Mining Emerging Patterns by Streaming Feature Selection

Kui Yu[1, 2], Wei Ding[2], Dan A. Simovici[2], and Xindong Wu ✉ [1,3]
[1]Department of Computer Science, Hefei University of Technology, Hefei, 230009, China
[2]Department of Computer Science, University of Massachusetts Boston, Boston, 02125, USA
[3]Department of Computer Science, University of Vermont, Burlington, 05405, USA
{ykui,ding,dsim}@cs.umb.edu, xwu@uvm.edu (✉ corresponding author)

ABSTRACT
Building an accurate emerging pattern classifier with a high-dimensional dataset is a challenging issue. The problem becomes even more difficult if the whole feature space is unavailable before learning starts. This paper presents a new technique on mining emerging patterns using streaming feature selection. We model high feature dimensions with streaming features, that is, features arrive and are processed one at a time. As features flow in one by one, we online evaluate each coming feature to determine whether it is useful for mining predictive emerging patterns (EPs) by exploiting the relationship between feature relevance and EP discriminability (the predictive ability of an EP). We employ this relationship to guide an online EP mining process. This new approach can mine EPs from a high-dimensional dataset, even when its entire feature set is unavailable before learning. The experiments on a broad range of datasets validate the effectiveness of the proposed approach against other well-established methods, in terms of predictive accuracy, pattern numbers and running time.

Categories and Subject Descriptors
I.5.2 [**Computing Methodologies**]: Design Methodology-classifier design and evaluation, feature evaluation and selection

General Terms
Algorithms, Experimentation

Keywords
Emerging patterns, Feature relevance, Streaming features

1. INTRODUCTION
An emerging pattern (EP for short) is a pattern whose support value changes significantly from one class to another [9]. Highly accurate classifiers can be built by aggregating the differentiating power of EPs [7, 10].

Mining EPs is still a daunting problem when the number of feature dimensions can be in the thousands, as it is difficult to store, retrieve, prune, and sort them efficiently for classification with a huge number of candidate EPs. With the advent of emerging massive datasets involved with hundreds of thousands of features, such as in image processing, gene expression data, text data, and so on, this pattern search space is rather huge and even smoothing the full feature space sometimes becomes very costly or simply impossible. Therefore, mining EPs from such a space has to face two challenging research issues: (1) how to efficiently mine a small set of strongly predictive EPs from a high-dimensional dataset; and (2) how to mine strongly predictive EPs from a large feature space as exhaustive search over it is either very time-consuming or simply infeasible.

In this paper, we propose a new approach to battle these two challenging issues. A novel contribution of our approach is that it uses streaming features to model a high-dimensional feature space, and then integrates streaming feature selection into the EP mining process to help efficient and effective discovery of a small set of strongly predictive EPs in a large feature space yet to get promising performances.

The concept of streaming features has been proposed to handle feature selection in a changing feature space over time [19, 24]. Unlike data streams, with streaming features, feature dimensions are modeled as a feature stream, and features flow in one by one and each feature is processed upon its arrival. Recent research has shown that streaming feature selection is effective and efficient with not only a huge feature space but also an unknown full feature space before learning [19]. However, if we consider streaming feature selection and EP mining as a whole, aggregating all features and samples to mine EPs is a hard research problem:

(1) Online data processing. Since feature dimensions flow in one by one, it is required to online transform, map and partition arriving features. Firstly, converting a real-world dataset into a desired encoded dataset of all items is infeasible before mining starts. Secondly, the mapping between the item numbers and real-world features needs to be constructed and updated as features flow in one by one over time. Thirdly, as a feature is available, we must online divide the data of each class accordingly instead of dividing all data with all features in advance.

(2) Dynamical EP mining. With streaming features, one solution to mine EPs is to employ streaming feature selection to dynamically control the EP mining process. The problem is how to integrate streaming feature selection into this EP mining process to get an accurate EP classifier.

In this paper, we propose *EPSF* (mining ***E***merging ***P***atterns by ***S***treaming ***F***eature selection). More specifically, *EPSF* assumes that features arrive one at a time, and each feature is online processed upon its arrival. With the online processing, a two-level framework is proposed to handle dynamical EP mining. In the first level, by exploiting the relations between feature relevance and EP discriminability (the predictive ability of an EP), *EPSF* online builds an influential feature pool by evaluating each arriving feature whether it is useful for mining strongly predictive EPs. In the second level, *EPSF* online builds a candidate EP pool by using this feature pool to online guide EP mining. As features flow in one by one, by interleaving the two levels, *EPSF* provides a natural way to integrate streaming feature selection and EP mining for the high feature dimension challenge in EP mining.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 gives the preliminary and Section 4 presents our approach. Section 5 reports our experimental results. Finally, Section 6 provides our conclusion and future work.

KDD'12, August 12–16, 2012, Beijing, China.

2. RELATED WORK

Dong and Li [9] introduced EPs to represent strong contrasts between different classes of data. In addition, a Jump Emerging Pattern (JEP for short) is a special type of EPs whose support increases from zero in one class to non-zero in the other class [12]. Like other patterns composed of conjunctive combinations of elements, EPs can be easily understood and used directly in a wide range of applications, such as failure detection [16] and discovering knowledge in gene expression data [8, 18].

One of the biggest challenges of mining EPs is the high computational cost due to an exponential number of candidate patterns [17]. A number of interestingness measures are defined to reduce the number of discovered EPs without sacrificing their discriminative power. Dong and Li [9] first proposed a border-based approach inspired by the Max-Miner algorithm [2]. In their approach, borders are used to represent candidates and subsets of EPs and the border differentiation operation is used to discover EPs. The ConsEPMiner algorithm follows a level-wise, candidate generation-and-test approach to mine EPs [23]. Bailey et al. [3] proposed a fast algorithm for mining JEPs, which is much faster than the border-based approach. Later Bailey et al. [4] presented a new algorithm for efficiently mining EPs by computing minimal hypergraph transversals. Inspired by the FP-tree, a CP-Tree miner based on the CP-tree data structure was presented to improve EP mining performance [6]. Despite the significant improvement on EP mining, [17] demonstrated that those previous techniques were unable to handle more than sixty dimensions. They proposed a ZBDD EP-miner using Zero-Suppressed Binary Decision Diagrams (ZBDDs) for mining EPs from high-dimensional data.

Much research on EPs has largely focused on classification. Dong et al. [10] proposed the first EP classifier, called CAEP (***C***lassification by ***A***ggregating ***E***merging ***P***atterns). Based on CAEP, Li et al. proposed a JEP-classifier which is distinct from the CAEP classifier [12].The JEP-classifier uses JEPs exclusively because JEPs discriminate between different classes more strongly than any other type of EPs. Both of these classifiers mine EPs by a border-based approach. Meanwhile, Li et al. [13] also presented a lazy EP classifier based on an instance-based EP discovery, called DeEPs, to improve the efficiency and accuracy of CAEP and JEP-classifier. In addition, Fan and Ramamohanarao [7] proposed a robust EP-classifier, called SJEP-classifier, using a strong jumping emerging pattern that is a special JEP whose support is zero in one class but non-zero and satisfying a minimal support threshold in the other class. The SJEP-classifier integrates the CP-tree miner into the EP classifier, and uses many fewer JEPs than the JEP-classifier.

