 0 and 1. For the representation of texture we use a feature vector consisting of mean and standard deviation values calculated from the result of filtering an original image with a bank of Gabor functions. These filters are scaled and rotated versions of the base function, which is a product of a Gaussian and a sine function. By using m orientations and n different scales we get a feature vector consisting of mean (μ) and standard deviation (σ) values of each of the filtered images and thus having a size of $2 \times m \times n$ values.

In the next step we aggregate all calculated image features and employ a clustering algorithm to reduce the number of values into a chosen number of groups. In this way, we create a dictionary that consists of the most representative color and texture features of the images in the learning set. The clustering is performed using the k-Means algorithm with a histogram intersection measure for comparing color feature vectors f_c and Gabor feature distance for comparing texture feature vectors f_t. Centroids resulting from the clustering operation become the elements of the dictionary and are labeled $B_1, \dots, B_k$ in case of color and $T_1, \dots, T_k$ in case of texture features, where k is the feature dictionary size. These identifiers are then used to describe the images in the database by associating an appropriate label with every tile of each image. This is performed by finding the closest centroid to a feature vector calculated for a given image tile, using appropriate distance measures for each of the features. The dictionary created for the learning set is reused during the classification phase.

Figure 2 presents an example of such a symbolic image representation, showing labels of its individual tiles and the representation of the whole image as a binary database transaction and a database transaction with recurrent items. Figure 3 compares and contrasts representations of images belonging to two different categories: *flower* and *food*.

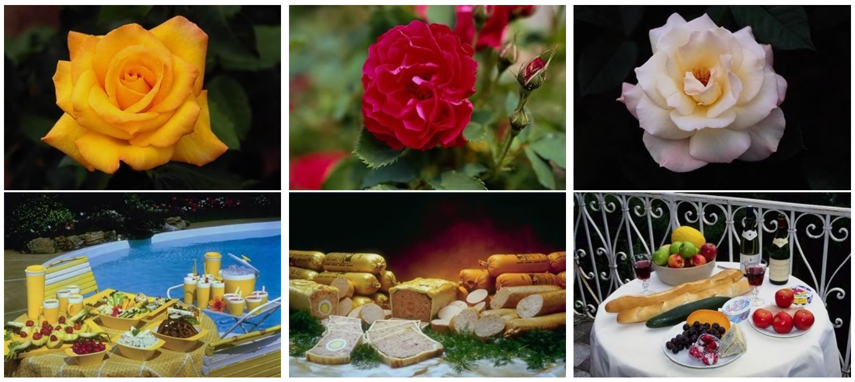
Results We have used the following parameter values for the experiments: images partitioned into 8×8 tiles ($x = y = 8$), the sizes of feature vectors $|f_c| = 162$ ($h = 18$, $s = 3$, $v = 3$) and $|f_t| = 48$ ($m = 6$, $n = 4$) values. The dictionary size was set at $k = 8$ values. The parameters are dataset dependent: the number of tiles should be chosen based on the resolution of analyzed images and the dictionary size reflects the diversity of the dataset.

The accuracy achieved by applying the classifier based on jumping emerging patterns with occurrence counts for several threshold values and compared with other frequently used classification methods is presented in Table 4. All experiments have been conducted as a ten-fold cross-validation using the Weka


B3 T4 B7 T7 B2 T8 B1 T2 B4 T8 B4 T4 B6 T8 B6 T8
B3 T7 B3 T3 B6 T3 B6 T4 B6 T4 B8 T4 B6 T2 B6 T8
B3 T8 B6 T8 B6 T8 B6 T8 B8 T1 B8 T4 B6 T2 B2 T8
B3 T8 B6 T4 B6 T4 B6 T8 B8 T4 B8 T8 B6 T8 B6 T8
B6 T3 B6 T4 B8 T7 B6 T7 B6 T4 B6 T8 B6 T8 B6 T8
B3 T8 B6 T8 B6 T2 B4 T7 B6 T8 B6 T8 B6 T8 B6 T8
B7 T8 B6 T8 B6 T8 B6 T2 B6 T3 B6 T8 B6 T8 B6 T8
B3 T8 B7 T8 B6 T8 B3 T8 B6 T8 B6 T8 B6 T8 B6 T8

