MSJEP Classifier: "Modified Strong Jumping Emerging Patterns" for Fast Efficient Mining and for handling attributes whose values are associated with taxonomies

Mohammed K. Hassan [1*], Ahmed K. Hassan[2], Ali I. Eldesouky[2]

[1]Mechatronics Department, Faculty of Engineering, Horus University in Egypt (HUE), New Damietta, 34517, Egypt

[2]Department of Computers and Systems, Faculty of Engineering, Mansoura University, Mansoura, Egypt

Emails: mkhassan@horus.edu.eg, ahmed.hassan2017@gmail.com, : ali_eldesouky@yahoo.com

Abstract

Modified Strong Jumping Emerging Patterns (MSJEPs) are those itemsets whose support increases from zero in one data set to non-zero in the other dataset with support constraints greater than the minimum support threshold (ζ). The support constraint of MSJEP removes potentially less useful JEPs while retaining those with high discriminating power. Contrast Pattern (CP)-tree-based discovery algorithm used for SJEP mining is a main-memory-based method. When the data set is large, it is unrealistic to assume that the CP-tree can fit in the main memory. The main idea to handle this problem is to first partition the data set into a set of projected data sets and then for each projected data set, we construct and mine its corresponding CP-tree. Trees of the projected data sets are called Separated Contrast Pattern Tree "SCP-trees" and Patterns generated from it are Called MSJEPs" Modified Strong Jumping Emerging Patterns". Our proposal also investigates the weakness of emerging patterns in handling attributes whose values are associated with taxonomies and proposes using an MSJEP classifier to achieve better accuracy, better speed, and also handling attributes in taxonomy.

Keywords: Data mining, emerging patterns, classification, machine learning, mining methods, and algorithms

1. Introduction

Data mining is naming to the branch of science interested in extracting valuable knowledge from useless tons of data so it's like the extraction of gold from the sand which is called gold mining, so the right name to the data mining should be knowledge mining or knowledge discovery in database (KDD). But the first name is the most popular and we will use it interchangeably with KDD.

Data mining can be used in important business decision-making or business strategies. Its applications are wide-ranging, including medicine, banking, engineering, and so many others. Data mining aims to discover interesting or useful patterns and relationships in a large volume of data. Major tasks in data mining include concept description, association, classification, prediction, clustering, evolution analysis, and outlier analysis [1].

CLASSIFICATION is one of the major operations in data mining and it's the core of this work. Classification has also been studied on a wide range in statistics, machine learning, neural networks, and expert systems over decades [2, 3] and has been successfully applied to a wide range of application areas, such as scientific experiments, medical diagnosis, weather prediction, credit approval, customer segmentation, target marketing, and fraud detection [4, 5].

DOI: https://doi.org/10.54216/JISIoT.000201

Classification based on patterns is a relatively new trend. "A Pattern is an expression describing a subset of the data" [6]

A Strong Jumping Emerging Pattern (SJEP) is defined as an item set whose support increases suddenly from zero in one class of data to nonzero in another class, the ratio of support-increase being infinite like previous classifiers called **Emerging patterns (EPs)** and **Jumping Emerging Pattern (JEP)** [7, 8] but also must satisfy the condition of minimum support threshold in the positive dataset which makes this classifier more accurate. SJEPs represent knowledge that discriminates between the different classes of a dataset. This classifier achieves higher accuracy than other state-of-the-art classifiers such as C4.5 [9] and CBA [10]. The SJEP-Classifier suffers from some weaknesses. The classifier depends on CP-Tree to mine SJEPs and the technique is memory-based. Hence, in large datasets, it's unrealistic to build such a tree. This work tries to solve the problem by proposing an algorithm as we cut this tree into equivalent small CP-Trees and we call these trees "Separated CP-Trees" or "Equivalent CP-Trees" and the mined patterns from SCP-Trees are called **Modified Strong Jumping Emerging Patterns (MSJEPs)**.
This work also investigates the weakness of emerging patterns in handling attributes whose values are associated with taxonomies and proposes using an MSJEP classifier to achieve better accuracy, better speed, and also handling attributes in taxonomy.

In Section 2: The progression of emerging patterns is defined.

In Section 3: The new technique for generating MSJEPs is proposed.

InSection4: A new technique for generating generalized EP with better accuracy and how to handle attributes in taxonomy is presented.

In section 5: Experimental results are given to compare the accuracy of the proposed classifier "MSJEP" with the other five classifiers of "the-state-of-art" classifiers on some databases from the UCI machine repository [14]. The experimental results show that our proposed classifier is more accurate than other classifiers. Other experimental results are given to compare the number of EPs, SJEPs, Generalized EPs, and Generalized SJEPs used in the mining process and also the execution time of the last two types.

In section 6: our proposal gives the conclusion and the future work.

2. Emerging Patterns Family

Suppose that a dataset D is defined upon a set of attributes $\{B_1, B_2, \ldots, B_n\}$. For each attribute B_i, this set of permitted values is called the domain of that attribute, denoted as domain (B_i). Attributes can be either categorical or continuous.

For a continuous attribute, our proposal assumes that its value range is discretized into intervals. This work calls each (attribute, categorical-value) or (attribute, continuous-interval) pair an item. (Sex, male) and (age, [18, 60]) are two examples of items.

By aggregating all the domain categorical-values and continuous-intervals across all attributes, the proposal obtains the set of all items in D, denoted as I, where I = {domain (B_1) ∩ domain (B_2)…. domain (B_n)}. All items are mapped from I including (attribute, categorical-value) and (attribute, continuous-interval) pairs to consecutive positive integers, i.e., use 1 to represent the first item in I, 2 to the second item, and so on. By doing this, the original dataset can be treated as a transaction database [1].

A set X of items is also called an item set, which is defined as a subset of I. It is assuming that any instance S contains an itemset X, if $X \subseteq S$. The support of an itemset X in a dataset D, $supp_D(X)$, is $count_D(X)=|D|$, where $count_D(X)$ is the number of instances in D containing X[11].

2.1 Emerging Patterns (EPs)

Emerging Patterns are concerned with two classes of data. This work defines the growth rate of an item set concerning both classes as follows:

Definition 1 Given two different classes of datasets D_1 and D_2, the growth rate of an item set X from D_1 to D_2 is defined as GrowthRate(X) = GR(X) =

DOI: https://doi.org/10.54216/JISIoT.000201

$$\begin{cases} 0 & if\ \text{supp}_{D1}(X) = 0\ and\ \text{supp}_{D2}(X) = 0 \\ \infty & if\ \text{supp}_{D1}(X) = 0\ and\ \text{supp}_{D2}(X) > 0 \\ \dfrac{\text{supp}_{D2}(X)}{\text{supp}_{D1}(X)} & otherwise \end{cases}$$

EPs are those item sets with large growth rates from D_1 to D_2.