Limited to the EP mining techniques, existing EP classifiers still cannot handle a dataset with more than sixty dimensions. Although the ZBDD EP-miner can deal with a high-dimensional dataset, like previous approaches, it still suffers from an explosive number of EPs, even with a rather high support threshold. It is still a challenging research issue to mine a small but strongly predictive EPs from a huge number of candidate EPs. In a recent study, Yu et al. addressed the concept of causal associative classification using Bayesian networks to help construct EP classifiers [21]. The study has shown that integrating causal-structure learning into EP mining can efficiently extract a minimal set of strongly predictive patterns from high-dimensional data and get highly accurate EP classifiers. In comparison to [21], we propose a new approach on mining emerging patterns by streaming feature selection. The method is capable of dealing with an unknown or very large full feature space, while exhaustive search is either very time-consuming or simply infeasible.

3. PRELIMINARY KNOWLEDGE

Assume we have a dataset D defined upon a set of N features $F=(F_1, F_2, \dots, F_N)$ and the class attribute C. For every feature F_i, $i = 1, \dots, N$, we assume it is in a discrete domain $\mathrm{dom}(F_i)$. Let I be the set of all items, $I = \bigcup_{i=1}^{N} \mathrm{dom}(F_i)$. An itemset X is a subset of I and its $\mathrm{support_D} = \mathrm{count_D}(X)/|D|$, where $\mathrm{count_D}(X)$ is the number of instances in D containing X and $|D|$ is the number of instances in D. Let $C = \{C_1, C_2, \dots, C_K\}$ be a finite set of K distinct class labels. The dataset D can be partitioned into $D_1, D_2, \dots, D_K$,where D_j consists of instances with class label C_j, $j = 1, \dots, K$. The ***G***rowth ***R***ate (GR for short) of X from D_l to D_m, $l, m = 1, \dots, K$ and $l \neq m$, is defined as follows.

Definition 1. (GR: Growth Rate) [9] $GR_{D_l \to D_m}(X) = \frac{\mathrm{support}_{D_m}(X)}{\mathrm{support}_{D_l}(X)}$. If $\mathrm{support}_{D_m}(X) = 0$ and $\mathrm{support}_{D_l}(X) = 0$, then $GR_{D_l \to D_m}(X) = 0$; if $\mathrm{support}_{D_m}(X) \neq 0$ but $\mathrm{support}_{D_l}(X) = 0$, then $GR_{D_l \to D_m}(X) = \infty$.

Definition 2. (EP: Emerging Pattern) [9] Given a threshold $\rho > 1$, an EP from D_l to D_m is an itemset X where $\mathrm{GR}_{D_l \to D_m}(X) \geq \rho$.

An EP e from D_l to D_m is also called an EP of D_m. If $\mathrm{GR}(e) = \infty$, e is called a Jumping EP (JEP). The goal of EP mining is to extract the EP set E_i for each class C_i which consists of EPs from $D - D_i$ to D_i, given a minimum growth rate threshold and a minimum support threshold.

Definition 3. (**Growth Rate Improvement**) [23] Given an EP e, the growth rate improvement of e, $\mathrm{Rateimp}(e)$, is defined as the minimum difference between its growth rate and the growth rates of all of its subsets,

$$\mathrm{Rateimp}(e) = \min\big(\forall e' \subset e, \mathrm{GR}(e) - \mathrm{GR}(e')\big).$$

A positive growth rate improvement threshold ensures a concise and representative set of EPs which are not subsumed by one another and consist of items that are strong contributors to their predictive power. The growth rate improvement can also help to reduce the search space by eliminating EPs that are uninteresting or redundant.

An illustrating example is given in Tables 1 and 2 using the Balloon dataset from the UCI machine learning repository [5]. Assuming the minimum support threshold is 0.2 and the growth rate threshold $\rho > 1$, the candidate EPs are of two classes T (the inflated class) and F (the not inflated class) with 20 samples and 4 features: *color*, *size*, *act* and *age*.

Table 1 The candidate EPs from class T to class F

Candidate EP	Support (class T)	Support (class F)	$GR_{T \to F}(e)$
{act=dip}	0.33	1	3
{age=child}	0.33	1	3
{act=dip, age=child}	0	1	∞

Table 2 The candidate EPs from class F to class T

Candidate EP	Support (class F)	Support (class T)	$GR_{F \to T}(e)$
{act=stretch}	0	0.67	∞
{age=adult}	0	0.7	∞

From Definition 2, in Table 2, both {act=stretch} and {age=adult} are EPs of class T. In Table 1, by Definition 3, we can see that {act=dip, age=child} is the EP of class F since both {act=dip} and {age=child} are subsumed by it.

When applying EPs to classification, we get all the EPs of each class C_i in a training set that are subsets of a test instance t to contribute to the final decision as to whether t should be labeled as C_i. More specifically, we derive k scores for t, one score per class, by feeding the EPs of each class into a scoring function. Then $\text{label}(t) = \text{argmax}_{C_i \in C}\, \text{score}(t, C_i)$. The following definition provides the scoring function of the EP-Classifier [10].

Definition 4 (Aggregate Score). Given an instance t and a set E_i of EPs of class $C_i \in C$ mined from the training data, the aggregate score of t for C_i is defined as

$$\text{score}(t, C_i) = \sum_{e \subseteq t, e \in E_i} \frac{GR_{D-D_{C_i} \to D_{C_i}}(e)}{GR_{D-D_{C_i} \to D_{C_i}}(e)+1} * \text{support}_{C_i}(e)$$

A potential problem in Definition 4 is that the number of EPs from different classes is likely unbalanced. If a class C_i contains more EPs than another class C_j, a test instance tends to obtain a higher score for C_i than for C_j, even when the test actually belongs to C_j. To solve this problem, Dong et al. [10] presented the concept of a base score for each class C_i named $\text{baseScore}(C_i)$, which is first calculated from the training instances of the class. With the base score, the new score of an instance t for C_i, named $\text{normScore}(t, C_i)$, is defined as the ratio of the score calculated by Definition 4 and the base score of C_i,

$$\text{normScore}(t, C_i) = \frac{\text{score}(t, C_i)}{\text{baseScore}(C_i)}.$$

The class with the highest normScore wins, and ties are broken by choosing the largest class as the winner.

4. EMERGING PATTERN MINING BY STREAMING FEATURE SELECTION

4.1 Feature Relevance and EP Discriminability

It is infeasible to examine a search space covering all possible item combinations for a large high-dimensional dataset. The question is whether some feature values could be pruned before mining EPs. As illustrated in Tables 1 to 2, the final set of EPs does not contain features *size* and *color*, since they have no impact on the predictive ability of their corresponding EPs. We propose to integrate streaming feature selection to significantly improve EP mining in high-dimensional data to produce an accurate EP classifier, which otherwise would be impossible due to the large search space. In this section, we first analyze on the relationships between feature relevance and EP discriminability, and then evaluate the degree of features relevance with the discriminative power of EPs as the features flow over time.

An input feature can be in one of three disjoint categories, namely, strongly relevant, weakly relevant, and irrelevant, with respect to its relevance to the class attribute [11]. Weakly relevant features can be further divided into redundant features and non-redundant features, and then an optimal feature subset should contain weakly relevant but non-redundant features and strongly relevant features [22]. In the following definitions, let F be a full set of features, F_i denote the i^{th} input feature, C be the class attribute and P(C|S) is the probability distribution of different classes given a feature subset $S \subseteq F$.

Definition 5 (Strong Relevance) A feature F_i is strongly relevant to C iff $\forall S \subseteq F - \{F_i\}\ \text{s.t.}\ P(C|S) \neq P(C|S, F_i)$.