DB	Representation
$\mathcal{D}$	$B_1,\ B_2,\ B_3,\ B_4,$ $B_6,\ B_7,\ B_8$ $T_1,\ T_2,\ T_3,\ T_4,$ $T_7,\ T_8$
$\mathcal{D}^{\mathrm{r}}$	$1 \cdot B_1,\ 2 \cdot B_2,\ 8 \cdot B_3,\ 3 \cdot B_4,$ $41 \cdot B_6,\ 3 \cdot B_7,\ 6 \cdot B_8$ $1 \cdot T_1,\ 5 \cdot T_2,\ 4 \cdot T_3,\ 11 \cdot T_4,$ $5 \cdot T_7,\ 38 \cdot T_8$

Fig. 2. A symbolic representation of an image from the *food* dataset. $\mathcal{D}$ – binary transaction system, $\mathcal{D}^{\mathrm{r}}$ – transaction system with recurrent items.



Database	Representation
D_1	$T_1^{\mathrm{r}} = \{37 \cdot B_2,\ 15 \cdot B_3,\ 1 \cdot B_5,\ 11 \cdot B_7,\ 1 \cdot T_1,\ 59 \cdot T_2,\ 4 \cdot T_4\}$ $T_2^{\mathrm{r}} = \{2 \cdot B_1,\ 8 \cdot B_2,\ 25 \cdot B_4,\ 4 \cdot B_6,\ 24 \cdot B_7,\ 1 \cdot B_8,\ 58 \cdot T_2,\ 5 \cdot T_4,\ 1 \cdot T_7\}$ $T_3^{\mathrm{r}} = \{4 \cdot B_2,\ 36 \cdot B_3,\ 6 \cdot B_5,\ 6 \cdot B_6,\ 12 \cdot B_8,\ 1 \cdot T_1,\ 52 \cdot T_2,\ 11 \cdot T_4\}$
D_2	$T_4^{\mathrm{r}} = \{34 \cdot B_2,\ 10 \cdot B_3,\ 4 \cdot B_5,\ 3 \cdot B_7,\ 13 \cdot B_8,$ $1 \cdot T_1,\ 18 \cdot T_2,\ 3 \cdot T_3,\ 33 \cdot T_4,\ 9 \cdot T_6\}$ $T_5^{\mathrm{r}} = \{4 \cdot B_1,\ 17 \cdot B_2,\ 8 \cdot B_3,\ 3 \cdot B_5,\ 12 \cdot B_6,\ 12 \cdot B_7,\ 8 \cdot B_8,$ $4 \cdot T_1,\ 21 \cdot T_2,\ 38 \cdot T_4,\ 1 \cdot T_6\}$ $T_6^{\mathrm{r}} = \{1 \cdot B_1,\ 11 \cdot B_2,\ 9 \cdot B_3,\ 2 \cdot B_4,\ 10 \cdot B_5,\ 2 \cdot B_6,\ 9 \cdot B_7,\ 20 \cdot B_8,$ $3 \cdot T_1,\ 9 \cdot T_2,\ 6 \cdot T_3,\ 13 \cdot T_4,\ 7 \cdot T_5,\ 4 \cdot T_6,\ 3 \cdot T_7,\ 19 \cdot T_8\}$

Fig. 3. Examples of *flower* (upper row, D_1) and *food* (bottom row, D_2) images, along with their symbolic representation (numbered from left to right).

package [19], having discretized the data into 10 equal-frequency bins for all algorithms, except the occJEP method. The parameters of all used classifiers have been left at their default values. The results are not directly comparable with those presented in [17], as currently the occJEP patterns are not limited to any specific length and the seed number for random instance selection during cross-validation was different than before.

