Innovation In Data Mining Repeating Patterns Of Data Stream Dependent and Uncertain

Sima jahanshahi[1] and Ali Asghar Safaei[2]

[1] Department of Computer Engineering, University International, Qeshm, Iran
[2]Medical Informatics Group, University of Medical Sciences, Tarbiat Modares University, Tehran, Iran
safaei@eetd.kntu.ac.ir

Abstract: In this paper, four different topics such as exploration of repetitive patterns, uncertain data mining, data stream mining and exploring a set of functional dependencies combined to explore repeating patterns in a stream of data dependent was used. To explore repeating patterns of flow, obscuring the windowing technique Landmark and fade time and also to explore the dependence of AD-Miner algorithm is used. The presented algorithm, compared with the existing algorithm, the UF-Streaming. To assess their own method implemented in c and c ++ and then the same conditions are compared to existing methods. The results show that the proposed method of using the window technique Landmark and disappears when it is more efficient than the existing method that uses the slideshow window.
[Sima jahanshahi, Ali Asghar Safaei. **Innovation in data mining repeating patterns of data stream dependent and uncertain.** *Researcher* 2014;6(11):14-22]. (ISSN: 1553-9865). http://www.sciencepub.net/researcher. 3. doi:10.7537/marsrsj061114.03.

Key words: Data stream- Land mark window- data dependent - -Sliding window – Time Fading window – Data stream dependent and uncertain

1. Introduction

Explore the repetitive patterns is an important research field in the database and the data are explored. This method patterns which frequently appear in data sets have been introduced as repetitive patterns. This method in most everyday applications, such as data sets related to the sale of the supermarket, data sets related to the weather and environmental data sets used. However, a great deal of research has been done on exploring repeating patterns and most of them are focused on precise data; so that users have accurate information about the item in the data set. But in most everyday situations (real world), Member of the presence or absence of items in the data set, the data are not exact (quantum physics dataset, the dataset of thermal sensors for environmental monitoring and data collection). This type of data, called uncertain data that the presence or absence of an item with a probability, we show (which is also called indeterminate amount). In addition, the collection of data on the real world is not static and will always have a steady stream of new data with fast speed. Explore the dynamics of this type of data collection, data flow, have been explored.

Literature Review

Algorithm Apriori [1] in 1994 by Agrawal and srikant to explore specific static data sets, respectively. Each pattern is associated with a level of confidence that the data set describes a number of models.

Model (or set of items) to be considered as a repeating pattern, if and only if the value is greater than or equal to the pattern assures and a certain minimum threshold have set by the user. Algorithm Apriori, will run from the top down and the framework for exploring repeating patterns in data sets identified the test. To solve this problem, algorithms based on Apriori, Hen et al [3], FP-growth algorithm was presented in 2000 as the candidate of the production process to prevent. In this algorithm, there are two processes: (1) creating repetitive patterns of tree growth FP, 2- tree FP, a generalization of the tree structure that receives the contents of the data set. The algorithm requires a review of the data set to form the FP-tree.

The first is to follow the same patterns. All models are unique and delete all duplicate patterns by reducing the amount of trust units ordered. Similarly, the potential size of the FP-tree is also reduced. Then, algorithms, data sets for the second time to check the FP-tree shape. After creating the tree, we need to examine the data set. Tree FP, all information contained in the data set to be included. Then all repeating patterns can be grown by a process of recursively obtained from any branch of the tree. FP-Streaming algorithms have been proposed by Giannella et al in 2004 [3] that it is a specific data stream mining algorithms. As previously described, the first stage of the FP-Streaming, called the FP-growth is a preMinsup explore the data available. The FP-growth algorithm finds all repeating patterns, the next step of the algorithm FP-Streaming, storage and maintenance of these patterns is a tree structure called FP-Streaming. The FP-Streaming, each track shows a repeating pattern. Each node in the FP-Streaming Schedule window contains slides that have multiple levels of confidence for each transaction category. Note that, FP-Streaming Algorithms for Data Mining data streams are clear. The problem comes when we want to process streaming data is uncertain and there are many challenges in this field. UF-Streaming algorithms have been proposed by Hao and Leung [14] the repetitive patterns from a population of uncertain data streams using the window slides explore the fixed size of the conventional.