Definition 6 (Weak Relevance) A feature F_i is weakly relevant to C iff it is not strongly relevant, and

$$\exists S \subset F - \{F_i\}\ \text{s.t.}\ P(C|S) \neq P(C|S, F_i).$$

Definition 7 (Irrelevance) A feature F_i is irrelevant to C iff it is neither strongly nor weakly relevant, and

$$\forall S \subseteq F - \{F_i\}\ \text{s.t.}\ P(C|S, F_i) = P(C|S).$$

As defined in Definitions 1 and 2, the discriminability of an EP is determined by its support value and growth rate. Proposition 1 below establishes the relations between non-EPs and irrelevant features.

Proposition 1 For $\forall F_i \in F$, $\forall f \in \text{dom}(F_i)$, and $\forall C_i \in dom(C)$, $GR_{D-D_{C_i} \to D_{C_i}}(F_i = f) = 1$ holds iff F_i is irrelevant to C.

Proof: Assume a dataset D has two classes $C = \{C_p, C_n\}$, D_p represents C_p class data, D_n represents C_n class data, $\sup_{D_p}(F_i = f)$ is the support value of the itemset $\{F_i = f\}$ in D_p, and $\sup_{D_n}(F_i = f)$ is its support value in D_n. Then $GR(F_i = f)$ from D_n to D_p is calculated as follows.

$$\begin{aligned} GR_{D_n \to D_p}(F_i = f) &= \sup_{D_p}(F_i = f) / \sup_{D_n}(F_i = f) \\ &= P(F_i = f | C = C_p) / P(F_i = f | C = C_n) \\ &= \frac{P(F_i = f, C = C_p)}{P(C = C_p)} \Big/ \frac{P(F_i = f, C = C_n)}{P(C = C_n)} \\ &= \frac{P(C = C_p | F_i = f) P(F_i = f)}{P(C = C_p)} \Big/ \frac{P(C = C_n | F_i = f) P(F_i = f)}{P(C = C_n)} \\ &= \frac{P(C = C_n)}{P(C = C_p)} * \frac{P(C = C_p | F_i = f)}{P(C = C_n | F_i = f)} \end{aligned}$$

On the one hand, if the term $GR_{D_n \to D_p}(F_i = f) = 1$ holds, then

$$\frac{P(C = C_p)}{P(C = C_n)} = \frac{P(C = C_p | F_i = f)}{P(C = C_n | F_i = f)}$$

As $P(C = C_p) + P(C = C_n) = 1$ and $P(C = C_p | F_i = f) + P(C = C_n | F_i = f) = 1$, we get $P(C = C_p | F_i = f) = P(C = C_p)$ (with $\frac{a}{b} = \frac{c}{d}$ equivalent to $\frac{a}{b+a} = \frac{c}{d+c}$), as well as $P(C = C_n | F_i = f) = P(C = C_n)$.

According to Definition 7, for any assignments $f \in \text{dom}(F_i)$ and $c \in \text{dom}(C)$ to F and C, $P(C = c | F_i = f) = P(C = c)$ holds, therefore F_i is irrelevant to C. Clearly, from D_p to D_n, if $GR(F_i = f) = 1$, we can also prove that F_i is irrelevant to C. On the other hand, if F_i is irrelevant to C, we get

$$\begin{aligned} GR_{D_n \to D_p}(F_i = f) &= \frac{P(C = C_n)}{P(C = C_p)} * \frac{P(C = C_p | F_i = f)}{P(C = C_n | F_i = f)} \\ &= \frac{1 - P(C = C_p)}{P(C = C_p)} * \frac{P(C = C_p | F_i = f)}{1 - P(C = C_p | F_i = f)} \\ &= 1 \end{aligned}$$

Thus, Proposition 1 is proven. □

Definition 8 (Markov Blanket) [11] Let $M_i \subset F - \{F_i\}$ be a subset of features. If F_i is conditionally independent of $F - M_i - \{F_i\}$ given M_i, then M_i is a Markov blanket for F_i.

Definition 9 (Redundant Feature) [22] A feature is redundant hence should be removed from F, if and only if it is weakly relevant and has a Markov blanket M within F.

From Definition 3, as for an EP e, if we can find an $e' \subset e$ to make $\text{rateimp}(e) \leq 0$, then e is an uninteresting or redundant EP, and might be replaced by a subset. Thus, avoiding generation of

those redundant EPs in advance will improve search efficiency. Proposition 2 below explains the relationship between feature redundancy and EP redundancy.

Proposition 2 For $\exists F_i \in F,\ \exists S \subset F,\ \forall f \in dom(F_i), \forall s \subset \bigcup_{i=1}^{|S|} dom(S_i)$, and $\forall C_i \in dom(C)$, $GR_{D-D_{C_i}\to D_{C_i}}(F_i = f, S = s) = GR_{D-D_{C_i}\to D_{C_i}}(S = s)$ holds iff F_i is redundant to C conditioning on the subset S.

Proof: $GR(F_i = f, S = s)$ from D_n to D_p is calculated as follows.

$$GR_{D_n\to D_p}(F_i = f, S = s) = \sup_{D_p}(F_i = f, S = s) \Big/ \sup_{D_n}(F_i = f, S = s)$$
$$= P(F_i = f, S = s | C = C_p) \Big/ P(F_i = f, S = s | C = C_n)$$
$$= \frac{P(F_i = f, S = s, C = C_p)}{P(C = C_p)} \Bigg/ \frac{P(F_i = f, S = s, C = C_n)}{P(C = C_n)}$$
$$= \frac{P(C = C_p | F_i = f, S = s) P(F_i = f, S = s)}{P(C = C_p)} \Bigg/ \frac{P(C = C_n | F_i = f, S = s) P(F_i = f, S = s)}{P(C = C_n)}$$
$$= \frac{P(C = C_p | F_i = f, S = s)}{P(C = C_n | F_i = f, S = s)} * \frac{P(C = C_n)}{P(C = C_p)}$$

From D_n to D_p, $GR(S = s) = P(S = s | C = C_p) / P(S = s | C = C_n)$

$$\frac{P(C = C_p | F_i = f, S = s)}{P(C = C_n | F_i = f, S = s)} * \frac{P(C = C_n)}{P(C = C_p)} = \frac{P(S = s | C = C_p)}{P(S = s | C = C_n)}$$
$$\frac{P(C = C_p | F_i = f, S = s)}{P(C = C_n | F_i = f, S = s)} = \frac{P(S = s | C = C_p) P(C = C_p)}{P(S = s | C = C_n) P(C = C_n)}$$
$$\frac{P(C = C_p | F_i = f, S = s)}{P(C = C_n | F_i = f, S = s)} = \frac{P(C = C_p | S = s)}{P(C = C_n | S = s)}$$

Using the same reasoning in proving Proposition 1, we can get two equations: $P(C = C_p | F_i = f, S = s) = P(C = C_p | S = s)$ and $P(C = C_n | F_i = f, S = s) = P(C = C_n | S = s)$. By Definitions 8 and 9, we can find a subset $S \subset F$ as a Markov blanket of F_i, and for any assignments $f \in dom(F_i)$, $s \in dom(S)$ and $c \in dom(C)$ to F_i, S and C, $P(C = c | F_i = f, S = s) = P(C = c | S = s)$ holds, and so F_i is redundant to C given S.