Table 4. Classification accuracy of four image datasets. The performance of the classifier based on jumping emerging patterns with occurrence counts (occJEP) compared to: regular jumping emerging patterns (JEP), C4.5 and support vector machine (SVM), each after discretization into 10 equal-frequency bins.

method	θ	accuracy (%)					
		flower/ food	*flower/ elephant*	*flower/ mountain*	*food/ elephant*	*food/ mountain*	*elephant/ mountain*
occJEP	1	89.50	84.38	90.63	-	73.00	-
	1.5	94.79	96.35	98.44	78.50	87.00	87.50
	2	**97.92**	**98.96**	**97.92**	88.00	91.00	**88.50**
	2.5	92.71	97.92	95.31	83.00	90.50	85.50
	3	89.06	97.92	95.31	74.00	87.00	80.50
JEP	-	95.83	91.67	96.35	**88.50**	**93.50**	83.50
C4.5	-	93.23	89.58	85.94	87.50	92.50	82.00
SVM	-	90.63	91.15	93.75	87.50	84.50	84.50

8.2 Text Dataset

We have used the ApteMod version of the Reuters corpus [20], which originally contains 10788 documents classified into 90 categories, to assess the performance of our classifier. As the categories are highly imbalanced (the most common class contains 3937 documents, while the least common only 1), we have presented here the results of classification of the problem reduced to differentiating between the two classes with the greatest number of documents and all other combined, i.e. the new category labels are *earn* (36.5% of all instances), *acq* (21.4%) and *other* (42.1%).

Feature Representation Document representation has been generated by: stemming each word in the corpus using the Porter's stemmer, ignoring words, which appear on the stoplist provided with the corpus, and finally creating a vector containing the number of occurrences of words in the particular document.

Results We have selected the 100 most relevant attributes from the resulting data, as measured by the χ^2 statistic, and sampled randomly 215 instances for cross-validation experiments, the results of which are presented in Table 5.

Table 5. Classification accuracy of the Reuters dataset, along with precision and recall values for each of the classes, and the number of discovered emerging patterns / C4.5 tree size

			earn		*acq*		*other*		
method	θ	accuracy	precision	recall	precision	recall	precision	recall	patterns
occJEP	1	85.12	96.2	84.7	76.6	96.1	95.5	89.4	10029
	1.5	85.58	96.2	84.7	77.8	96.1	95.5	90.4	9276
	2	84.65	96.2	87.9	78.9	95.7	**96.6**	91.5	7274
	2.5	84.19	96.2	**87.9**	77.6	95.7	96.6	90.4	7015
	10	83.72	98.00	86.2	79.3	**97.9**	95.5	91.3	3891
JEP	-	66.98	86.8	55.0	46.2	47.1	70.7	85.3	45870
C4.5	-	73.49	92.9	65.0	67.5	52.9	69.2	88.5	51
SVM	-	**86.98**	**98.1**	85.0	**85.4**	68.6	82.8	**97.1**	-

9 Conclusions and Future Work

We have proposed an extension of the border-based emerging patterns mining algorithm to allow discovering jumping emerging patterns with occurrence counts. Such patterns may be used to build accurate classifiers for transactional data containing recurrent attributes. By avoiding both discretization and using all values from the attribute domain, we considerably reduce the space of items and exploit the natural order of occurrence counts. We have shown that areas that could possibly benefit from using such an approach include image and text data classification.

The presented results show that the proposed classifier may achieve equal or better performance than well-known tree-based C4.5 algorithm and support vector machines (SVMs). The approach is most promising in the area of multimedia data mining, as it allows reasoning about quantitative features of images and possibly – after further research – also about their spatial relationships.

Another advantage of using a pattern-based classifier over other algorithms is the ability to analyze the created classifier, which describes the differences between two sets of data in a way easily understandable by a human. This greater insight into problem domain is an important point in many applications, e.g. bioinformatics.

The biggest drawback of the method lies in the number of discovered patterns, which is however less than in the case of regular JEPs found in discretized data. It is thus a possible area of future work to reduce the set of discovered patterns and further limit the computational complexity without influencing the classification accuracy.

