Dynamic Support Range based Rare Pattern Mining over Data Streams

SunithaVanamala[1]*, L.Padma Sree[2], S. Durga Bhavani[3]
Lecturer, Department of CS, TSWRDCW, Warangal East, Warangal Telangana, India[1]
Professor, Department of ECE, VNR Vignan Jyothi Institute of Technology and Science, Hyderabad, Telangana, India[2]
Retd.Professor, School of Information Technology, JNTU, Hyderabad, Telangana, India[3]

***Abstract*—Rare itemset mining is a relatively recent topic of study in data mining. In certain application domains, such as online banking transaction analysis, sensor data analysis, and stock market analysis, rare patterns are patterns with low support and high confidence that are extremely interesting when compared to frequent patterns. Numerous applications generate large amounts of continuous data streams. We require efficient algorithms capable of processing data streams in order to analyze them and find unique patterns. The strategies developed for static databases cannot be used to data streams. As a result, we require algorithms created expressly for data stream processing in order to extract critical unique patterns. Rare pattern mining is still in its infancy, with only a few ways available. To address this is developed the Dynamic Support Range-based Hybrid-Eclat Algorithm (DSRHEA), an Eclat-based technique for mining unique patterns from a data stream using bit-set vertical mining with two item-based optimizations. The detected patterns are kept in a prefix-based rare pattern tree that uses double hashing to maintain the unusual pattern in the data stream. Testing showed that the proposed method did well in terms of how long it took to run, how many rare patterns it made and accuracy.**

***Keywords*—Depth first search; Hybrid-Eclat algorithm; SRP-tree; itemset; frequent-pattern support; rare-pattern support; pivot; data stream; rare itemset; infrequent itemset**

I. INTRODUCTION

Information systems play a vital role in identifying trends, making decisions, and adapting to the emerging changes in the market. In the current global scenario, wherein the organisations have to be sharp in its analysis of trends, there is a need for entities to be adaptable to new and dynamic solutions that can provide insights in effective ways. Utility mining and pattern detection solutions play a big role in the FMCG market [1].

Many of the earlier studies focusing on the application of business intelligence solutions in the pattern mining or utility mining space have discussed leveraging the data from information systems for analysis. However, there are very few systems among the explored models that exhibit the scope for dynamic analysis models. It is imperative that the current condition in the competitive scenario requires entities to focus on conditions wherein dynamic selection of data is targeted for informed decisions. In an illustrative scenario of a supermarket environment, the frequent mining set possibilities keep emerging over time and could lead to a paradigm shift in how each pattern or itemset billing happens periodically.

Thus, there is a need for consistent follow-up of emerging trends and periodic analysis that shall help in covering the process effectively. In this manuscript, the focus is on developing a linear and systematic approach for dynamic analysis of the pattern set designs, which shall help in accomplishing the requisite tasks. The goal of this paper is to make a dynamic model of how items are classified into different labels based on the patterns that appear in the data systems over a certain amount of time.

To speed up the process of discovering new association rules, it is crucial to identify common patterns in a dataset. Because association rules presume that every record in the database has the same occurrence frequency, they can only utilize one min-SUP per record. There are several drawbacks of mining frequent itemsets (as well as frequent patterns) with a simple min-SUP constraint in a non-uniform database. Finding patterns involving uncommon items will be impossible if min-SUP is set very high. In order to recognize patterns that include both common and unique things, we need to reduce min-SUP to a lower value. A combinatorial explosion may happen because these common objects are linked together in all possible ways. This could lead to too many patterns depending on the needs of the individual and the application.

The model proposed in this study is adapted on the basis of pivot, support, for the itemset conditions emerging in the domain. In the further sections of this report, the emphasis is on related work review, wherein key studies in the domain are explored to understand the critical success factors and constraints to consider in developing a comprehensive solution. Followed by, in the further section, the methods adapted for the model are discussed, that can help in accomplishing the proposed model. Experimental study is discussed in Section 4 and conclusion discussed in Section 5.

A. Motivation

Because rare situations represent considerable obstacles for data mining methods [2], they deserve special attention. The fundamental mining difficulties, on the other hand, have not yet been adequately investigated. Indeed, much of the literature in this area is devoted to adapting the Apriori [3]-based generic itemset mining framework to various exemptions of the frequent-itemset as well as frequent association ideas [4].Whereas these methods often cover a considerable amount of the search-space for uncommon itemsets and relationships, they are nevertheless insufficient because several rare-associations are ignored owing to excessive computational

*Corresponding Author
ISSN 2156-5570 (Online); ISSN 2158-107X (Print)

costs or unnecessarily restricted definitions. As an outcome, these strategies will be unable to collect a large amount of potentially good itemsets that are interesting in particular domains such as fraud detection, etc.