On the other hand, if F_i is redundant to C, from D_n to D_p, then

$$GR_{D_n\to D_p}(F_i = f, S = s) = \frac{P(C = C_p | F_i = f, S = s)}{P(C = C_n | F_i = f, S = s)} * \frac{P(C = C_n)}{P(C = C_p)}$$
$$= \frac{1 - P(C = C_n | F_i = f, S = s)}{P(C = C_n | F_i = f, S = s)} * \frac{P(C = C_n)}{P(C = C_p)}$$
$$= \frac{1 - P(C = C_n | S = s)}{P(C = C_n | S = s)} * \frac{P(C = C_n)}{P(C = C_p)}$$
$$= \frac{P(C = C_p | S = s)}{P(C = C_n | S = s)} * \frac{P(C = C_n)}{P(C = C_p)}$$
$$= \frac{P(S = s | C = C_p)}{P(S = s | C = C_n)} = GR_{D_n\to D_p}(S = s)$$

Thus, we have proven Proposition 2. □

Proposition 2 shows that if F_i is redundant to C given a subset S, then an itemset $\forall f \in dom(F_i)$ together with itemsets $\forall s \in \bigcup_{i=1}^{|S|} dom(S_i)$ contains the same predictive information as the itemsets $\forall s \in \bigcup_{i=1}^{|S|} dom(S_i)$.

According to Propositions 1 and 2, we can extract EPs without considering irrelevant and redundant features. We thus integrate streaming feature selection into the EP mining process to avoid generating non-EPs and redundant EPs.

Since the irrelevant features can be easily identified, we face two challenges here: (1) how to identify redundant features when the features stream in; and (2) how to online extract EPs from the current feature pool. We give two more propositions below to handle redundant features. From Definition 8, since the Markov blanket of C subsumes the information that all of the other features have about C, we set a Markov blanket of C (MB(C) for short) as an empty set initially, and gradually build the MB(C) as features flow in one by one over time. Clearly, by Definition 9, we can get Proposition 3 to identify whether a new arriving feature is redundant for the time being.

Proposition 3 As the features flow in one by one, a current Markov blanket of C at time t is denoted as $CMB(C)_t$. Assume an arriving feature F_i at time t+1 is weakly relevant to C. If $\exists S \subseteq CMB(C)_t$ such that $P(C|F_i, S) = P(C|S)$, then F_i can be removed.

With Proposition 3, if the new feature F_i is added into CMB(C), we can get Proposition 4 to determine which of the features in CMB(C) become redundant as F_i is added.

Proposition 4 With $CMB(C)_t$ at time t, as a new feature F_i arrives at time t+1, we suppose there does not exist any $MB(F_i)$ within $CMB(C)_t$. If $\exists Y \in CMB(C)_t$ and $\exists S \subseteq \{CMB(C)_t \cup F_i\} - \{Y\}\ s.t.\ P(C|Y, S) = P(C|S)$, then Y can be removed from $CMB(C)_t$.

4.2 Mining Emerging Patterns with Streaming Feature Selection

With the theoretical analysis in Section 4.1, we propose an algorithm *EPSF* (mining ***E***merging ***P***atterns by ***S***treaming ***F***eature selection) in Fig. 1.

1. Initialization of the minimum support threshold α, growth rate threshold ρ, $CMB(C) = \{\}$, and setting C as the class attribute
2. Input a new feature F_i
3. //Identify and remove irrelevant features
4. If $P(C|F_i) = P(C)$, discard F_i and goto step 2;
5. //Identify redundant features by Proposition 3
6. If $\exists S \subseteq CMB(C)$ s.t. $P(C|F_i, S) = P(C|S)$, go to step 2
7. //Add F_i into the current feature pool $CMB(C)$
8. $CMB(C) = CMB(C) \cup F_i$;
9. //Convert feature F_i into a set of itemsets
10. $I_{F_i} = convert(F_i)$, $I_{F_i} \in dom(F_i)$
11. //Map between I_{F_i} and F_i
12. $map\_form = mapping(F_i, I_{F_i})$
13. For i=1: |C| // |C| denotes the number of classes
14. //Mine 1-itemset EPs for each class with the thresholds α, ρ
15. $EP_i = mineEP(I_{F_i}, \alpha, \rho)$
16. //Add EP_i to the current EP pool CEP
17. $CEP = CEP \cup EP_i$
18. Endfor
19. //Update CMB(C) by Proposition 4
20. For each feature Y within CMB(C) excluding F_i
21. If $\exists S \subseteq CMB(C)$ s.t. $P(C|Y, S) = P(C|S)$
22. $CMB(C) = CMB(C) - Y$
23. //Update CEP by removing EPs generated from feature Y
24. For each $y \in I_y$
25. if $y \in CEP$, then $CEP = CEP - y$ endif
26. Endfor
27. $map\_form = map\_form - I_y$ //Update map_form
28. Endif
29. Endfor
30. Repeat steps 2 to 29 until all features have arrived
31. Find all EPs from CEP with map_form
32. Classify unlabeled instances by the mined EPs

Fig. 1. The *EPSF* algorithm

EPSF online builds two pools: a feature pool and an EP pool. The feature pool stores the influential features which are useful for mining predictive EPs and dynamically changes as the features flow in one by one, while the EP pool keeps candidate EPs which are mined from the feature pool and online updates as the feature pool changes over time. As a feature arrives, if it is a strongly relevant or non-redundant feature, *EPSF* adds it into the feature pool, online transforms it into a set of itemsets, and online mines EPs which are added into the EP pool. With the features flowing over time, the current EP pool online updates with the changing feature pool. In order to fast respond to this change, we only online mine 1-itemset EPs as features arrive one by one, and then mine all EPs with these 1-itemset EPs as all features have arrived. More specifically, *EPSF* consists of the following key stages:

- Online mining 1-itemset EPs (steps 2 to 18). As a new feature F_i arrives, *EPSF* first assesses whether it is an irrelevant one; and if so, it is discarded. Otherwise, we evaluate whether it is redundant to C by Proposition 3; and if so, it is also discarded. If not, it is added to the feature pool CMB(C). And then, *EPSF* converts feature F_i into a set of itemsets I_{F_i} and has a mapping between I_{F_i} and F_i. This mapping can guarantee that itemsets contain items mapped from the same feature, and their supersets should be pruned. With I_{F_i} and the mapping, *EPSF* divides each class data, mines EPs for each class and stores the EPs in a candidate EP pool named CEP.
- Online updating CEP and the map_form (steps 19 to 30). Due to F_i inclusion, *EPSF* updates the feature pool CMB(C) by removing redundant features. If some features are removed from CMB(C), then we update CEP and map_form.

The *EPSF* algorithm is relevant for large datasets with high feature dimensions, as it does not need to store the whole data in the memory to check whether a new feature is redundant and to update CMB(C) when a new feature is added. As a novel contribution, *EPSF* can mine EPs from a high dimensional dataset without knowing its entire feature set in advance. When the features flow in one by one, each feature is processed upon its arrival. Feature redundancy checking (step 6) and CMB(C) updating (steps 20-22) are both conducted within the current CMB(C), not in the whole feature set.

5. EXPERIMENTAL RESULTS

5.1 Experimental Setup

In order to thoroughly evaluate the proposed *EPSF* algorithm, thirty six datasets in Table 3 are selected from the UCI machine learning repository (the first 24 datasets), very high-dimensional biomedical datasets (*hiva*, *ovarian-cancer*, *lymphoma,* and *breast-cancer)*, NIPS 2003 feature selection challenge datasets (*madelon*, *arcene*, *dorothea,* and *dexter*), and four frequently studied public microarray datasets (the last 4 datasets), respectively.