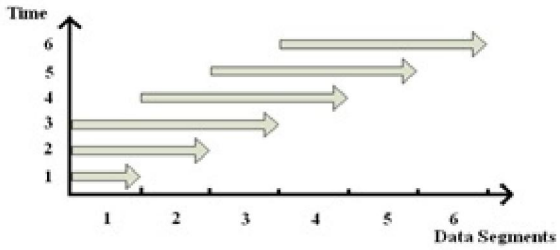
UF-Streaming algorithm first calls to repetitive patterns from among those available in the current transaction (using a minimum threshold of reliability). Repetitive pattern that is larger than the threshold value, assure safety is minimized. The UF-Streaming repetitive

patterns in the structure of a tree, assure the values stored in such a way that at each node, X is a list of values to be stored safely w. When a new stream is released, sliding windows and assure the values are pasted repeating patterns and amounts expected to ensure that the window is inserted through the newest trend, and they are exploring the oldest found in the window displays. This process is repeated for each class of flows. The reliability of each repeating pattern X, the sum of all values ensures that w (one for each category in the window slide) is calculated. ExpSup (X, Bi) the uncertainties related to X in the category Bi shows. Then, in a moment of T X amount of confidence in the slide window that contains w handle uncertain data the categories of $B_{T-w+1},...,B_T$ is calculated as follows:

$$\exp Sup(X, \bigcup_{i=T-w+1}^{T} B_i) = \sum_{i=T-w+1}^{T} \exp Sup(X, B_i)$$

Window model slide

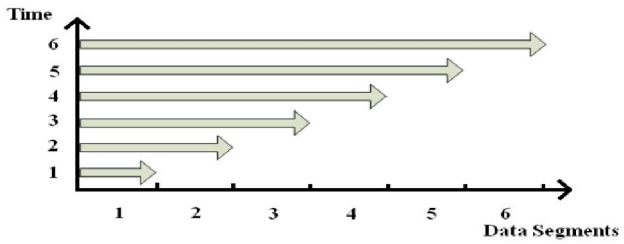
The window slides into user needs the size of the sliding window and the window in a set period of time [15].



As seen in the figure, the vectors show the windows. Initially, the first piece covers the w window. When the next period is reached, the window slides up 2 sizes grow. Then, in the third period, the window slides up to A3 size grows to the maximum capacity of the slideshow window in the model. In the next window period, the amount of slip to the right, and the second fragment to fragment 4 covers and a fragment from the right end to remove it (the oldest piece of data) to maintain the size of the window. The process is repeated sliding window is shown in Fig.

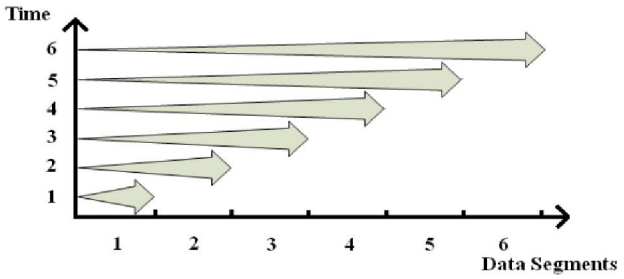
Landmark Window Model

Model window Landmark, the technique is not a fixed window size [16, 34-38,45]. From a given point in time begins and ends in the window size increases. Note that the fixed point is the end point moves with time. As shown in the figure, the left pane of the fragment Landmark begins. A window is a piece of data. In the next period, a Landmark endpoint window to move to the next piece of data, but the basic point remains constant. Over time, the size of the window increases Landmark and every piece of data is stored [46,47].



Window Model for Time Axis

The model assumes that the window disappears when the data is more accessible than the old data [17,41-43]. The model for each fragment, the amount of weight it takes into account the weight is reduced with time. In other words, over time, to share a little older than the set of all data. How to process the data in a model when the window disappears. When the model reaches the next data block and the old data is less important, the window disappears when the right starts [44,51]. The starting point is always the leftmost side of the window when it starts to fade. It can be observed that, unlike the slide window, the window fades as time passes, the fragment does not delete. Also, unlike the window Landmark, the data do not consider the same for everyone. Remove old data so that users can focus on the data available.



Conditional Limitations

Terms of upper and lower boundaries of the areas of feature or review was considered. Minimum and maximum values for all attributes in all the relationships we choose the corresponding SQL commands. Thus, the normal SQL commands for character and numeric values for character sets in lexicographical order of the symbol character traits. Since the calculation of the overall cost calculation in a query may be O (n * m) is. Where n is the number of attributes in all tables in this complex and m is the maximum value of the tuple in the table.

Dependence of the unit

Principle of inclusion dependencies can be used to compute transitivity and implementation of all components in a specific order.