B. Problem Statement

The technique of evaluating previously unknown datasets for classifying into usable information is known as data mining. This data is collected and accumulated in common places, such as data warehouses, for efficient investigation, data-mining algorithms, helping financial decisions, as well as other information needs, effect on cost savings as well as profit improvements. Data mining is often referred to as information discovery as well as actionable insights. An itemset is a group of connected events/features in a dataset. Frequent-itemset mining is important in data mining since it uncovers latent, intriguing relationships among items in the dataset depending on the specific user-defined support and confidence levels. The Association-Rule Mining approach uncovers incredibly useful itemsets that are crucial in decision-making. The problem of frequent itemset mining, which entails mining only those itemsets that appear frequently in the input data, has dominated previous study in the topic. As a result, unusual events/features aren't captured in item sets. Those itemsets, also known as uncommon itemsets, are more intriguing than frequent itemsets in particular sectors, such as the detection of computer intrusions and fraudulent transactions in financial institutions. As a result, current study will focus on the other side of the ARM approach, which should be given equal weight in decision-making, as rare itemsets can be more valuable than regular itemsets. The same basic technique that may be used to find common itemsets can also be used to find rare itemsets. The goal of this paper is to create a collection of algorithms for mining-rare itemsets from datastream.

II. Related Work

Several frequency-related data mining techniques based on Apriori [5] have been developed, with mining outputs typically relating to common or expected events. Approach, rare-pattern mining (RPM) [6-8], was created to solve this issue, and numerous RPM methods have been suggested in the recent past. Rare-patterns, as opposed to frequent as well as common patterns, may be more beneficial in some circumstances (e.g., itemsets as well as association rules) that are vital for real-world applications. The most common types of static rare itemset mining techniques are support-threshold, no-support-threshold, and limitations are well discussed in [7]. Since the support-threshold of rare-itemsets is lesser than that of frequent itemsets, establishing lesser or different support thresholds makes it easier to generate uncommon itemsets. Darrab et al. [9] presented the multiple-support approach, which successfully discovered unusual itemsets by using various minimal support thresholds. Koh et al., [10] presented the Apriori-reverse approach to find unusual rules in totally scattered itemsets, which only includes items underneath the maximal support level. Szathmary et al. [11] later suggested a "rare-itemset mining approach (ARIMA)". The rarity method, which is a top-down rare-itemset mining automated system, was introduced by Troiano et al. [12]. Until now, a variety of RPM algorithms, such as CFP-growth++ [13], have been widely proposed. Several of them [14-16] are focused on dealing with variable data streams. Several algorithms relying on the Apriori method have a large number of candidates.

Tanbeer et al. [17] established a two-phase technique. First, a compact forest architecture CPS tree is used to keep the sliding window's contents in a predetermined sequence. All closed, frequent patterns are discovered in the second step of the tree reconstruction. An item must be constructed for each time it is used during the time period in order to use this approach, which takes further time and distance to compute.

The three processes described in Deypir et al. [18] involve the startup of the window, the updating of the window, and the production of patterns. The sequence is an interesting paradigm for tackling typical pattern mining issues since it doesn't need to examine the entire history of team and handling and could just handle a restricted range of recent events. It took a lot of memory and processing time to use sliding window techniques in the past, on the other hand.

One scan reads the databases and loads the transactions to find the most current frequent itemsets using the sliding window concept, but they were using an FP-like tree that requires more time, the model's clustering algorithm is limited, thus the different method is used to address the problem [19].

Sets of rare itemsets (SRP) were created by David Huang [20]. The new transaction's components are added to a tree data structure using the FP tree technique. Most FP trees are formed by placing all transactions' components in decreasing order of support count. As a result, it is impossible to organize data streams in a logical sequence. An architecture known as a connections table is utilized to maintain track of the sliding window's components in canonical order in order to overcome this problem. Rare items are made when a product or service path doesn't have enough support.

To save time and money, pruning is used in RP-Tree [21]. Patterns based on FP-Tree are stored in a Tree based hierarchical file system.

According to Sunitha et al [22], using bitsets as well as vertical pattern mining, different minimum item support (MIS) values are employed to overcome the problem of missing uncommon pattern sets with significant support but high confidence.

Using the approach developed by Sunitha et al. [23], researchers were able to uncover beneficial atypical association rules. Such an unusual association rule mining technique is carried out using a sliding window technique with a vertical bit sequence structure. When doing a search, a sliding window strategy is a good way to find connections that aren't obvious.

Multiple data streams and stream data are classified by an adaptive learner employing named time frames in the methodology provided by Manal Almuammar et al [24], [25].

The Éclat RP-growth approach suggested by Sunitha et al [26], [27] is developed from Zaki's Eclat [28]. Bit set mining in the form of a vertical pattern. This solution relies heavily on an effective pruning strategy and the construction of linear trees. Rahul et al. [29] suggested a compact feature extraction tree structure with better weight requirements to find common

patterns in a centralized database scanning over a continuous data stream. Outliers from incremental datasets may be identified using a single-pass rare pattern mining approach suggested by Anindita et al. [30].

III. Methods in Model

The model employs two supports, frequent-pattern Support (fps) and rare-pattern support (rps), as well as the Sliding Window size, which is the maximum number of blocks that can be present in each window. Because blocks may overlap between windows, the processed blocks data is reused in each window, and modifications are required based on newly arrived blocks and the oldest block that is about to expire upon the arrival of the new block. From steam data and a correlation measure, the model can detect both frequent and unusual Itemsets. Lift can be used to filter the rules.