2.2 Jumping Emerging Patterns (JEPs)

Suppose D contains two different classes: D_1 and D_2.

Definition 2: ***(Li et al. [7])*****. A Jumping Emerging Pattern (JEP):**

From D_1 to D_2 is an item set X that satisfies: $\text{supp}_{D1}(X) = 0$ and $\text{supp}_{D2}(X) > 0$.

2.3 Strong Jumping Emerging Patterns (SJEPs)

Definition 3: ***(Fan et al. [11])*** Given $\zeta > 0$ as a minimum support threshold, a Strong Jumping Emerging Pattern (SJEP) from D_1 to D_2 is an item set X that satisfies the following conditions:
1. $\text{Supp}_{D1}(X) = 0$ and $\text{Supp}_{D2}(X) \geq \zeta$ and
2. Any proper subset of X does not satisfy condition 1.

A negative dataset means the absence of (X) in (D) with respect to (X). An SJEP X from D_1 to D_2 is simply called an SJEP of D_2 and $\text{supp}_{D2}(X)$ is called the support of the SJEP. In this example, D_2 is called the positive dataset, and D_1 is called the negative dataset).

Like JEPs [12], SJEP also has strong predictive power due to its infinite growth rates. Unlike JEPs, SJEP requires that the support of SJEPs in D_2 (positive dataset) be above the minimum support threshold ζ (condition 1).

This ensures that an SJEP should cover at least a certain number of instances in a training data set. Condition 2 shows that any proper subset of an SJEP is not an SJEP, which means SJEP is the shortest JEP satisfying the support constraint. Consider that JEPs are actually item sets. A shorter JEP means fewer items (attributes). If this proposal can use fewer attributes to distinguish two classes, adding more attributes will not contribute to classification and will add more noise to the classification process, so we are interested in the shortest pattern to satisfy our definition and neglect the superset.

3. The Proposed classifier MSJEPs

The proposed MSJEP classifier retains the high accuracy of the SJEP classifier but tries to handle its weak point in memory management.
The SJEP classifier constructs one CP-tree for the whole data set [11]. Since this technique is memory-based; it's unrealistic that large data sets can fit the main memory.
So this proposal proposes a new classifier technique in which the CP-tree that has been proposed in [11] will be partitioned into equivalent small CP-trees to solve the problem of memory consumption.
Figure 1 shows the steps representing the proposed classifier

DOI: https://doi.org/10.54216/JISIoT.000201

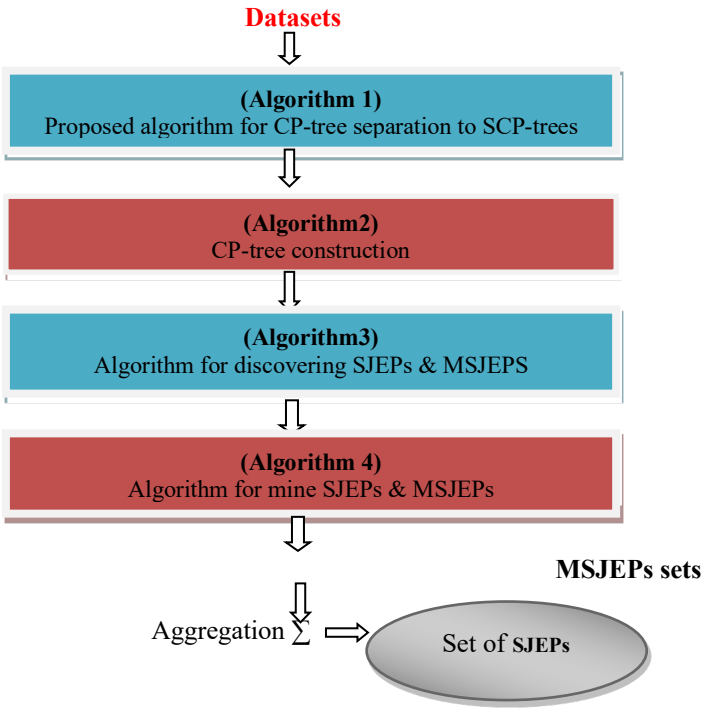

Fig (1) Hierarchy implementing mining processes for generating MSJEPs & SJEPs using 4 algorithms

3.1.1 Proposed Separated Contrast Pattern Trees (SCP-Tree):

This section proposes a technique for constructing a CP-tree in which the data set is partitioned into small data sets, each of them is represented by a small SCP-tree and this tree can be treated as the original CP-tree to generate SJEPs but here, each SCP-tree will generate a subset of SJEPs called MSJEPs holding the same characteristics.

After each SCP tree has generated a set of MSJEPs, we will free the memory from this tree and store the generated patterns. This operation will be repeated n times where n is equal to the number of SCP-trees. So this technique proposes a kind of memory management to solve the problem of the memory consumption that SJEP suffers from. The following algorithm is proposed for this purpose.

Algorithm 1 (proposed algorithm):

1- Let $D=D_1 \cap D_2$
2- Scan D once. Collect the set of items in D
(Along with their support ratios) support ratio for each item → [A].
3- [B] ← [A] sorted descending according to its support ratio.
4- If two or more items in [B] have the same minimum support ratio, we will sort them in lexicographical order.
5- If an item appears as a singleton set, we will reorder [B] putting this item preceding the others discarding lexicographical order.
6- Splitting dataset according to the first element in each pattern.

Example 1

In the data set shown in Table 1 [11].

DOI: https://doi.org/10.54216/JISIoT.000201

Table (1) An Example Data Set with Two Classes

ID	Class Label	Instances (Itemsets)	Item sets Ordered by≺
1	D_1	{a,c,d,e}	[e.a.c.d]
2	D_1	{a}	[a]
3	D_1	{b,e}	[e,b]
4	D_1	{b,c,d,e}	[e,b,c,d]
5	D_2	{a,b}	[a,b]
6	D_2	{c,e}	[e,c]
7	D_2	{a,b,c,d}	[a,b,c,d]
8	D_2	{d,e}	[e,d]

Instead of constructing a global CP-tree, this proposal performs the following. Starting from item e, the set of instances that contain e can be collected into an e-projected data set: three D_1 instances {e, a, c, d}, {e, b}, and {e, b, c, d} two D_2 instances {e, c} and {e, b}. The CP-tree for this projected data set is the same as the subtree R.e (means node of item e in the root) in Figure 2.
Similarly, the a-projected data set contains two D_1 instances ({a, c, d} and {a}) and two D2 instances ({a, b} and {a, b, c, d}).