Functional Dependencies

Since functional dependencies related to database schema design is essential, the study is the application of significant importance. Fully functional dependence has been studied for decades. Functional dependency relationships between attributes of a relation are:

Functional dependence implies that the value of an attribute using the some other unique feature-set. Classical functional dependencies in the relational database schema design, normalization of relations, and in order to avoid redundancy and data anomalies blades are used. This class of dependencies, no resilience to exceptions and potential noise data whatsoever was considered. The approximate dependencies, those dependencies are others who do not hold the entire data set, resulting in noise and flexible exception**.**

Approximate dependencies search algorithm

Obtain a minimal set of all dependencies of a search algorithm, called AD-Miner use. For each attribute A of such a relation schema R, the dependence of the form to search for all possible combinations of features that are minimal or to the left to find the dependence**.** The search process for a fixed sequence of attributes (eg alphabetically) we consider. List of features that can be used for left-dependence, all attributes of R, other than the A. Left to find dependencies, each character in turn and the beginning of the end of the list are selected. For each attribute X is selected as a depth first search procedure to apply has a minimal set of features that include X, and the dependence of the degree of accuracy is acceptable [48].

Key features of AD-Miner algorithm

This algorithm is the elasticity with respect to exceptions and noisy data and the accuracy of all the discovered dependencies are calculated. In addition, this algorithm is dependent on the position of each tuple that is not true dependence (index tuples in a relation) is also indicated. This feature is especially useful when you want to find exceptions and data in a data set to be incompatible. Another important feature of AD-Miner, an increase of it is considered [51].

Explore the repetitive patterns of the model window Landmark

Landmark popup window size is not fixed model techniques. This model every piece of data from the beginning to the end of a given point of time when the current folder. Because the model considers all data equally, no data will be discarded. Also, the window size can be considered as additive. Proposed algorithm works as follows.

Consider the uncertain stream of data

And a user-specified minimum threshold to consider (minsup) is vital. LUF-Streaming algorithms proposed Landmark model in an environment with uncertain data for exploring repeating patterns of data flow is used (ie, the expected value is greater than or equal to the confidence threshold). The algorithm includes three main models:

Explore the batch

Explore the current

Hersey Model

Explore the batch

This model of repetitive patterns in each batch of call flow UF-growth algorithm is explored. Data flows are not necessarily uniformly distributed. A unique pattern may be repeated later. Here, instead of using UF-growth algorithm with a confidence threshold minsup, the models UF-growth algorithm using a lower threshold preMinsup helps prevent shedding pattern.

For this reason, sub-models, each repeating pattern X (ie X greater than or equal to the expected reliability preMinsup) in any class of uncertain data flows on it. The handle can make X amount to the sum of (most transactions on) product (probability independent item in the pattern of X) is calculated by the following equation can be expressed. [36]

$$\exp Sup(X, B_j) = \sum_{i=1, t_i \in B_j}^{w} (\prod_{x \in X} P(x, t_i))$$

In this equation, w is the number of categories.

Exploration model streaming

Explore the repetitive patterns in the sub-model of the flow in a tree structure maintains the LUF-Streaming-called each node of the tree a template and contains information about the reliability of the current (from the beginning up to the present Landmark).

Hersey Model

Finally, when the user requests a search final results, the repetitive patterns of sub Hersey's definitely not a duplicate, the pruning. Note that the patterns are pruned to avoid the rapid exploration of the batch; repetitive patterns preMinsup finds a minor (ie, models with much greater certainty than the preMinsup that may be smaller than minsup). The repetitive pattern in the sub-LUF-Streaming maintenance and reliability expected values by exploring the model is updated. When a user requests a certain repetitive patterns Hersey model these patterns with much greater reliability equal to or less than minsup eliminates was for preMinsup [52,53].

Explore the repetitive patterns with a model-based window

Model Window Landmark whole pieces of data with the same degree of care considered. However, in everyday applications, users are demanding more focus on current data (but do not want to lose the influence of old data). For example, in a supermarket dataset, where managers are now calling for a greater focus on the more popular products was important. At the same time, they will not completely discard the old sales data.

To meet this vision, the window disappears when a method is correct. Here, we introduce our TUF-Streaming Algorithms Group (which consists of three algorithms) that uses time-fading window model.

1) A simple algorithm called TUF-Streaming simple.

2) Algorithm, with storage space that TUF-Streaming (space) is to reduce the amount of memory used.