A. Basic Terminology

Let $\{i_1, i_2, i_3 \ldots., i_n\}$, in be the list of Items I in the database. The data stream is a continuous stream $\{t_1, t_2, \ldots\ldots t_n, t_{n+1},\}$, with each transaction containing a subset of I items. A nonempty itemset with one or more items from items list I is called an item set. A k-itemset is an itemset that contains k items. The percentage of transactions in the database that have the itemset X marked as $|X|$ is known as support for an itemset X.

Each block is a series of transactions, and the total number of transactions in the block reflects the block size $|B|$ which is time sensitive and may change across blocks.

1) Frequent itemset: The given itemset X is considered to as frequent-pattern support if and only if $s_x > fps$ the support s_x exceeds the frequent-pattern Support fps.

2) Rare itemset: A pattern X is a rare pattern if $(rps \le s_x < fps)$, the support s_x of the corresponding pattern X is lesser thanfrequent-pattern Support (fps) but greater than or equal to rare-pattern support (rps).

3) Rare-pattern support and frequent-pattern support: The support of an itemset represents the frequency of corresponding itemset towards the set of records (buffered from data stream). The default range of support is in between 0 and 1 that includes 1, and standard measure of the support is 0.5. The default rare pattern support *rps* represents as $0 < rps \le 0.5$, the default frequent pattern support *fps* represents as $0.5 < fps \le 1$.

*4) Buffered Records' Time Frame $(brtf)$:*The amount of buffering time threshold of the recent transactions of the data stream, which should be greater than zero time units and having minimum transactions streamed from the corresponding data stream, which shall be greater than records' count threshold rct.

5) Pivot points: Maximum and minimum soft margins are the support average considered for developing the key points which can stand critical for the estimation of rare patterns and frequent patterns. Soft margins estimation is a statistical approach, which denotes the disparity between mean and deviation as well as the aggregate of mean and deviation as minimum and maximum soft margins in respective order.

B. Dynamic Support Range Assessment Process

For a given Buffered Records' Time Frame $(brtf)$, find the number of records $\{brc \exists brc \ge rct\}$ buffered as a set R.

Find all possible patterns of diversified sizes from the records listed as a set R, which represents as a set P.

Discover support of each pattern $\{p_i \exists p_i \in P \wedge 1 \le i \le |P|\}$ as follows.

Let the notation S be the set, such that each i^{th} entry represents the support s_i of i^{th} pattern $\{p_i \exists p_i \in P\}$.

$\forall_{i=1}^{|P|} \{p_i \exists p_i \in P\}$ Begin

$$s_i = \frac{1}{|R|}\left(\sum_{j=1}^{|R|}\{1 \exists (p_i \subseteq r_j) \wedge (r_j \in R)\}\right)$$

$S \leftarrow s_i$

End

1) Estimating minimum and maximum supports by soft margins: The fundamental objective of using the soft margins is to track the corresponding ranges of a metric in the learned transactions. The Soft margin values estimated from the learned transactions shall remain standard measures for the present transactions [31]. The estimation of the minimum and maximum supports of the proposed mining rare patterns from data stream is follows:

$$\mu = \frac{1}{|S|}\left(\sum_{i=1}^{|S|}\{s_i \exists s_i \in S\}\right)$$

$$\delta = \frac{1}{|S|}\sum_{i=0}^{|S|}\left\{\sqrt{(\mu - s_i)^2} \exists s_i \in S\right\}$$

$$fps = sm + \delta$$

// Finding the standard measure of the supports listed in set S

// Find the standard deviation of the supports listed in set S

// rare-pattern support (rps)

Algorithm1: Algorithmic Description of Calculating Dynamic Min and Max support ranges

//finding supports of each pattern from initial block//
For each record r of the initial block B_1, Begin

Find all possible patterns of count 2^{n-1}, here n represents the number of elements in the record $\{r \exists r \in B_1\}$

Move the discovered patterns count 2^{n-1} to the set P, if the corresponding patterns do not exists in the set P
End

For each pattern p of the set P Begin

Find the support s_p of pattern p as

$$s_p = \frac{no\ of\ records\ having\ the\ pattern\ p}{total\ number\ of\ records}$$

Move the support s_p to the set S

End

Find the average μ of supports listed in set s as

$$\mu = \frac{sum\ of\ supports\ of\ all\ patterns}{total\ number\ of\ patterns}$$

Find the root mean square deviation δ of the supports listed in set S as,

$$\delta = \frac{sum\ of\ absolute\ difference\ of\ average\ from\ each\ support\ listed\ in\ set\ S}{total\ number\ of\ supports}$$

Find the pivot P of the supports S of all patterns as follows

Find the frequent pattern coefficient fps as,

$$fps = \mu + \delta$$

Find the rare pattern support rps as

$$rps = \mu - \delta$$

C. Dynamic Support Range based Hybrid-Eclat Algorithm (DSRHEA)

DSRHEA is referred as hybrid Éclat because it uses the BFS (Breadth First Search) and DFS (Depth First Search) algorithms. The algorithm's characteristics are listed below.