References

1. Dong, G., Li, J.: Efficient mining of emerging patterns: Discovering trends and differences. In: KDD '99: Proceedings of the fifth ACM SIGKDD international

conference on Knowledge discovery and data mining, New York, NY, USA, ACM (1999) 43–52

2. Dong, G., Zhang, X., Wong, L., Li, J.: CAEP: Classification by aggregating emerging patterns. In Arikawa, S., Furukawa, K., eds.: Proceedings of Second International Conference on Discovery Science. Volume 1721 of Lecture Notes in Computer Science., Springer-Verlag (1999) 30–42
3. Dong, G., Li, J.: Mining border descriptions of emerging patterns from dataset pairs. Knowledge and Information Systems **8**(2) (2005) 178–202
4. Li, J., Dong, G., Ramamohanarao, K.: Instance-based classification by emerging patterns. In: Proceedings of 4th European Conference on Principles of Data Mining and Knowledge Discovery, PKDD 2000. Lecture Notes in Computer Science, Springer-Verlag (2000) 191–200
5. Li, J., Dong, G., Ramamohanarao, K., Wong, L.: DeEPs: A new instance-based lazy discovery and classification system. Machine Learning **54**(2) (2004) 99–124
6. Li, J., Dong, G., Ramamohanarao, K.: Making use of the most expressive jumping emerging patterns for classification. Knowledge and Information Systems **3**(2) (2001) 1–29
7. Li, J., Ramamohanarao, K., Dong, G.: The space of jumping emerging patterns and its incremental maintenance algorithms. In: ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2000) 551–558
8. Fan, H., Ramamohanarao, K.: An efficient single-scan algorithm for mining essential jumping emerging patterns for classification. In Cheng, M.S., Yu, P.S., Liu, B., eds.: Pacific-Asia Conference on Knowledge Discovery and Data Mining. Volume 2336 of Lecture Notes in Computer Science., Springer-Verlag (2002) 456–462
9. Fan, H., Ramamohanarao, K.: Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers. IEEE Transactions on Knowledge and Data Engineering **18**(6) (2006) 721–737
10. Terlecki, P., Walczak, K.: On the relation between rough set reducts and jumping emerging patterns. Information Sciences **177**(1) (2007) 74–83
11. Li, J., Liu, G., Wong, L.: Mining statistically important equivalence classes and delta-discriminative emerging pattern. In: Proceedings of 13th International Conference on Knowledge Discovery and Data Mining, San Jose, California (2007) 430–439
12. Li, J., Wong, L.: Emerging patterns and gene expression data. Genome Informatics **12** (2001) 3–13
13. Li, J., Wong, L.: Identifying good diagnostic gene groups from gene expression profiles using the concept of emerging patterns. Bioinformatics **18** (2002) 725–734
14. Zaïane, O.R., Han, J., Zhu, H.: Mining recurrent items in multimedia with progressive resolution refinement. In: Proceedings of the 16th International Conference on Data Engineering, San Diego, CA, USA (2000) 461–470
15. Ong, K.L., Ng, W.K., Lim, E.P.: Mining multi-level rules with recurrent items using FP'-Tree. In: Proceedings of the Third International Conference on Information, Communications and Signal Processing. (2001)
16. Rak, R., Kurgan, L.A., Reformat, M.: A tree-projection-based algorithm for multi-label recurrent-item associative-classification rule generation. Data and Knowledge Engineering **64**(1) (2008) 171–197
17. Kobyliński, Ł., Walczak, K.: Jumping emerging patterns with occurrence count in image classification. In Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A., eds.: Pacific-Asia Conference on Knowledge Discovery and Data Mining. Volume 5012 of Lecture Notes in Artificial Intelligence., Springer-Verlag (2008) 904–909

18. Wang, J.Z., Li, J., Wiederhold, G.: SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. IEEE Trans. on Patt. Anal. and Machine Intell. **23** (2001) 947–963
19. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2nd edn. Morgan Kaufmann, San Francisco (2005)
20. Lewis, D.D., Williams, K.: Reuters-21578 corpus ApteMod version