Our comparative study involves three types of comparisons, using ten-fold cross-validation on all datasets.

- Comparing against state-of-the-art EP classifiers, CAEP [10] and CE-EP [21].
- Comparing *EPSF* with three well-known associative classifiers: CBA [15], CMAR [14] and CPAR [20].
- Comparing the predictive accuracy of *EPSF* with state-of-the-art non-associative classifiers, including Decision Tree J48, SVM, and AdaBoost using their Weka implementations with default parameters.

We use the method proposed by Aliferis et al. [1] to discretize continuous features for continuous datasets. In the experiments, we set the minimum confidence threshold to 0.8 for CBA and CMAR, and set the growth rate to 20 for *EPSF*, CAPE and CE-EP. To test the impact of the minimum support threshold, we set seven minimum supports for *EPSF*, CE-EP, CAEP, CBA and CMAR, including 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, and 0.4, respectively. The parameters for CPAR are set the same as those reported in [20]. CBA, CMAR, and CPAR are implemented in LUCS KDD Software Library, while *EPSF*, CAEP and CE-EP are implemented in C++. The experiments are performed on a Window 7 DELL workstation with an Intel Xeon 2.93 GHz processor and 12.0GB RAM.

Table 3 Summary of 36 Datasets.
#: number of features, SIZE: number of instances

Dataset	#	SIZE	Dataset	#	SIZE
australian	14	690	promoters	57	106
breast-w	9	3,146	spect	22	267
crx	15	690	spectf	44	267
cleve	13	303	tictactoe	9	958
diabetes	8	768	vote	16	435
german	20	1,000	wdbc	30	569
house-votes	16	230	madelon	500	2,000
hepatitis	19	155	hiva	1,617	4,229
horse-colic	22	368	ovarian-cancer	2,190	216
hypothyroid	25	3,163	lymphoma	7,399	227
heart	13	270	dexter	20,000	300
infant	86	5,337	breast-cancer	17,816	286
ionosphere	34	351	arcene	10,000	100
kr-vs-kp	36	3,196	dorothea	100,000	800
labor	16	57	colon	2,000	62
liver	6	345	leukemia	7,129	72
mushroom	22	8,124	lung-cancer	12,533	181
pima	8	768	prostate	6,033	102

5.2 Comparison of Predictive Accuracy

Tables 4 to 6 report detailed results in terms of predictive accuracy of the proposed *EPSF* classifier and the other eight classifiers, including two EP, three associative and three non-associative classifiers on the thirty six benchmark datasets. We select the best predictive accuracy under the seven minimum supports as the results for our comparative study. In Table 6, CBA, CMAR and CPAR only have results on the twenty four low dimensional datasets as they fail to deal with a high feature space. The best result is highlighted in bold face for each dataset and the symbol "/" denotes a classifier runs out of memory due to a huge number of candidate patterns.

To further investigate the classification results, we conduct paired t-tests at a 95% significance level and summarize the win/tie/loss counts of *EPSF* against the other algorithms in Table 7 (note: if a classifier fails to run on a dataset while *EPSF* works, then *EPSF* wins).

In Table 7, *EPSF* generally outperforms CAEP on all thirty six datasets and CBA, CMAR and CPAR on twenty four low dimensional datasets. *EPSF* is also superior to CE-EP which is a state-of-the-art EP classifier for handling high feature dimensions. Meanwhile, in comparison with the well-known non-associative classifiers, *EPSF* is significantly superior to J48 and AdaBoost and also very competitive with SVM as shown in Table 7. We have demonstrated in the experiments that the integration of streaming feature selection into EP mining can avoid generating non-EPs and redundant EPs. This enables *EPSF* to not only handle high-dimensional datasets such as the last twelve datasets in Table 3, but also produce very promising predictive accuracy.

Table 4 Comparison of predictive accuracy (%): *EPSF*, CAEP and CE-EP

Dataset	*EPSF*	CAEP	CE-EP	Dataset	*EPSF*	CAEP	CE-EP
australian	**87.50**	84.71	83.97	promoters	71.00	/	**72.00**
breast-w	**96.88**	**96.88**	**96.88**	spect	**72.69**	66.92	**72.69**
crx	**85.29**	84.85	82.21	spectf	**86.92**	/	83.85
cleve	82.76	**85.52**	84.83	tictactoe	72.11	**82.95**	69.58
diabetes	**72.11**	68.95	**72.11**	vote	95.48	90.00	**95.95**
german	69.10	**72.80**	71.50	wdbc	**83.39**	81.96	81.79
house-votes	**96.82**	90.91	**96.82**	madelon	**61.20**	/	59.00
hepatitis	82.67	**86.00**	85.33	hiva	**95.17**	/	93.70
horse-colic	85.28	79.72	**85.83**	ovarian-cancer	**93.81**	/	92.86
hypothyroid	**73.03**	67.78	72.82	lymphoma	76.82	/	**77.73**
heart	85.56	**85.93**	83.70	dexter	**89.67**	/	88.33
infant	91.46	/	**94.92**	arcene	80.00	/	**86.67**
ionosphere	**93.24**	90.29	92.94	breast-cancer	**92.96**	/	92.22
kr-vs-kp	**92.42**	83.49	92.23	dorothea	94.94	/	**95.06**
labor	92.00	94.00	**96.00**	colon	91.67	/	**95.00**
liver	**61.76**	57.65	**61.76**	leukemia	**100.00**	/	**100.00**
mushroom	**97.30**	96.18	96.18	lung-cancer	98.89	/	**99.44**
pima	**72.11**	68.95	**72.11**	prostate	**95.00**	/	94.00

Table 5 Comparison of predictive accuracy (%): *EPSF*, J48, SVM and AdaBoost

Dataset	*EPSF*	J48	SVM	AdaBoost	Dataset	*EPSF*	J48	SVM	AdaBoost
australian	**87.50**	85.79	85.36	86.38	promoters	71.00	63.21	**79.25**	66.04
breast-w	96.88	94.56	**97.00**	94.85	spect	**72.69**	65.54	70.04	69.66
crx	85.29	84.20	**85.51**	85.36	spectf	**86.92**	62.57	81.25	75.40
cleve	82.76	75.91	82.51	**84.82**	tictactoe	72.11	85.70	**98.33**	72.31
diabetes	72.11	72.00	**73.18**	**73.18**	vote	**95.48**	94.01	94.93	82.87
german	69.10	74.00	**76.20**	71.60	wdbc	**83.39**	75.92	79.61	76.27
heart	**85.56**	77.03	83.33	81.11	madelon	**61.20**	57.50	56.35	60.50
hepatitis	82.67	80.65	**84.52**	80.65	hiva	95.17	96.39	94.70	**96.47**
horse-colic	**85.28**	81.79	81.79	83.70	lymphoma	76.82	70.93	**77.53**	60.79
hypothyroid	73.03	**95.64**	95.57	95.23	breast-cancer	**92.96**	80.77	92.31	83.57
house-votes	96.82	96.52	**96.96**	**96.96**	ovarian-cancer	**93.81**	89.35	93.52	90.74
infant	91.46	95.39	95.41	**95.43**	dorothea	**94.94**	/	/	/
ionosphere	**93.24**	92.02	91.74	89.46	arcene	80.00	56.00	**81.00**	71.00
kr-vs-kp	92.42	**99.31**	94.99	93.84	dexter	89.67	82.67	**91.33**	81.33
labor	92.00	**92.98**	85.96	87.72	colon	**91.67**	79.03	85.48	85.48
liver	**61.76**	60.00	60.29	60.87	leukemia	**100**	90.28	98.61	**100**
mushroom	97.30	**100**	99.11	98.44	lung-cancer	98.89	90.61	**100**	96.69
pima	72.11	72.01	**73.18**	**73.18**	prostate	**95.00**	88.24	94.12	91.18