The algorithm is based on Éclat and instead of $tid - set$, it uses bit-set to represent the transaction set of an itemset. The significant feature of this format is that it only scans data once to extract unusual patterns, which is necessary for stream data processing because it cannot be stored.

The support of one itemset is found by scanning the transaction block once. As explained in Algorithm1, a rare item in the current block may become frequent or vice versa in the data stream, hence all items with $support >= fps$ are examined and used to construct higher order items.

Prefix Node is used to store itemset since it saves memory. The prefix is stored once for all suffix items in the node, and the itemset is generated by concatenating prefixes and suffix items. The prefix is the first (n-1) item in an itemset, while the suffix is the nth item. $PrefixBVlist$ Is also utilized, which initially has a two-item $prefixbvnode$ that must be expanded to generate huge 'n' items, where n is a user-defined number.

The intersection of bit-sets of related parent one itemsets is used to generate two itemsets using BFS. The prefix items, with the exception of the suffix item, should all be the same to intersect any two items. Abd and acd, for example, can be intersected but not abd and acd since the initial $(n-1)$ items of Abd and acd, $(ab\ and\ ac)$ are not the same. The same criterion should be applied to higher order itemsets. A $prefixbvnode$ is constructed for each one item, with the supplied $oneitem$ as the prefix, and suffix items are generated for eligible two items, with an array list containing all $prefixbvnodes$ generated. For example, if the provided window has four items, let's call them $a,b,c,d,$ then the three $prefixbvnodes$ are constructed and added to the Array List. List of nodes a, b, $and\ c$, their respective suffix items.

To generate higher order item sets, the proposed algorithm uses an optimization based on two item sets,. As mentioned in Algorithm2, the generated huge objects are kept in a two-level $hashmap$. Algorithms 3 and 4 are used to update the $hashtable$ with the infrequent itemsets in the prefix node. The Stack data structure is used to implement DFS.

Algorithm 2: Dynamic Support Range based Hybrid-Eclat Algorithm (DSRHEA)

$DSRHEA()$

$blockid = 0;$

$pefefixBvNodeList\ pbvlist;$ //holds oneitems

$BlockSupports\ bsup;$ // block wise parameters

$Globals\ gb;$ //support parameters of datastream

While(datastream)

Begin

$blocksize = 0;$

While (currenttimeunit)

Begin updateitemslistwithnewtransaction();

$Blocksize++;$

End

$bsup = updateSupportthresoldsArrays();\ filteroneitemslist(bsup);$
$generateLargeItemsWithPrefixBasedBFSnDFS(rpt,\ gb,\ bsup);$

$Blockid++;$

$displayRarepatterns();$

End

1) Optimization based on bigram itemsets: To generate two items, the algorithm uses breadth first search, all two items are generated and stored into a $hashtable$, these two itemsets are used to prune the number of candidates while generating large item sets for eg: itemset $\{abcd\}$ is joined with $\{abcd\}$ to generate large item $\{abcde\}$ of size 5, however, if the itemset $\{de\}$ doesn't satisfy the support criteria for rare itemset, then obviously itemset $\{abcde\}$ will not satisfy the support criteria, so before joining itemsets, support $(n-1)^{th}$ item of the first itemset, $(n-1)^{th}$ item of the second itemset) is checked in two itemsets $hashtable$ to prune the unfruitful candidate rare itemsets. This pruning strategy reduces the number of candidate itemsets and intersection operations while generating large itemsets, hence the algorithm's efficiency is improved in terms of execution time.

Algorithm 3: Discovering

n- Itemsets

Method: generateLargeItemswithPrefixBasedBFSnDFS

Input: one Item list, fps root potential Items, $rpt, gb, bsup$.

Output: Set of $frequent\ pattern\ support\ > fps$.

$numOfOneitems < -oneitemsList.size();$
$rpt.insertOrUpdateOneItems(this.oneItemsList, gb, bsup);$
$oneItemI = 0;$

While $(oneItemI <\ numOfOneitems)$

Create a $prefixnode(\ pbv1)$

for a $prefix\ oneItemI;$

For each remaining $oneitem\ oneItemJ$

Perform inter section $(oneItemI, oneItemJ)$, calculate support and generate all two

Items using vertical mining in BFS that satisfy support criteria $(bsup.currentsupport2)$ andAppend to pbv1

$twoItemspbvnodelist.add(\ pbv1)$

$rpt.insertOrUpdateLargeItems(\ pbv1,\ gb);$ //update $hashtable$ with two items of current oneitem

For each $prefixbvnode\ pbv1$ in twoItems pbv node list

If(isExtendible $(\ pbv1)$) //extendible if node has more than one suffix items

$Pbvstack.push(\ pbv1)$

While $(!\ pbvstack.empty())$

$pbvcurrentnode = pbvstack.peek();$

If $(!isExtendible(\ pbv1))$

$rpt.insertorupdatelargeitem(pbvstack.pop())$

//the items in popped $prefixbvnode$ are inserted into $hashtable$ of $largeitems$

Continue;

Else

Create $pbvnewNode;$ //for storing generated large items

$Pbvnewnode.prefixitem.addall(\ pbvcurrentnode, prefixitemset)$

Generate and update $Largeitems$ //two itemset o optimization.