This approach will split our dataset into small projected data sets; each will be used to build a separate CP-tree. As shown inFigure(2) this proposal presents an algorithm that separates the complete CP-tree inFigure(3) into two trees as inFigure(2). Each tree can be used separately to mine a set of SJEPs (Strong Jumping Emerging Patterns) called MSJEPs (Modified Strong Jumping Emerging Patterns) and then free the memory and use the other tree for mining another set of MSJEPs. The aggregation of these sets of MSJEPs gives the same original set of SJEPs but with a kind of memory management, also this operation can be done distributed on several machines for better performance. Thus, our approach has more scalability due to its adaptable performance with resources and time.

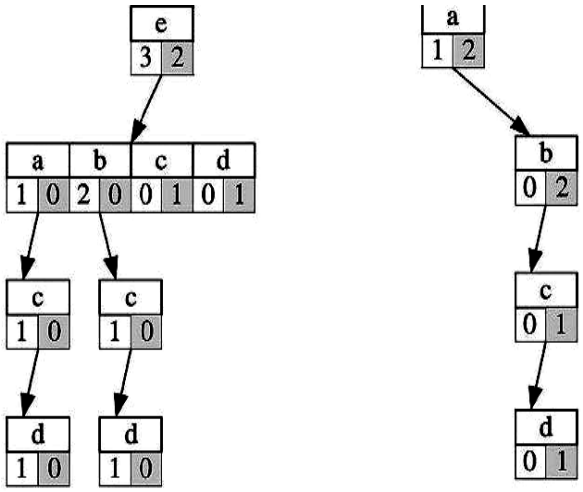

Figure (2) Separating the original CP-tree to a set of SCP-tree according to the new algorithm

3.1.2 Support Ratio of Individual Item

Assume D contains D1 and D2. Let I = {i_1, i2,…, i_n) be the set of all items appearing in D. Note that, for an item $i \in$ I, this proposal has a singleton item set {i} ⊂ I.

Definition 4: ***(Fan et al. [11])*** Given $\zeta > 0$ as a minimum support threshold, the **support ratio** of item i between D_1 and D_2, denoted as SupportRatio(i), is defined as

DOI: https://doi.org/10.54216/JISIoT.000201

$$Support\ Ratio\ (i) = \begin{cases} 0 & if\ \text{supp}_{D1}(\{i\})<\xi \wedge \text{supp}_{D2}(\{i\})<\xi \\ \infty & if\ (\text{supp}_{D1}(\{i\})=0 \wedge \text{supp}_{D2}(\{i\})\geq\xi) \vee \\ & (\text{supp}_{D1}(\{i\})\geq\xi \wedge \text{supp}_{D2}(\{i\})=0) \\ \max\left(\frac{\text{supp}_{D2}(\{i\})}{\text{supp}_{D1}(\{i\})}, \frac{\text{supp}_{D1}(\{i\})}{\text{supp}_{D2}(\{i\})}\right) & otherwise \end{cases}$$

The larger the support ratio of an item, the sharper discriminating power can this item provide.

Based on the definition of the support ratio (Definition 3), this proposal defines the preceding order on I. Let i, j ∈ I will be two items and i ≺ j,

- . If Support Ratio(i) > Support Ratio (j) or
- . If Support Ratio (i) = Support Ratio (j) and i < j (in lexicographical order).

3.2.1 The Contrast Pattern Tree (CP-tree):

The CP-tree data structure (an ordered multi-way tree structure) taken from the FP-tree [11], is used for EP mining for the first time [10]. A CP-tree registers the counts in both the positive and negative classes. An example of a CP-tree is illustrated in Figure (3).

Because every training instance is sorted by its support ratio according to the order "≺" between both classes when inserting into the CP-tree, items with a high ratio, which mostly generated as SJEP located closer to the root. The CP-tree mapping to an item set is a one-to-one mapping. Using the predefined order ≺, this proposal can produce the complete set of paths (itemsets) systematically through depth-first searches of the CP-tree. The CP-tree-based algorithm searches the CP-tree depth-first from the root and performs a powerful technique, node merge, along with the search.
The CP-tree-based algorithm can discover SJEPs of both D_1 and D_2 from the CP-tree at the same time, a "single-scan" algorithm not like the FP-growth algorithm that performs pattern mining from leaf to root and must create many conditional FP-trees during the mining process [11].

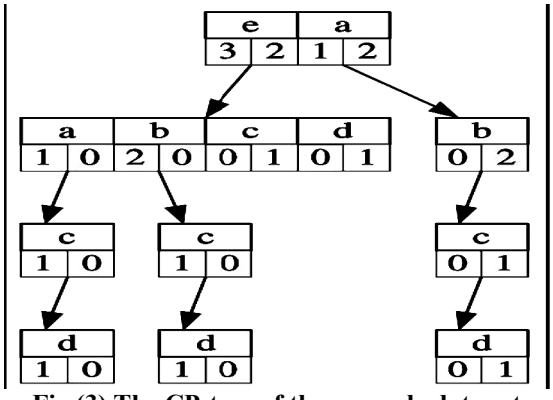

Fig (3) The CP-tree of the example data set

3.2.2 The Construction of the Contrast Pattern Tree:

In this section, our work begins to show the first step of the mining process by building CP-tree from the dataset for both classes that will be used for the mining process.

Let $D=D_1 \cap D_2$ be the training data set containing two classes and ζ be the minimum support threshold. Based on its definition, the CP-tree of input D is constructed using the following procedure: [11]

DOI: https://doi.org/10.54216/JISIoT.000201

1. Scan D once. Collect the set of items (along with their support ratios) whose support ratios are more than zero, denoted as J. Sort J in support ratio-descending order, denoted as L.
2. Create the root of the CP-tree, R, with R.itemNumber = 0. For each instance inst in D do:

Select from inst the items that are contained in L and sort them according to the order of L. Let the sorted item list in inst be [p,P], where $_p$ is the first element and P is the remaining list. Call the function insert tree ([p,P],R) (Algorithm 2).