3) Algorithm with storage time of the TUF-Streaming (time) is the algorithm reduces the execution time.

Simple algorithm: TUF-Streaming (Simplified)

The first group of algorithms TUF-Streaming is simple. The main steps of the algorithm are described as follows. First, for each class of uncertain data streams, with preMinsup UF-growth algorithm, we used to have to

find repeating patterns (i.e. patterns with much greater confidence pre Minsup than one category). Then, repeating patterns to explore and make their expected values in a tree structure called the TUF-Streaming, save a tree where each node, corresponding to an X pattern to ensure it maintains a list of values. Note that the window disappears when not slide and can grow over time.

Explore the accession process TUF-Streaming for each batch in the data stream is not repeated. By letting X amount of confidence in our views. Then, at time T, the value of the fade time can make X probe with a total maximum value of certain categories of X calculated (by the fade factor is established when called weight).

$$\exp Sup(X, \bigcup_{i=1}^{T} B_i) = \sum_{i=1}^{T} (\exp Sup(X, B_i) * \alpha^{T-i})$$

Although simple TUF-Streaming finds all repeating patterns, it may be a large amount of memory is required. As a continuous data stream is infinite, make X amount of storage for each category of flows can be important. Because it can create a list (window) for each node is unlimited. A detailed analysis of this equation shows that the total of the amounts to make sure X is the weight of X in all categories.

Unlike the slideshow window (which requires shedding old category), the elimination of the need for disposal or removal of those not older is important. Instead, a greater weight to those older than those in consideration was considered. As a specific example, for the Landmark, the algorithm weights the same for all categories (as they think they are old or new categories) accounted for. However, the algorithm is storage space under the TUF-Streaming (space) we introduce the need to maintain the sequence is detail for each category. Equation in a recursive function is expressed as follows:

$$\exp Sup(X, \bigcup_{i=1}^{T} B_i) = [\exp Sup(X, \bigcup_{i=1}^{T-1} B_i) * \alpha] + \exp Sup(X, B_T)$$

By doing this, the algorithm is only a single value (ie $\exp Sup(X, \bigcup_{i=1}^{T} B_i)$ instead of $\exp Sup(X, B_i)$ infinite list of maintenance.

Storage algorithms Time: TUF-Streaming (Time)

TUF-Streaming (space) in the vast amount of space required to store an infinite list of values to ensure the reliability of each node in the TUF-Streaming decrease. However, the recursive formula shown in Equation 3.4, we see that the reliability of the pattern X to make X amount of time T directly depends on the time T-1. For this reason, TUF-Streaming (space) required to meet any node in the TUF-Streaming After exploring each category (even if the corresponding pattern on the handle, not repeated) In order to calculate the reliability for repetitive patterns. On one hand, it may incur long runtime; on the other hand, if you need to check the nodes in the same cluster, then the results are perhaps not accurate.

Analytical Assessment

The discovery of the dependence of the function, we assume that | R | = n The number of attributes in Asky may relation R, | r | = m the number of attributes of relation r (which has the structure R is), | fr | = f the number of attachments that are evaluated There, | LHS | maximum number of attributes attached to the left and | D | = d in the range attribute values are the average number of discrete values (ie, the number of members of the MB-Set on). The algorithm used to calculate the MB-Set the operation of the order of O (nm), and the set M to a dependence of order O (d | LHS |) is. Because each member of M, the number d logical AND operation must be performed, the complexity of computing the set F of order O (d | LHS | +1) and the total overall complexity of the algorithm is O (nm + fd | LHS | +1 high).

The condition occurs when the left needs to develop any of the dependencies is not found, in other words the degree of accuracy of all attachments that include the first option on the left is acceptable. In this case 1 | LHS | and f n2 has the lowest mean value. Thus the overall complexity of the algorithm is at best of order O (nm + n2.d2).

In this case, all dependencies should be left to develop enough left no option for adding to the left. Thus | LHS | n- 1 will be equal to the total number of attachments that are testing the entire search space (ie, 2 N dependence) is included. But in practice, this condition is extremely rare, especially in the case of real data.

Then, five algorithms for the detection of repetitive patterns from uncertain data streams using the model window and the window disappears when we introduce Landmark. In this section, we explore a number of categories displays repetitive patterns [50]. Two algorithms are proposed to assess the amount of memory consumption and run it.