Table 6 Comparison of predictive accuracy (%): *EPSF*, CBA, CMAR and CPAR

Dataset	*EPSF*	CBA	CMAR	CPAR	Dataset	*EPSF*	CBA	CMAR	CPAR
australian	**87.50**	86.96	86.96	85.51	ionosphere	**93.24**	88.88	90.58	88.88
breast-w	**96.88**	94.09	90.82	92.95	kr-vs-kp	92.42	**93.56**	89.41	88.71
crx	85.29	**86.52**	85.51	85.51	labor	**92.00**	54.33	89.17	80.33
cleve	82.76	83.12	**85.82**	78.61	liver	**61.76**	60.90	4.12	58.14
diabetes	72.11	73.18	64.24	**73.31**	mushroom	97.30	78.67	**99.37**	98.66
german	69.10	**74.50**	71.00	65.70	pima	72.11	**73.45**	63.94	67.97
house-votes	96.82	**96.96**	**96.96**	**96.96**	promoters	**71.00**	28.13	42.50	63.00
hepatitis	82.67	49.50	**83.33**	72.34	spect	**72.69**	64.42	62.66	64.42
horse-colic	**85.28**	83.69	83.91	82.02	spectf	**86.92**	55.84	80.07	54.74
hypothyroid	73.03	**94.78**	90.00	89.56	tictactoe	72.11	**100**	99.26	71.43
heart	**85.56**	84.07	84.07	77.41	vote	**95.48**	95.40	95.40	94.01
infant	**91.46**	63.72	90.00	84.30	wdbc	83.39	**95.79**	95.61	92.91

Table 7 Win/tie/loss counts of *EPSF* vs. the other 8 classifiers (pairwise t-test at 95% significance level)

	CAEP	CE-EP	CBA	CMAR	CPAR	J48	SVM	AdaBoost
EPSF	28/3/5	13/15/8	13/3/8	13/6/5	17/3/4	24/5/7	14/9/13	20/7/9

Table 8 Comparison of running time (in seconds): *EPSF*, CAEP and CE-EP

Dataset	*EPSF*	CAEP	CE-EP	Dataset	*EPSF*	CAEP	CE-EP	Dataset	*EPSF*	CAEP	CE-EP
australian	16	50	43	ionosphere	30	146	43	madelon	23	/	32
breast-w	38	51	51	kr-vs-kp	43	481	48	hiva	163	/	36
crx	17	42	31	labor	17	42	31	ovarian-cancer	68	/	34
cleve	26	53	31	liver	10	10	10	lymphoma	44	/	32
diabetes	26	48	31	mushroom	44	100	64	dexter	387	/	38
german	28	129	31	pima	27	46	45	arcene	30	/	34
house-votes	27	54	27	promoters	16	/	30	breast-cancer	958	/	47
hepatitis	21	43	37	spect	16	87	30	dorothea	440	/	164
horse-colic	20	51	31	spectf	17	/	31	colon	18	/	32
hypothyroid	30	107	32	tictactoe	16	33	31	leukemia	22	/	50
heart	26	50	45	vote	17	47	30	lung-cancer	117	/	42
infant	41	/	50	wdbc	24	85	31	prostate	27	/	34

5.3 Comparison of Numbers of Patterns

Figures 2 to 4 compare the numbers of patterns mined by *EPSF* against CBA, CMAR, CAEP and CE-EP, since these five classifiers all focus on generating patterns with the support-confidence framework. We report the average numbers of mined patterns over all seven minimum support thresholds. Since on *wdbc*, *kr-vs-kp*, *ionosphere*, *horse-colic* and *german*, CAEP cannot run using all the support thresholds due to huge numbers of patterns, the number of patterns on those datasets is averaged over the available support thresholds.

Fig.2 only plots 21 low-dimensional datasets since CAEP cannot run on the datasets of *infant*, *promoters* and *spectf*. In Fig.3, the X-axis denotes the twenty four datasets corresponding to the first twenty four datasets in Table 3 while in Fig.4 the X-axis denotes all of thirty six datasets corresponding to Table 3. It is clear that *EPSF* selects many fewer patterns than CAEP, CBA, and CMAR on all low-dimensional datasets.

In Fig. 4, *EPSF* also selects fewer patterns than CE-EP on most of the thirty six datasets. These results illustrate that both *EPSF* and CE-EP can select a small set of strongly predictive EPs from a very high dimensional dataset. In Fig.4, numbers 25 to 36 correspond to the last twelve high-dimensional datasets in Table 3. We can see that even with very high feature dimensions, the numbers of patterns selected by CE-EP and *EPSF* do not change much in comparison with those on the twenty four low-dimensional datasets.

5.4 Comparison of Running Time

The running time (in seconds) of *EPSF*, CAEP and CE-EP contains all learning time, including importing datasets, ten-fold cross validation learning and testing. Table 8 shows the running time of *EPSF* against CAEP and CE-EP, respectively. In Table 8, we can see that *EPSF* is faster than CAEP on all the datasets. Compared to CE-EP, on the first twenty four low-dimensional datasets, *EPSF* is faster than CE-EP. But on the last twelve high dimensional datasets, *EPSF* is not faster than CE-EP on some datasets. This is because at each fold cross-validation, *EPSF* takes all features into account to mine EPs while CE-EP discovers the direct causes and direct effects of the class attribute before EP mining, and then mines EPs in this reduced feature space at each fold cross-validation instead of all features. Thus, in Figures 5 to 6, we only plot the running time of EP mining (not including the time of training and testing classifiers) in one fold with the support threshold up to 0.2.

In Fig.5, the X-axis denotes the same twenty one datasets as Fig. 2. In Fig.6, on the X-axis, numbers 1 to 3 denote the datasets of *infant*, *promoters* and *spectf* respectively and numbers 4 to 15 denote the last twelve high-dimensional datasets in Table 3. From Fig.5, we can see that *EPSF* is still faster than CAEP and CE-EP on all twenty one low-dimensional datasets.

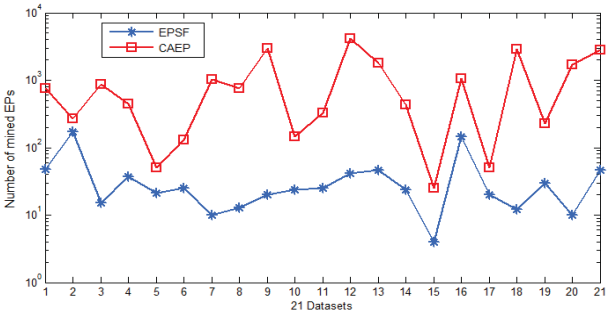

Fig.2. Numbers of mined EPs: *EPSF* vs. CAEP (the 21 datasets on the X-axis are: 1.australian, 2. breast-w, 3.crx, 4.cleve, 5.diabetes, 6.german,7. house-votes, 8.hepatitis, 9.horse-colic,10. hypothyroid, 11.heart, 12.ionosphere, 13.kr-vs-kp, 14.labor, 15. liver, 16.mushroom, 17. pima, 18.spect, 19.tictactoe, 20. vote, 21. wdbc).