Push the generated node onto the stack if numberof $suffixitems\ in\ the\ node > 0$

2) Structure of two level hash-table: To store and maintain the patterns generated from the data stream, The tree is implemented as a Linear tree, implemented using array list, each entry in the array list is a $hashmap$ (level map), the level map keeps itemsets of same size, n level maps are to be created to store itemsets of n length, n is the max itemset size maintained. The structure of two-level $hashmap$ is shown in Fig. 1. The rare patterns are extracted from $hashtable$ that satisfy support criteria as given in rare item definition.

3) Discounting: When the number of blocks in window exceeds, the discounting of old block supports is performed, then the support of newly arrived block are added as described in algorithms 3,4 and 5.

Algorithm 4: insertOrUpdateOneItems

1. If $(globals.Blockid == 0)$
2. Create a $levelwisepreixmap\ and\ suffixmap$ with items in one items list
3. Else
4. If $(globals.Blockid >= gb.windowsize)$
5. Begin
6. $discountOldBlockSupports(bsup);$
7. End
8. Insert or update

$levelwisepreixmap\ and\ suffixmap$ with items in $oneitems$ list

Algorithm5: insertOrUpdateLargeItems

1. For $(;rootSize <= pfxLength;rootSize++)$
2. {
3. $rootpt.add(new\ LevelWisePrefixMap(rootSize, 200));$
4. If $(globals.Blockid == 0)$
5. Create a $levelwisepreixmap\ and\ suffixmap$ with items in $prefixnode$
6. Else
7. If $(globals.Blockid >= gb.windowsize)$
8. DiscountOldBlockSupports $(bsup);$
9. Create or update $levelwiseprefixmap\ and\ suffixmap$ with items in $prefixnode$

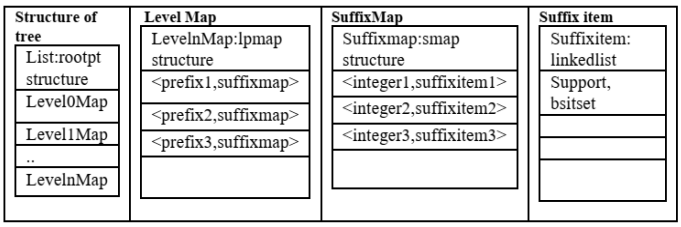

Fig. 1. Hash based Two Level Tree Structure.

IV. Experimental Results

Experiments have been carried on different datasets listed in FIMI repository [32].The implementation has been done using Java, which is interpreted on Intel core i5 2.4 GHz computer running Windows 10 and equipped with 8GB of RAM. To establish the performance significance of the DSRHEA, the rare patterns count and speed of execution have been compared with contemporary model SRPTREE. The details of the dataset, window size, and dynamic supports observed, as well as resultant values of the performance indicators are listed in Table I.

The above stated Fig. 2 to 4 demonstrates the rare patterns discovered by both proposed DSRHEA, and contemporary model SRPTREE. Fig. 2 to 4 states that the contemporary model SRPTREE having significant count of missing rare patterns that compared to the DSRHEA. It is noting that marginal ratio of missing patterns as 24%, 24%, and 1% in the respective order of datasets T10I4D100K, T40I10D100K, and Kosark (250K). According to these statistics, the missing pattern ratio of SRPTREE that compared to DSRHEA is proportionate to the number of rare patterns discovered.

Fig. 5 illustrates the process time observed from DSRHEA and SRPTREE to discover rare patterns from different datasets. The statistics related to process time indicating that DSRHEA is outperforming SRPTREE with minimal process time, which has observed as 22.807%, 47.466%, and 10.052% in the respective order of datasets T10I4D100K, T40I10D100K, and Kosark (250K).

TABLE I. Statistics of Inputs and Outcomes of the Experimental Study

Dataset details		Dynamic supports		Rare Patterns Count		Process time in mille seconds	
Dataset	**Block Size**	**FPS**	**RPS**	**DSRHEA**	**SRPTREE**	**DSRHEA**	**SRPTREE**
T10I4D100K	25K	0.00116	0.00054	16532	12539	22.807	129.602
T40I10D100K	25K	0.00899	0.00147	732166	557045	47.466	70.825
Kosark (250K)	25K	0.00492	0.00175	21075	21055	10.052	41.55

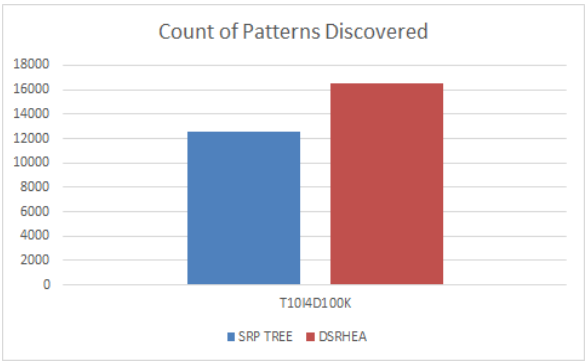

Fig. 2. Performance Comparison of DSRHEA and SRP Tree towards Rare Pattern Discovery from T10I4D100K Data Set.