Algorithm 2: **[11]**

insert tree ([p,P],T) (function called by CP-tree construction)
1 search T for T.items [i] = p;
2 if T.items[i] is not found **then**
3 Insert p at the appropriate place in T obeying the ≺order, denoted as T:items[i];
4 Let T.D_1counts [i] = T.D_2counts[i] = 0;
5 Increment T.itemNumber by 1;
6 switch the class label of the instance [p,P] do
7 Case class D_1 increment T.D_1counts[i] by 1;
8 Case class D_2 increment T.D_2counts[i] by 1;
end
9 if P is nonempty **then**
10 if T.items[i]'s subtree is empty **then** create a new node N with N.itemNumber =0 as T.items[i]'s subtree ;
11 Let N be T.items[i]'s subtree, call insert tree (P,N);
end

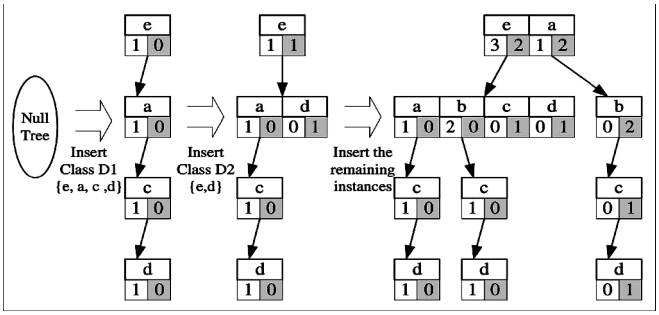

Fig (4) Constructing the CP-tree step by step: from an empty tree to the full tree.

3.3 USING THE CONTRAST PATTERN TREE TO DISCOVER SJEPS

After building SCP trees in section 3.2, the mining process is illustrated in this section. The mining process is explained step by step using the following example. Let the minimum count (in absolute occurrence) be 2. The initial CP-tree of the example data set is shown inFigure 5a, where R denotes its root and N denotes R.e's subtree.

First perform a depth-first search of the CP-tree for SJEPs, which is equivalent to the exploration of the pattern space:{e}, {e, a}, {e, a, c}, {e, a, c, d}, {e, b}, {e, b, c }, {e, b, c, d}, {e, c}, {e, d}, {a}, {a, b}, {a, b, c}, and {a, b, c, d}. However, only the counts of {e}, {e, a}, {e, a, c}, and {e, a, c, d} are correctly registered in the tree. Note that there is an "a" (1: 0) in the node R.e. So, we need to merge R.e's subtree with R itself to adjust the counts of a in R. For the same reason, in the sub data set containing "e", the counts of "b" in the node R.e may not be correct. By merging nodes along with the depth-first search, we can make sure the counts encountered are correct for determining SJEPs. Basically, the function merges all the nodes of "N" into corresponding sections of "R".

The process of merging N with R is shown inFigure 5 and the details of the merging operation are presented in algorithm 3 to adjust the count of the remaining items in the CP-tree [11].

Note that: There is an a (1:0) in the node R.e. So, we need to merge R.e's subtree with R itself to adjust the counts of a in R.

DOI: https://doi.org/10.54216/JISIoT.000201

Algorithm 3: [11]

merge_tree (T_1; T_2) (Function called in mine_tree(T, ∝))

*/*given two sub trees of a CP-tree, T_1 and T_2, the function merges T_1's nodes into T_2. T_2 is updated (including new-node generation and existing-node changes, but no nodes deletion), while T1 remains unchanged */*

1 **foreach** T_1.items[i] **do**
2 search T_2 for T_2.items[j] = T_1.items[i];
3 **if** T_2.items[j] found **then**
4 $T_2.D_1counts[j] = T_2.D_1counts[j] + T_1.D_1counts[i]$;
5 $T_2.D_2counts[j] = T_2.D_2counts[j] + T_1.D_2counts[i]$;
else
6 insert T_1.items[i] with both its D_1 and D_2 counts and child (T_1.childs[i]) at the appropriate place in T_2 obeying the order ≺ , denoted as T_2.items[j];
7 increment T_2.itemNumber by 1;
8 **if** T_1.items[i]'s subtree, M is not empty **then**
9 **if** T_2.items[j]'s subtree is empty then
10 create a new node N with N.itemNumber = 0 as T_2.items[j]'s subtree;
else N ←T_2.items[j]'s subtree;
11 call merge tree (M,N);
end
end

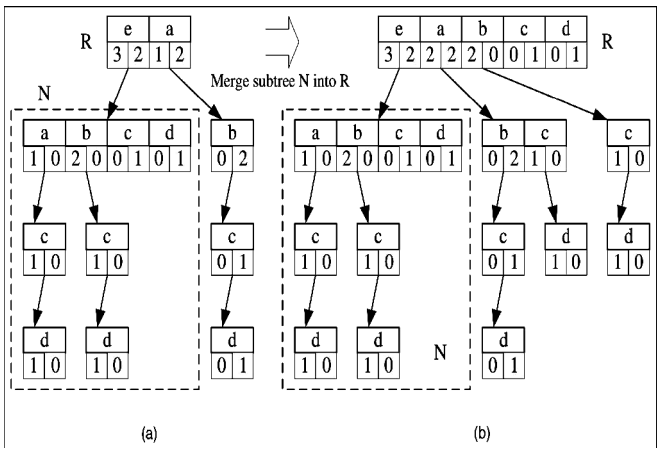

Fig (5) (a) The original CP-tree and (b) the CP-tree after merge (N, R)

3.4 Algorithms for Mining SJEPs

The mining is initiated by calling the recursive function mine-tree() (Algorithm 4) with two arguments: the root of CP-tree R and an empty item set. The item set ∝, which is initially empty, will grow one item at a time when mine-tree() is called recursively. After completing the search of the CP-tree, select only those minimal patterns by filtering out those that are supersets of others. The remaining minimal ones are SJEPs since they satisfy the minimum support threshold ζ.

Algorithm 4: [11]

mine tree (T,∝) (Function called for mining SJEPs)
/ T is a subtree of the CP-tree and ∝ is an accumulating item set */*
/ ζ is a minimum support threshold for SJEPs */*
/ $|D_i|$ = the total number of instances in $|D_i|$, where i=1,2 */*

1 **foreach** item of T, T.items[i] **do**
2 **if** T.items[i]'s subtree M is not empty **then** merge (M, T);
3 β= ∝ ∩ T.items[i] ;
4 if ($T.D_1counts[i] = 0$ ^ $(T.D_2counts[i] \geq |D_2| . \zeta)$
then generate an SJEP β of D_2 with supp(β)=$T.D_2$.counts[i];
5 **else if** $(T.D_2counts[i]=0)$ ^ $(T.D_1counts[i] \geq |D_1|.\zeta)$

DOI: https://doi.org/10.54216/JISIoT.000201

then generate an SJEP β of D_1 with supp(β)=T.D_1.counts[i]; /* Go deeper searching for longer SJEPs */
6 **else if** (T.items[i]'s subtree N is not empty) ^ (T.D_1counts[i] ≥ $|D_1|.\zeta$ / T.D_2counts[i] ≥ $|D_2|$.ζ) then call mine-tree (N,β);
/ Finish mining the subtree and free its memory */*
/ This allows mining of large data sets*/*
7 **delete** T:items[i]'s subtree;
End

4 EPs and attributes in Taxonomies (Concept hierarchies)

A concept hierarchy for a given numerical attribute defines a discretization of the attribute. Concept hierarchies can be used to reduce the data by collecting and replacing low-level concepts (such as numerical values for the attribute age or salary (**figure (6)**) with higher-level concepts (such as youth, middle-aged, or senior) or salary ranges [3].