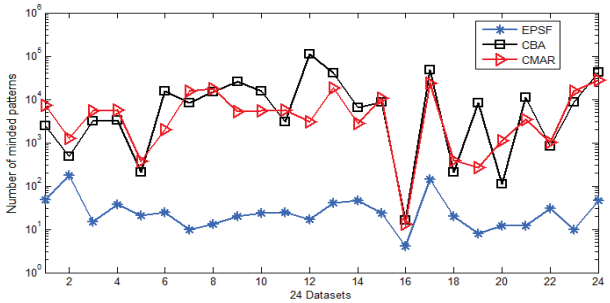

Fig.3. Numbers of mined patterns: *EPSF*, CBA and CMAR

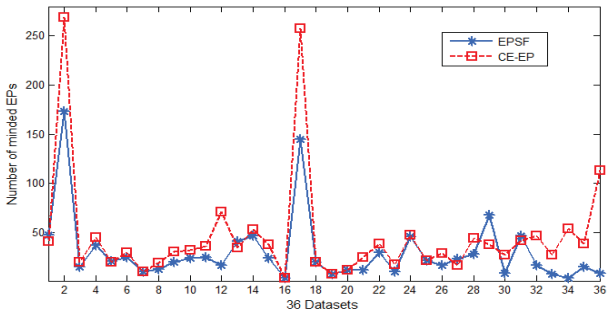

Fig.4. Numbers of mined EPs: *EPSF* against CE-EP

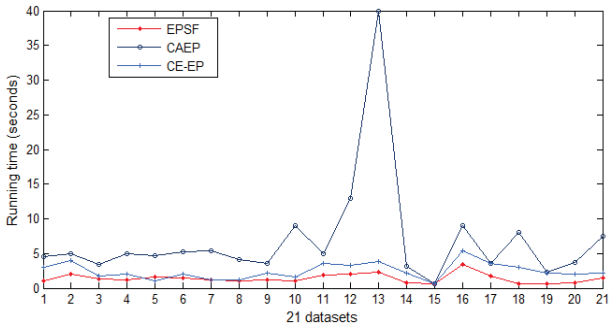

Fig.5. EP mining time: *EPSF* against CAEP and CE-EP

In Fig.6, *EPSF* is only slower than CE-EP on 3 datasets: *hiva*, *dexter* and *breast-cancer*. In summary, *EPSF* is faster than CE-EP on thirty three out of the thirty six datasets.

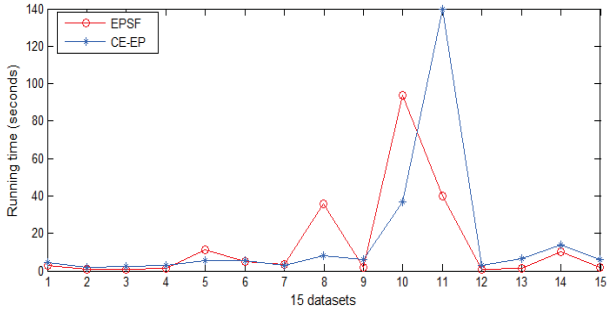

Fig.6. EP mining time: *EPSF* against CE-EP

5.5 Analysis on the Predictive Accuracy under Different Growth Rate Thresholds

To further explore the performance of *EPSF*, CE-EP, and CAEP, we conduct an analysis on the predictive accuracy of *EPSF*, CE-EP, and CAEP under seven minimum growth rate thresholds, as shown in Figures 7 to 9, where GR stands for ***G***rowth ***R***ate thresholds and the minimum support threshold is fixed at 0.1. Since on *infant*, *ionosphere*, *promoters* and *spectf*, CAEP cannot run under all seven growth rate thresholds, Fig.7 plots the predictive accuracy of the remaining 20 low-dimensional datasets under seven growth rate thresholds. In Figures 8 and 9, the X-axis denotes all of the thirty six datasets corresponding to Table 3. From Figures 7 to 9, we can see that CAEP, CE-EP and *EPSF* are not sensitive to the minimum growth rate thresholds at all, especially for CE-EP and *EPSF*.

5.6 Mining EPs without Smoothing through the Whole Feature Space

In comparison with CE-EP, *EPSF* can handle not only a large feature space, but also a high-dimensional dataset without knowing its entire feature set in advance. Sometimes, if a feature space is so large that exhaustive search over this whole feature space is either very time-consuming or simply infeasible. *EPSF* provides a solution to this problem, by processing features one by one upon its arrival and stopping this process using the EPs seen so far with a user-specified criterion. CE-EP cannot deal with this situation since it needs all features in advance to identify the causes and effects of the class attribute. We evaluate this performance of *EPSF* on only four gene datasets in Fig. 10 due to the page limit. For each dataset, we randomly select ten samples as the testing instances (five positive and five negative) and the rest for training. SVM and AdaBoost are used as baselines on the training and testing sets with all features. Without the knowledge of the whole feature space in advance, *EPSF* mines EPs on the training samples as the features flow in one by one and evaluates the current EPs on the testing samples.

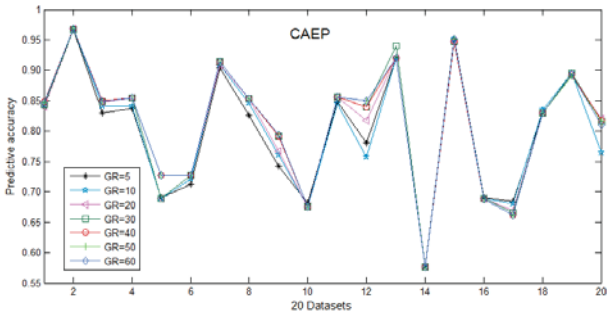

Fig.7. The impact of growth rate thresholds on CAEP (the 20 datasets on the X-axis are: 1.australian, 2. breast-w, 3.crx, 4.cleve, 5.diabetes, 6.german, 7. house-votes, 8.hepatitis, 9.horse-colic, 10. hypothyroid, 11.heart, 12.kr-vs-kp, 13.labor, 14. liver, 15.mushroom, 16. pima, 17.spect, 18.tictactoe, 19. vote, 20. wdbc).

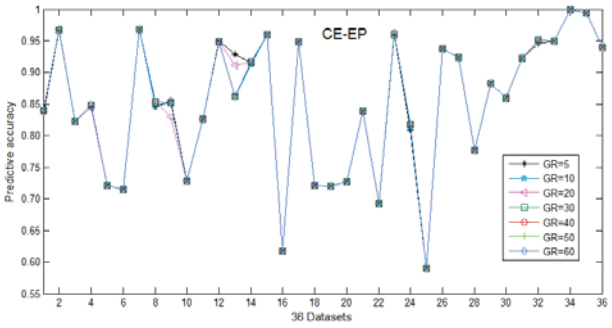

Fig.8. The impact of growth rate thresholds on CE-EP

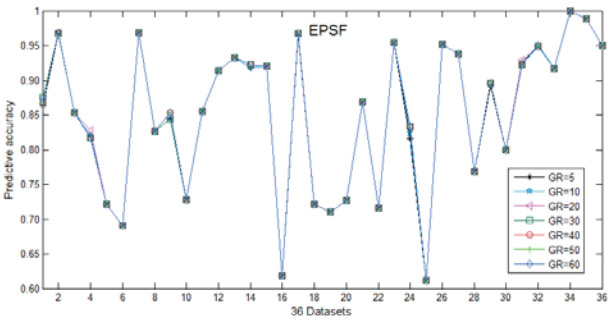

Fig.9. The impact of growth rate thresholds on *EPSF*

On the colon dataset, when the percentage of arrived features is up to 20% or 50%, the predictive accuracy of *EPSF* is the same as SVM. And when all features arrive, the accuracy of *EPSF* is up to 100%, and is better than SVM. For the remaining datasets, *EPSF* is never worse than AdaBoost and is also up to the accuracy of SVM without exhaustive search over the full feature space. This demonstrates that *EPSF* provides an effective and efficient solution to the EP mining problem when smoothing through the whole feature space is expensive or simply impossible.