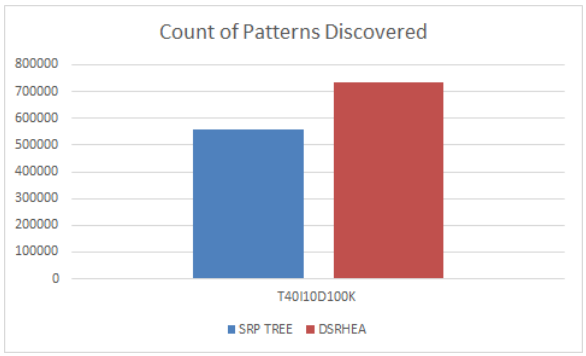

Fig. 3. Performance Comparison of DSRHEA and SRP Tree towards Rare Pattern Discovery from T40I10D100K Data Set.

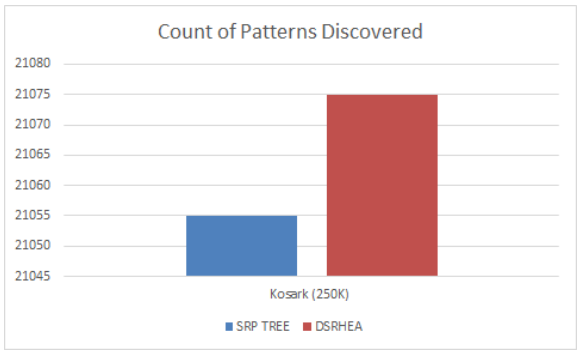

Fig. 4. Performance Comparison of DSRHEA and SRP Tree towards Rare Pattern Discovery Kosark Data Set.

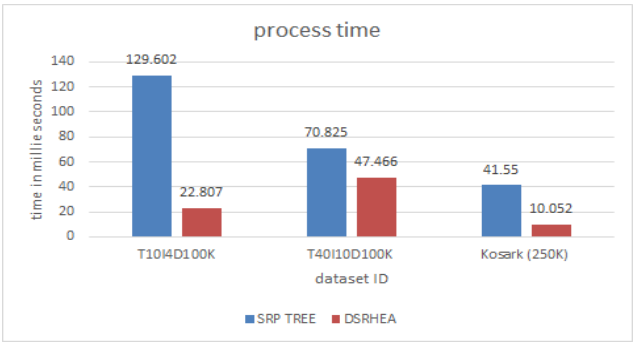

Fig. 5. Performance Comparison of DSRHEA and SRP Tree towards Processing Time.

A. Cross Validation

The performance study was carried out using cross-validation. The DSRHEA cross-validation metrics were compared to the values obtained in an experimental investigation on the same dataset utilizing modern models HECLAT RPStream and SRPTree [20] with static and dynamic supports. The comparison investigation is presented in full in the following cross-validation metric examination.

Human annotation was used to identify high utility itemsets with changeable values of the six items in order to undertake cross-validation. Human annotation partitioned the total itemsets identified into two, reflecting positive and negative labels in descending order. There are a total of 21794 rare itemsets that have been considered. Among these itemsets, 14206 have been labeled as positive (really uncommon itemsets), whereas 7588 have been designated as negative (falsely rare itemsets). Positive itemsets have limited support (rare support) and maximal confidence, while negative itemsets have confidence less than the threshold and redundant itemsets. The experimental statistics (cross-validation metrics) are provided in the following performance analysis.

The DSRHEA model outperformed the other alternatives significantly, as illustrated by the figurative illustration in Fig. 6.

The metric accuracy denotes the positive predictive value, which is the ratio of true positives to the total of true positives and false negatives. The metric precision values for DSRHEA, HECLAT RPStream, and SRPTree with static and dynamic support were 0.997, 0.9584, 0.8976, and 0.9175, respectively. According to these results, the DSRHEA model exceeds the other models in terms of positive pattern predictive value, as determined by precision.

The measure specificity can be used to calculate the real detection rate of negatively labeled uncommon patterns, which is derived as the ratio of true negatives to the sum of true negatives and false positives. The specificity shows the best method for removing patterns that aren't actually unusual (having negative label). The metric specificity values for DSRHEA, HECLAT RPStream, and SRPTree with static and dynamic support were 0.9836, 0.9657, 0.8836, and 0.9136, respectively. These numbers suggest that the DSRHEA model surpasses the other models in spotting patterns that aren't actually infrequent, as shown by specificity.