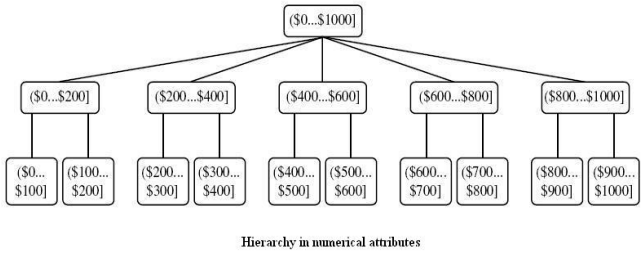

Fig (6) Examples of taxonomic hierarchies (Numeric attribute)

Categorical data are discrete data. Categorical attributes have a finite (but possibly large) number of distinct values, with no ordering among the values. Examples include geographic location and job category.

A geographic hierarchy can be defined by specifying the total ordering among these attributes at the schema level, such as street < city < province or state < country seeFigure(7).

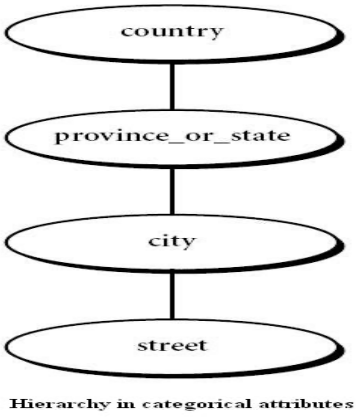

Fig (7) hierarchy in categorical attributes

4.1 Problem Description

Emerging patterns have only been capable of representing contrasts between datasets whose attributes are non-hierarchical. The patterns discovered have been at the lowest level of representation. This can lead to redundancy that discovers the same patterns, such as attribute values that are distinct, e.g., cities that are all part of the same state. This creates two problems.

First, the support of each such pattern in isolation will be less than the support of the group of related patterns.

DOI: https://doi.org/10.54216/JISIoT.000201

Second, a large group of similar patterns is more difficult to understand and verify, compared to a single, more general pattern. So our motivation has been to illustrate a previous algorithm to mine generalized emerging patterns in [13], which has the capability of dealing with data sets whose attribute values are associated with taxonomies.

But ***(Fan et al. [11])*** show that SJEPs are better than EPs; more accurate, 10 times faster, and have stronger differentiating power between classes, and more resistant to noise.

A critical challenge in this problem is that attribute hierarchies greatly expand the size of the space of potential EPs that could be mined. But our proposal shows that algorithm no 1 which is used to divide the dataset into small projected datasets minimizes the problem when this proposal takes each group of SJEPs (MSJEPs) generated from each SCP-tree and applies the same technique used for discovering the generalized emerging patterns.

4.2 Generalized EPs and generalized MSJEPs

Given a dataset D, we aim to mine the most general set of generalized EPs which contains items from any level of the taxonomies and have infinite growth rates from one class to all others. Therefore, all legal generalized EPs must have the following properties:

(1) At least one of its specializations must have non-zero support in the positive class.
(2) None of its specializations should have non-zero support in the negative class.

This is according to EPs characteristics.

This work proposes a new kind of pattern called generalized MSJEPs. It has the same two conditions above but also must satisfy the minimum support threshold in the positive data set. MSJEP is then more accurate, faster and noise resistant than EP.

4.3 Algorithm G Tree

Let us motivate the algorithm by first considering a brute-force technique used to mine generalized EPs. Our proposal permits the brute-force approach to take each MSJEP instead of EPs from the collection and enumerate all possible generalized patterns using ancestors of the items MSJEP contains (higher level in CP-Tree). For each generalized pattern, check through the negative instances to see whether it is supported in the negative class and delete it if it has non-zero support in the negative class, and check also the support constraint (ζ) in the positive class. All generalized patterns that do not have support in the negative class are considered as legal generalized MSJEPs. Before each generalized MSJEP is added to the final set, it must be checked against the MSJEPs that are already in the set to make sure that it is removed if it is a specialization of any existing patterns or the specializations of it are removed if it is considered more general than any existing patterns. The enumeration of generalized patterns is now achieved through building a set enumeration tree for each input non-hierarchical MSJEP.

4.4 G Tree Construction

G Tree is a set-enumeration tree constructed from a set of patterns resulting from the depth-first search in the CP-tree. The tree represents all possible generalized patterns that can be enumerated from the leaf-level MSJEPs.

The tree has a root, which is empty and doesn't have any value. The height of the tree is equal to the length of the MSJEP. Each path from the root to the leaf represents one possible generalized pattern.

There is an imposed ordering on the tree, such as paths from left to right and they are patterns from the most specific (ex. state) to the most general (ex. country). Before constructing the tree, this proposal first divides the set of non-hierarchical MSJEPs into some groups (G_1 to G_k) according to the combination of attributes in each pattern, such that each group should only contain patterns with items from the same combination of attributes. For each group, this proposal then constructs one tree for all MSJEPs in that group. For example, consider a dataset where $\{a_1, b_2\}$ and $\{a_1, b_3\}$ are both non hierarchical MSJEPs.

Each of these MSJEPs will result in the same generalization tree. Consequently, these patterns should be grouped, so that a single tree can be constructed for both patterns, and then pruning on the negative patterns need to occur only once to achieve the support constraint.

The Proposed algorithm for constructing a tree is outlined as follows In **figure (8)** we see an example of attributes in the hierarchy that we make G tree to implement them

DOI: https://doi.org/10.54216/JISIoT.000201

Given E = {input non-hierarchical minimal MSJEPs}
for each group of EPs $G_k \in L$ do
Tree $r \leftarrow$ buildTree (G_k)
BuildTree (group G)
Initialize empty tree with root node r
For each EP $P_j \in G$ do
InsertPattern *(r, p_j , 1)*
return tree *r;*
Let *p[l]* denote the L^{th} item in pattern p
e.g., if $p = (a_1, b_1)$, p [1] = a_1, p [2] = b_1
assign ordering of values in attribute l and their generalizations, most specific to most general

insertPattern (node n, pattern p, l)
For each item $i \in p[l] \cap generalizations\ (p[l])$
If ($i \notin n.children$) then
Insert i in *n.children* according to the order of attribute l
Find node $n_i \in n.children$ that corresponds to item i
InsertPattern (n_i, p, $++l$).