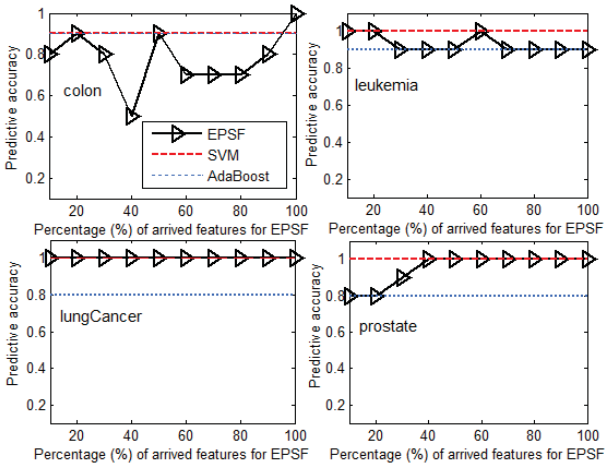

Fig.10. Comparative performance of *EPSF* without knowing the entire feature set before learning starts

5.7 A Summary of Experimental Results

Based on the comparative study in Sections 5.2 to 5.6, we have the following observations:

(1) On all datasets, *EPSF* produces significantly smaller numbers of patterns. It is more accurate than the four associative classifiers (CAEP, CBA, CMAR, and CPAR) and the two state-of-the-art non-associative classifiers (J48 and AdaBoost), and is very competitive with SVM. Moreover, as associative classifiers, CAEP, CBA, CMAR and CPAR cannot deal with high-dimensional datasets. As for the running time, *EPSF* is faster than CAEP on all datasets.

(2) *EPSF* vs. CE-EP. Both *EPSF* and CE-EP can handle very high feature dimensions yet get promising predictive accuracy. When the entire feature set is known in advance, on three evaluation metrics, accuracy, number of patterns and running time, *EPSF* is superior to CE-EP, although they are quite close. This empirically verifies the relationships between feature relevance and EP discriminability. In addition, with streaming feature selection, compared to CE-EP, *EPSF* deals well with not only a high feature dimension, but also an unknown full feature space before learning. It is possible that *EPSF* can avoid an exhaustive search over the full feature space.

6. CONCLUSION AND FUTURE WORK

In this paper, we explored the relationships between feature relevance and EP discriminability. By employing the relationships, we integrated streaming feature selection to guide a dynamic EP mining process. This new approach can handle not only a large feature space, but also a high-dimensional dataset without knowing its entire feature set in advance. Experimental results have demonstrated the effectiveness and efficiency of our approach. We plan to apply our new approach to real planetary images that can generate infinite texture-based features.

ACKNOWLEDGMENTS

This work is supported by the National 863 Program of China (2012AA011005), the National Natural Science Foundation of China (61070131, 61175051 and 61005007), the US National Science Foundation (CCF-0905337), and the US NASA Research Award (NNX09AK86G).

REFERENCES

[1] C. F. Aliferis, I. Tsamardinos, A. Statnikov & L.E. Brown. (2003) Causal Explorer: a causal probabilistic network learning toolkit for biomedical discovery. METMBS'03.

[2] Roberto J. Bayardo. (1998) Efficiently mining long patterns from databases. SIGMOD'98, 85-93.

[3] J. Bailey, T. Manoukian & K. Ramamohanarao. (2002) Fast algorithms for mining emerging patterns. PKDD'02, 39-50.

[4] J. Bailey, T. Manoukian & K. Ramamohanarao. (2003) A fast algorithm for computing hypergraph transversals and its application in mining emerging patterns. ICDM'03, 485-488.

[5] C.L. Blake & C.J. Merz. (1998) UCI Repository of Machine Learning Databases.

[6] H. Fan & K. Ramamohanarao. (2002) An efficient single-scan algorithm for mining essential jumping emerging patterns for classification. PAKDD'02, 456-462.

[7] H. Fan & K. Ramamohanarao. (2006) 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), 721-737.

[8] G. Fang, G. Pandey, W. Wang, M. Gupta, M. Steinbach, & V. Kumar. (2012) Mining low-support discriminative patterns from dense and high-dimensional data. IEEE Transactions on Knowledge and Data Engineering, 24(2), 279 - 294.

[9] G. Dong & J. Li. (1999) Efficient mining of emerging patterns: discovering trends and differences. KDD'99, 43-52.

[10] G. Dong, X. Zhang, L. Wong, & J. Li. (1999) CAEP: Classification by Aggregating Emerging Patterns. DS'99, 30-42.

[11] R. Kohavi & G. H. John. (1997) Wrappers for feature subset selection. Artificial Intelligence, 97, 273-324.

[12] J. Li, G. Dong, & K. Ramamohanarao. (2000) Making use of the most expressive jumping emerging patterns for classification. PAKDD'00, 220-232.

[13] J. Li, G. Dong & K. Ramamohanarao (2000). Instance-based classification by emerging patterns. PKDD'00, 191-200.

[14] W. Li, J. Han, & J. Pei. (2001) CMAR: accurate and efficient classification based on multiple-class association rule. ICDM'01, 369–376.

[15] B. Liu, W. Hsu, & Y. Ma. (1998) Integrating classification and association rule mining. KDD'98, 80-86.

[16] D. Lo, H. Cheng, J. Han, S. Khoo, & C. Sun. (2009) Classification of software behaviors for failure detection: a discriminative pattern mining approach. KDD'09, 557-566.

[17] E. Loekito & J. Bailey. (2006) Fast mining of high dimensional expressive contrast patterns using zero suppressed binary decision diagrams. KDD'06, 307–316.

[18] S.Mao & G.Dong. (2005) Discovery of highly differentiative gene groups from microarray gene expression data using the gene club approach. J. Bioinformatics and Computational Biology, 3(6):1263-1280.

[19] X. Wu, K.Yu, H. Wang & W. Ding. (2010) Online streaming feature selection. ICML'10, 1159-1166.

[20] X. Yin & J. Han. (2003) CPAR: classification based on predictive association rule. SDM'03, 369-376.

[21] K. Yu, X. Wu, W. Ding, H. Wang & H. Yao. (2011) Causal associative classification. ICDM'11, 914-923.

[22] L.Yu & H. Liu. (2004) Efficient feature selection via analysis of relevance and redundancy. J. of Machine Learning Research, 5, 1205-1224.

[23] X. Zhang, G. Dong & K. Ramamohanarao. (2000) Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets. KDD'00, 310-314.

[24] J. Zhou, D. Foster, R.A. Stine & L.H. Ungar. (2006) Streamwise feature selection. J. of Machine Learning Research, 7, 1861-1885.