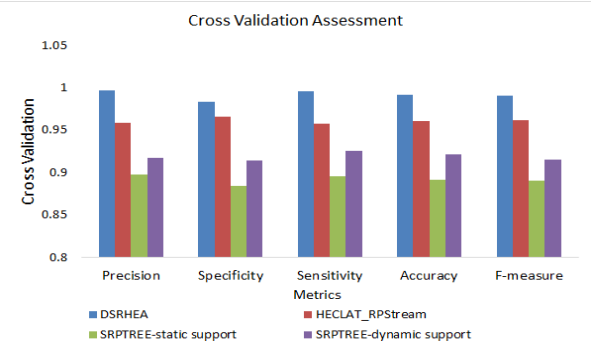

Fig. 6. The Related Performance Metric Values of the Models DSRHEA, HECLAT_RPStream, and SRPTree with Static and Dynamic Suport Discovered from Cross-Validation.

The true positive rate is calculated using sensitivity, which is the ratio of true positives to the sum of true positives and false negatives (rate of truly rare patterns discovered). The metric sensitivity values for DSRHEA, HECLAT RPStream, and SRPTree with static and dynamic support were 0.9957, 0.9577, 0.8957, and 0.9257, respectively. These values suggest that the DSRHEA model surpasses the other models in terms of detecting truly unusual patterns, as measured by sensitivity.

The metric accuracy used to characterize measurement approximations towards true-value is the ratio of the count of actually discovered rare patterns, truly discarded patterns that are not unusual, to the total count of patterns discovered and rejected (discovering truly rare patterns and discarding truly not the rare patterns). The metric accuracy values for DSRHEA, HECLAT RPStream, and SRPTree with static and dynamic support were 0.9915, 0.9605, 0.8915, and 0.9214, respectively. As assessed by accuracy, these values reveal that the model DSRHEA beats the other models in terms of detecting actually unusual patterns and eliminating those that aren't.

The metric F-measure, which is the reciprocal of the ratio of "product of precision and sensitivity" to "sum of the corresponding precision and sensitivity," represents the consistency of accuracy and sensitivity values displayed in the rare pattern mining process. The metric f-measure for DSRHEA, HECLAT RPStream, and SRPTree with static and dynamic support was 0.9903, 0.962, 0.8905, and 0.9155, respectively. In terms of consistency of values depicted for accuracy and sensitivity, which is represented by the f-measure, the model DSRHEA obviously beats the other models.

V. Conclusion

This article demonstrates how to implement rare pattern mining from data streams using a time-sensitive sliding window. The methods are éclat-based and implemented using bit-sets. The approach makes use of two-level hash tables to store newly formed large rare patterns. An optimization based on the support of two item sets is also a critical element of the method, as it increases the overall efficiency of the program in terms of time complexity by lowering the number of candidates and their join. In this application is used a range of input support criteria to do data analysis. Support is calculated dynamically. The proposed method has been proved to be more effective than earlier relevant algorithms. The essential points are as follows: 1) Two-item optimization, 2) Prefix node-based massive item generation, 3) Vertical mining with bit-sets instead of TID-sets are two ways to make more items, and 4) Calculation of dynamic support. The cross validation results indicating that the proposed DSRHEA is outperforming the other contemporary models in regard to identify truly rare patterns. The DHREA has shown 99% accuracy to discover truly rare patterns. The evolutionary techniques such as fuzzy reasoning can be used to discover rare patterns under variable contexts, which will be the future research.

References

[1] H. T. Nguyen, L. P. Phan, H. H. Huynh, and H. X. Huynh. (2021). “Collaborative Recommendation based on Implication Field,” Int. J.

Adv. Comput. Sci. Appl., vol. 12, no. 10, 2021, doi: 10.14569/IJACSA.2021.0121003.

[2] Ngai, E. W., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decision support systems, 50(3), 559-569.

[3] Liu, Y. (2010, January). Study on application of apriori algorithm in data mining. In 2010 Second international conference on computer modeling and simulation (Vol. 3, pp. 111-114). IEEE.

[4] Kruse, S., & Naumann, F. (2018). Efficient discovery of approximate dependencies. Proceedings of the VLDB Endowment, 11(7), 759-772.

[5] T. Slimani and A. Lazzez, "Efficient analysis of pattern and association rule mining approaches," International Journal of Information Technology and Computer Science, 6(3), pp. 70–81, 2014.

[6] Yanqing Ji, Hao Ying, John Tran, Peter Dews, Ayman Mansour, and R Michael Massanari. (2012). A method for mining infrequent causal associations and its application in finding adverse drug reaction signal pairs. IEEE Transactions on Knowledge and Data Engineering, 25(4):721–733, 2012.

[7] Yun Sing Koh and Sri Devi Ravana. (2016). Unsupervised rare pattern mining: a survey. ACM Transactions on Knowledge Discovery from Data, 10(4):1– 29, 2016.

[8] Kanimozhi SC Sadhasivam and Tamilarasi Angamuthu (2011). Mining rare itemset with automated support thresholds. Journal of Computer Science, 7(3):394, 2011.

[9] Darrab S, Ergenic B (2017) Vertical pattern mining algorithm for multiple support thresholds. In: International conference on knowledge based and intelligent information and engineering (KES). Procedia computer science, vol 112, pp 417–426.