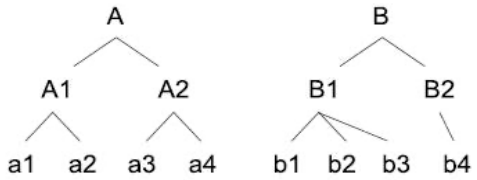

Fig (8) example to attributes in the hierarchy

4.5 The proposed Pruning from G Tree

In **figure (9)** we see an example for G Tree building based on the example in Figure 8. Note that underlined nodes correspond to nodes that are pruned in the discussion of this section.

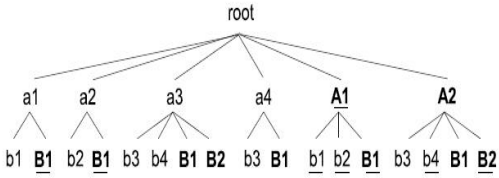

Fig (9) G Tree building based on the example in Figure 8 Note that underlined nodes correspond to nodes that are pruned

Pruning invalid SJEPs from the tree requires testing all negative instances one by one and positive instances for the support ratio

Pruning algorithm

For each pattern p from the negative dataset
PruneTree (root r, p, 1)

PruneTree (node n, pattern p)
For each item $i \in$ pattern p
If (attribute type of item $i \neq$ attribute type of *n.children*) then
skip to next item i
Else
for each node $n_c \in n_c.children$

DOI: https://doi.org/10.54216/JISIoT.000201

If (n_c.children ≠ null) then
pruneTree (n_c, p)
else delete n_c from n_c.*children*

For example; in Figure 9, once {A_2, B_1} has been extracted, {A_2, b_3} is redundant, since b3 is a specialization of B1.

Finally, the set of generalized EPs obtained from each group of input leaf-level EPs are aggregated together, in order to ensure that only the most general EPs are kept.

5 EXPERIMENTAL RESULTS

All experiments were conducted on a Dell Inspiron 6400 (Core Duo 1.86 GHz, 2Gbyte RAM) running on Windows XP Professional.

5.1 EXPERIMENT 1

Experimental results are given to compare the accuracy of the new classifier "MSJEP" with the other five classifiers of "the-state-of-art" classifiers on some databases from the UCI machine repository [14].

Table 2 describes the data sets used in the experiments in terms of the number of instances, attributes, and classes. Data sets were acquired from the UCI Machine Learning repository [14]. Each data set characteristic is explored from WEKA 3.6.1 for windows [15]. Explorer utility in WEKA gives the description and visualized charts for each attribute of the data set. Comparison between datasets characteristics is listed in the table (2)

Note: the last attribute in each dataset is the class used usually in classification.

5.1.1 Accuracy Comparison

Table 3 compares the accuracy of the MSJEP Classifier with five other classifiers. The five classifiers are Naive Bayes, C4.5, bagged C4.5, boosted C4.5, and Random Forest. All these classifiers are different examples to the state-of-the-art classifiers (Results are obtained using the WEKA implementation).

Results are reported as the mean classification performance over the 10 folds. The accuracy was obtained by using the methodology of stratified 10 fold cross-validation (CV-10).
10-fold Cross-Validation c/cs.:

- Break data into 10 sets of size n/10.
- Train on 9 datasets and test on 1.

Repeat 10 times and take a mean accuracy

Table (2) Description of Data Set

No	Dataset	# Instances	#Attributes	#Classes
1	Anneal	898	39	5
2	Audiology	226	70	24
3	Autos	205	26	6
4	Balance-Scale	625	5	3
5	Breast-cancer	286	10	2
6	Wisconsin-breast cancer	609	10	2
7	Horse-colic	368	23	2
8	Horse-colic.org	368	28	2
9	Credit rating	690	16	2
10	German credit	1000	21	2

DOI: https://doi.org/10.54216/JISIoT.000201

11	**Pima diabetes**	**768**	**9**	**2**
12	**Glass**	**214**	**10**	**6**
13	**Cleveland 14 heart diseases**	**303**	**14**	**2**
14	**Hungarian 14 heart diseases**	**294**	**14**	**2**
15	**Heart statlog**	**270**	**14**	**2**
16	**Hepatitis**	**155**	**20**	**2**
17	**Hypothyroid**	**3772**	**30**	**4**
18	**Ionosphere**	**351**	**35**	**2**
19	**Iris**	**150**	**5**	**3**
20	**Kr Vs. Kp**	**3196**	**37**	**2**
21	**Labor**	**57**	**17**	**2**
22	**Letter**	**20000**	**17**	**26**
23	**Lymphography**	**148**	**19**	**4**
24	**Mushroom**	**8124**	**23**	**2**
25	**Primary-tumor**	**339**	**18**	**21**
26	**Segment**	**2310**	**20**	**7**
27	**Sick**	**3772**	**30**	**2**
28	**Sonar**	**208**	**61**	**2**
29	**Soybean**	**683**	**36**	**19**
30	**Splice**	**3190**	**62**	**3**
31	**Vehicle**	**846**	**19**	**4**
32	**Vote**	**435**	**17**	**2**
33	**Vowel**	**990**	**14**	**11**
34	**Wave form**	**5000**	**41**	**3**
35	**Zoo**	**101**	**18**	**7**

The experimental results show that we retain the accuracy of the SJEP classifier that has been proposed in [11] but with a kind of memory management because of the proposed technique.

So, the proposed technique solves the problem of memory consumption in SJEP without affecting its superior accuracy.

.

Table (3) Accuracy comparison

Dataset	*Navie Bayes*	*C 4.5*	*Bagging*	*Boosted*	*Random Forest*	*M-SJEP*
Anneal	*95.95*	*98.57*	***98.76***	*83.63*	*99.41*	*95.00*
Anneal.Orig	*92.59*	*92.35*	*93.73*	*83.63*	***95.48***	*95.00*
Audiology	*76.79*	*77.26*	*76.00*	*46.46*	*77.08*	***79.32***
Autos	*67.26*	*81.77*	*66.55*	*44.90*	*81.80*	***89.02***
Balance-Scale	*71.56*	*77.82*	*83.37*	*71.77*	*80.11*	***87.12***
Breast-cancer	*72.59*	*74.28*	*69.10*	*71.62*	*69.70*	***96.96***
w breast cancer	*97.20*	*95.01*	*95.52*	*95.14*	*95.81*	***98.45***
Horse-colic	*80.98*	***85.16***	*84.85*	*82.53*	*85.02*	*84.17*
Horse-colic.org	*74.40*	*66.31*	*67.69*	***83.59***	*70.14*	*67.01*
Credit rating	*86.22*	*85.57*	*85.67*	*84.80*	*85.07*	***93.17***

DOI: https://doi.org/10.54216/JISIoT.000201

German credit	***74.97***	*71.25*	*74.13*	*71.27*	*73.78*	*74.00*
Pima diabetes	*75.25*	*74.49*	*75.66*	*74.92*	*74.44*	***78.95***
Glass	*71.56*	*67.63*	*72.48*	*44.89*	*76.16*	***77.40***
Cleveland 14 heart diseases	*83.34*	*76.94*	*80.16*	***83.47***	*80.31*	*82.96*
Hungarian 14 heart diseases	*84.57*	*80.22*	*79.94*	*81.41*	*79.49*	***86.12***
Heart statlog	*82.56*	*78.15*	*80.56*	*81.59*	*80.37*	***86.15***
Hepatitis	***84.18***	*79.22*	*82.00*	*81.37*	*82.47*	*83.33*
Hypothyroid	*98.54*	***99.54***	*99.55*	*92.97*	*99.19*	*98.07*
Ionosphere	*89.54*	*89.74*	*91.20*	*90.89*	*93.11*	***93.53***
Iris	*93.20*	*94.73*	*94.20*	***95.40***	*94.27*	*93.10*
Kr Vs. Kp	*87.81*	***99.44***	*99.05*	*93.84*	*98.86*	*99.22*
Labor	***90.60***	*78.60*	*83.63*	*88.37*	*86.90*	*82.00*
Letter	*64.07*	*88.03*	*90.07*	*7.09*	***92.45***	*86.81*
Lymphography	***83.13***	*75.84*	*77.37*	*75.44*	*80.74*	*79.03*
Mushroom	*95.76*	***100.00***	***100.00***	*96.29*	***100.00***	***100.00***
Primary-tumor	*49.71*	*41.39*	*43.90*	*28.91*	*42.16*	***66.02***
Segment	*80.17*	*96.79*	*96.58*	*28.52*	***97.69***	*96.04*
Sick	*92.75*	*98.72*	***98.75***	*97.12*	*98.23*	*96.83*
Sonar	*67.71*	*73.61*	*76.11*	*75.65*	*81.07*	***85.10***
Soybean	*92.94*	*91.78*	*87.00*	*27.96*	*92.16*	***93.06***
Splice	*95.41*	*94.03*	*96.07*	*86.53*	*97.04*	***99.04***
Vehicle	*44.68*	*72.28*	*73.25*	*39.81*	***74.08***	*71.36*
Vote	*90.02*	***96.57***	*95.70*	*95.43*	*95.95*	*94.76*
Vowel	*62.90*	*80.20*	*87.01*	*17.47*	***95.75***	*92.34*
Wave form	*80.01*	*75.25*	*81.84*	*66.78*	*81.86*	***83.30***
Zoo	*94.97*	*92.61*	*42.59*	*60.43*	*90.98*	***95.67***

5.1.2 Ranking

From the accuracy comparison listed above, we will compare between classifiers in terms of datasets each classifier won and we will arrange
them descending from the strongest to the weakest in the table (4). From table (4) this proposal illustrates MSJEP is more accurate than the other state-of-the-art classifiers, MSJEP wins 18 from 35, it is noted that a classifier wins more than the whole other classifiers win.

Table (4) Classifiers Ranking

Classifier	*No. of datasets*
Naive Bayes	**4**
C 4.5	**5**
Bagging	**3**
Boosted	**3**
Random Forest	**6**
MSJEPs	**18**

The second accurate classifier is Random Forest, the third C 4.5, the fourth Naive Bayes, the fifth and the sixth (weakest) are Bagging and Boosted. Results in Tables 3 and 4 are represented inFigurenos. 10 and 11 respectively

5.2 EXPERIMENT 2

Experiments are conducted to compare the performance of the G.EP algorithm and G.MSJEP on various data sets with different taxonomies structures. Comparison between G.EP and G.MSJEP in terms of the input stage, output stage, and execution time are presented in table (5).
The data sets used were acquired from the UCI Machine Learning repository. Since none of the UCI data sets have taxonomy structures defined on their attribute values, we manually added three-level hierarchies for numerical attributes by aggregating values into increasing, non-overlapping ranges [14]

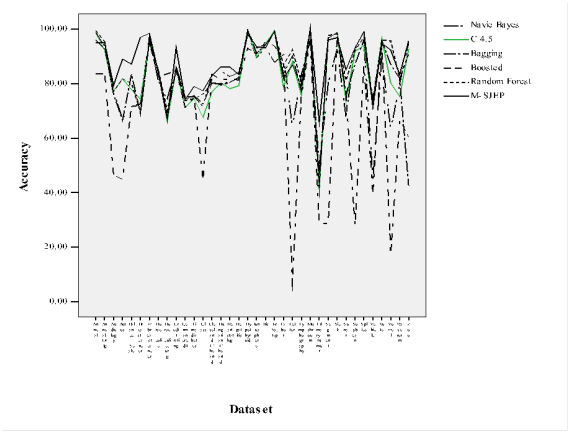

Fig (10) Accuracy comparison between classifiers against each data set

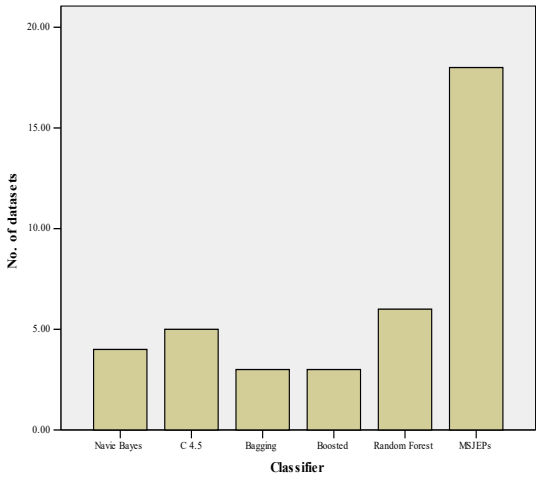

Fig (11) No of data sets each classifier won in the experiment

DOI: https://doi.org/10.54216/JISIoT.000201

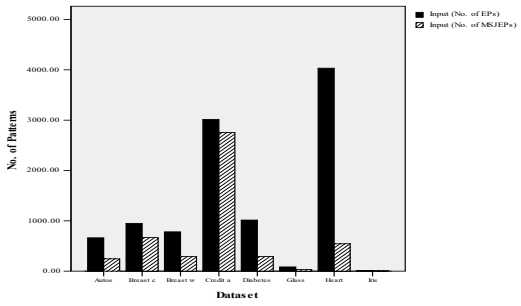

Fig (12) No. of EPS against MSJEPs in the input stage

Experimental results show that it is better to use MSJEPs instead of EPs. This will minimize the number of input patterns used to generate generalized patterns to handle attributes in taxonomyFigure(12).