[10] Yun Sing Koh and Nathan Rountree. (2005). Finding sporadic rules using aprioriinverse. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 97–106. Springer, 2005.

[11] Laszlo Szathmary, Amedeo Napoli, and Petko Valtchev. (2007). Towards rare itemset mining. In 19th IEEE International Conference on Tools with Artificial Intelligence, pages 305–312. IEEE, 2007.

[12] Luigi Troiano, Giacomo Scibelli, and Cosimo Birtolo. (2009). A fast algorithm for mining rare itemsets. In Ninth International Conference on Intelligent Systems Design and Applications, pages 1149–1155. IEEE, 2009.

[13] R Uday Kiran and P Krishna Reddy. (2011). Novel techniques to reduce search space in multiple minimum supports-based frequent pattern mining algorithms. In Proceedings of the 14th International Conference on Extending Database Technology, pages 11–20, 2011.

[14] C Sweetlin Hemalatha, Vijay Vaidehi, and R Lakshmi. (2015). Minimal infrequent pattern based approach for mining outliers in data streams. Expert Systems with Applications, 42(4):1998–2012, 2015.

[15] M.A. Thalor and S. Patil, "Incremental Learning on Non-stationary Data Stream using Ensemble Approach," International Journal of Electrical and Computer Engineering, Aug 1;6(4):1811, 2016.

[16] David Tse Jung Huang, Yun Sing Koh, Gillian Dobbie, and Russel Pears. (2014). Detecting changes in rare patterns from data streams. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 437–448. Springer, 2014.

[17] Tanbeer, S. K., Ahmed, C. F., Jeong, B. S., & Lee, Y. K. (2009). "Efficient single-pass frequent pattern mining using a prefix-tree", Information Sciences, vol. 179(5), pp. 559–583, 2009.

[18] Deypir, M., Sadreddini, M. H., & Hashemi, S., (2012). "Towards a variable size sliding window model for frequent item set mining over data streams", Computers & industrial engineering, Vol. 63(1), pp. 161–172, 2012.

[19] Gangin Lee, Unil Yun , Keun Ho Ryu, (2014). "Sliding window based weighted maximal frequent pattern mining over data streams", Expert Systems with Applications, vol. 41, pp. 694–708, 2014.

[20] David Huang, Yun Sing Koh, Gillian Dobbin. (2012)."Rare Pattern Mining on DataStream" Data Warehousing and Knowledge Discovery Lecture, Notes in Computer Science Volume 7448, 2012, pp 30.

[21] Tsang, S., Koh, Y.S., Dobbie, G. (2011). RP-Tree: Rare Pattern Tree Mining. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2011. LNCS, vol. 6862, pp. 277–288. Springer, Heidelberg (2011).

[22] SunithaVanamala, L.Padma Sree, S.Durga Bhavani. (2013). "Efficient rare association rule minig algorithm " Int. J. of Engineering Research and Applications (IJERA) , Vol. 3, Issue 3, pp.753-757 , 2013.

[23] Vanamala, S., Sree, L., & Bhavani, S. (2014). Rare association rule mining for data stream. International Conference on Computing and Communication Technologies, 1-6.

[24] Manal Almuammar, Maria Fasli. (2018). "Learning Patterns from Imbalanced Evolving Data Streams", Big Data (Big Data) 2018 IEEE International Conference on, pp. 2048- 2057, 2018.

[25] Manal Almuammar, Maria Fasli. (2017). "Pattern discovery from dynamic data streams using frequent pattern mining with multi-support thresholds", the Frontiers and Advances in Data Science (FADS) 2017 International Conference on, pp. 35-40, 2017.

[26] Vanamala S., Padma Sree L., Durga Bhavani S. (2021) Eclat_RPGrowth: Finding Rare Patterns Using Vertical Mining and Rare Pattern Tree. Computer Networks, Big Data and IoT. Lecture Notes on Data Engineering and Communications Technologies, vol 66. Springer, Singapore. https://doi.org/10.1007/978-981-16-0965-7_14.

[27] W. A. W. A. Bakar, M. Man, Z. Abdullah, and M. B. Man. (2021). "CRS-iEclat: Implementation of Critical Relative Support in iEclat Model for Rare Pattern Mining," Int. J. Adv. Comput. Sci. Appl., vol. 12, no. 7, 2021, doi: 10.14569/IJACSA.2021.0120722.

[28] Zaki M (2000) Scalable algorithms for association mining. IEEE Trans Knowl Data Eng 12(3):372–390.

[29] Rahul Anil Ghatage. (2015). Frequent Pattern Mining Over Data Stream Using Compact Sliding Window Tree & Sliding Window Model (IRJET) , Volume: 02, July-2015, eISSN: 2395 -0056 p-ISSN: 2395-0072 15.

[30] Anindita Borah, Bhabesh Nath. (2019). Incremental rare pattern based approach for identifying outliers in medical data, Applied Soft Computing,Volume 85,2019.

[31] https://www.investopedia.com/trading/using-pivot-points-for-predictions/.

[32] Frequent itemset mining dataset repository. http://fimi.uantwerpen.be/data/.