Also, Experiment results show that the number of the generalized MSJEPs is less than the number of generalized EPs produced from the G tree algorithm that scales well with attributes that are in a hierarchy and also this technique presents a kind of memory management.Figure(13)

Table (5) Experimental results for the number of patterns and execution time of EPs & MSJEPs classifiers

			input			output		execution time (sec)	
Dataset	**No Att**	**No. hierarchical Attributes**	**No. EPs**	**No Groups**	**No. MSJEP**	**No. G.EP**	**No. G.MSJEP**	**G Tree For EP**	**G Tree For MSJEP**
Autos	25	15	662	658	245	675	250	2.8	1.04
Breast cancer	9	3	949	302	667	933	695	0.4	0.29
Breast w	9	9	781	243	289	860	318	2.5	0.92
Credit a	15	6	3015	1375	2754	2982	2714	4.6	4.19
Diabetes	8	8	1015	224	292	1038	299	3.9	1.12
Glass	9	9	84	63	31	96	36	0.3	0.11
Heart	13	6	4032	1792	544	4016	542	5.7	0.77
Ionosphere	34	20	126330	79060	5638	125434	55993	1512	67.42
Iris	4	4	9	4	4	6	3	0.03	0.02
Vehicle	18	18	34463	14253	4201	36902	4498	331.3	40.4

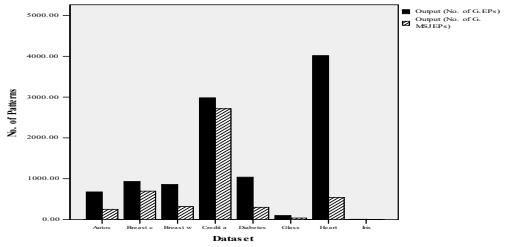

Fig (13) No. of G.EPS against G.MSJEPs in the output stage

Finally, Experiment results show that the time used to generate G.MSJEPs is less than the time used to generate G.EPs. So this technique presents a kind of time managementFigure(14).

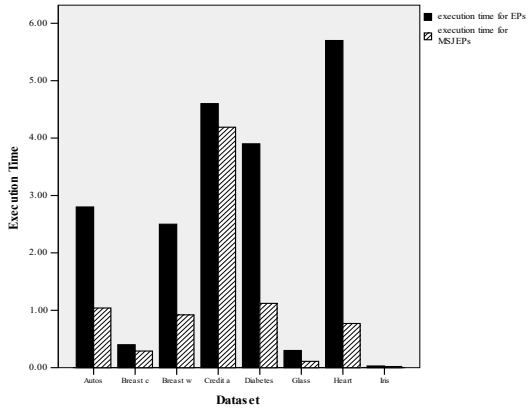

Fig (14) Execution time of G.EPS and G.MSJEPs.

6 CONCLUSION

This proposal proposes a new technique to solve the weakness point of the SJEP classifier in handling large datasets without affecting its superior accuracy. The proposed MSJEP and SJEP classifiers are 10 times faster than previous generations but MSJEP is better at dealing with large datasets by solving the problem of memory consumption in SJEP.

The algorithm which is used for SJEPs mining is a memory-based model. So it's unrealistic to handle large datasets and to build a large CP-tree for a huge dataset. This proposal proposes a technique to partition datasets into subsets, each one is implemented by a small tree and the whole set of these trees is called SCP-trees.

SCP trees can be constructed and mined consequently; each tree gives a set of MSJEPs. The aggregation process to all sets of MSJEPs gives the same set of SJEPs. We retain with its accuracy and speed and at the same time overcome the problem of memory size shortage. These trees can also be distributed over many hosts to manage time
This proposal illustrates the weakness of EPs to deal with attributes associated with hierarchies and proposes to use MSJEPs instead of EPs. This proposal names the new generation G.MSJEPs. Also, this classifier generates fewer patterns presenting a kind of memory management and takes less time in the generation process presenting also a kind of time management.

References:

[1] Kotagiri Ramamohanarao, James Bailey and Hongjian Fan: "Efficient Mining of Contrast Patterns and Their Applications to Classification". IEEE Transactions on Knowledge and Data Engineering, submitted 2007.

[2] T. Mitchell: "Machine Learning". McGraw Hill, 1997.

[3] J. Han and M. Kamber: "Data Mining: Concepts and Techniques". Morgan Kaufmann Publishers, 2000.

[4] U.M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth: "From Data Mining to Knowledge Discovery in Databases". AI Magazine, vol. 17, pp. 37-54, 1996.

[5] R. Brachman, T. Khabaza, W. Kloesgen, G. Piatetsky-Shapiro, and E. Simoudis: "Mining Business Databases" Comm. ACM, vol. 39, no. 11, pp. 42-48, 1996.

[6] Gregory Piatetsky-Shapiro and William J. Frawley: "Knowledge Discovery in Databases". AAAI/MIT Press, Cambridge, MA, 1991.

[7] J. Li, K. Ramamohanarao, and G. Dong: "The Space of Jumping Emerging Patterns and Its Incremental Maintenance Algorithms". Proc. 17th Int'l Conf. Machine Learning (ICML '00), pp. 551-558, 2000.

[8] J. Li, T. Manoukian, G. Dong, and K. Ramamohanarao: "Incremental Maintenance on the Border of the Space of Emerging Patterns". Data Mining and Knowledge Discovery, vol. 9, no. 1, pp. 89-116, 2004.

[9] J.R. Quinlan: "C4.5: Programs for Machine Learning". San Mateo, Calif.: Morgan Kaufmann, 1993.

[10] B. Liu, W. Hsu, and Y. Ma: "Integrating Classification and Association Rule Mining". Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining (KDD-98), pp. 80-86, 1998.

[11]Hongjian Fan and Kotagiri Ramamohanarao: "Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers". IEEE Transactions on Knowledge and Data Engineering, June 2006, vol.18 pp. 721-737.

[12] J. Li, G. Dong, and K. Ramamohanarao: "Making Use of the Most Expressive Jumping Emerging Patterns for Classification". Knowledge Information Systems, vol. 3, no. 2, pp. 131-145, 2001.

[13] Xiaoyuan Qian, James Bailey, and Christopher Leckie: Mining Generalized Emerging Patterns. Australian Conference on Artificial Intelligence 2006: 295-304.

[14] C.L. Blake and C.J. Merz, "UCI Repository of Machine Learning Databases," 1998, http://www.ics.uci.edu/~mlearn/MLRepository.html.

[15] WEKA, data mining tool for researches at the University of Waikato, New Zealand http://www.cs.waikato.ac.nz/ml/weka/ build 3.6.1 2009.

DOI: https://doi.org/10.54216/JISIoT.000201