The amount of memory used

When using the Model window Landmark, LUF-Streaming algorithms for storing the LUF-Streaming expSup amount of information needed to store repetitive pattern. When using the fade time, the algorithm TUF-Streaming (simple) requires a lot of space between the proposed algorithm is 5 because it requires a certain amount of reserves in the TUF-Streaming (where w is the number of classes exploring the Show that).

Instead, TUF-Streaming (space) requires a minimum amount of space because each node stores only a single value (ie a total amount of values), while TUF-Streaming (time) requires more space than the TUF-Streaming (space) is, for each node of the LV with the confidence to keep repeating pattern nicely (ie for a total amount of amount of amount of confidence + values). However, this method is limited (cf. an infinite list of values to ensure the TUF-Streaming (simple) for unlimited w).

Moreover, the algorithm TUF-Streaming (when) a slight increase in the space normally reduce execution time was considered. In other words, TUF-Streaming as the TUF-Streaming (space) acts. The amount of memory used in the algorithm TUF-Streaming (with delay) will be equal to (the amount of memory the TUF-Streaming (space) uses) with buffering delay () for each node of the tree (for each pattern repeat) collects there. Next, ensure the + value for the total amount will be considered. TUF-Stream stored in the tree.

Runtime

Then, as time runs out for the window Landmark, the algorithm needs to meet LUF-Streaming / update each node in the tree LUF-Stream for each category is given. However, we can raise the performance of the procedure so that only the tree nodes corresponding to the repeating patterns in the data stream or updated to meet the next.

In window model, both TUF-Streaming (simple) and TUF-Streaming (space) required to meet at TUF-Stream is an update of a node corresponding pattern is repetitive, despite whether or not the node for Surround b for a total amount meets. Instead, TUF-Streaming (time) only the nodes corresponding to the repeating patterns in the data stream needed to meet the current node is met.

Similarly, the theory is applied to the Landmark Model, except that they require much less runtime than the fade time and this is because the calculations easier. To simplify the calculation of the equation when it is easier to be sure (for Model Landmark), such expressions became simple.

Assessment Tests

In this section, we evaluate the proposed approach. Performance of the algorithms on two real datasets from the UCI database and evaluated.

1) Data Collection Nursery

2) Datasets Mushroom [37].

Stats dataset used is presented in Table 1. Table A.

Data Collection	Number of properties	Number of tuples
Nursery	**9**	**12960**
Mushroom	**22**	**8124**

Presented data sets with numerical attributes by incorporating their values are considered. Before starting the operation, the values of these attributes to partition the range into 5 equal parts were converted into discrete values. To equate the two algorithms, the threshold value of 1.0 is considered equal accuracy. Performance of the algorithm AD-Miner (which is more efficient than other methods explore dependencies) on a data set and obtains the results are shown in Table 2.

Datasets	AD_Miner (s)
Nursery	82
Mushroom	54

AD-Miner algorithm is capable of exploring affiliation with any degree of accuracy is. Then, too, such as TUF-Streaming (delay) the process of TUF-Streaming (space) runs out, ie run time, based on the evaluation of the obtained results, and they are almost identical.

This dataset, the dataset is known in the field of data mining. For datasets Nursery, we prepare two sets of data, (60-50) and (100-10). The numbers of data sets show the range of probability. For example, (60-50) in the data set means that the entire set of items (patterns), values between 0.5 and 0.6 have confidence.

Similarly, (100-10) in the data set, this means that all patterns, confidence levels will be between 0.1 - 1. Nursery data set consists of 12,960 flows. Our streams are separated into 20 separate data sets (each set consists of 648 streams). Mushroom data set for a total of 8124 stream there. However, we divided the data into 20 sets of data-set (on average, each batch consisting of 406 current or transaction). For each data set described above, we will review all aspects of the proposed method. 1) Influence of categories on runtime and 2) the impact on runtime minsup threshold set by the user. Then, the algorithm with the best performance compared with algorithms for uncertain data streams, the UF-streaming select Explore.

Assess the various categories of data

Testing 1. the empirical evaluation, we run the algorithm with different number of categories on the Nursery dataset compared. Minsup threshold of 1.2 is considered at all stages of the algorithm. For a proper comparison, all algorithms are proposed for some of TUF-Streaming with a note. By doing so, the search results returned by the LUF-Streaming with whole algorithm are in TUF-Streaming. In Figure 1, the x-axis and y-axis of the handle run-time show is considered. Figure 1 shows the running time and the number of data sets increased.

We observe that TUF-Streaming (simple) is the most time while both LUF-Streaming and TUF-Streaming (time) have the least amount of time.

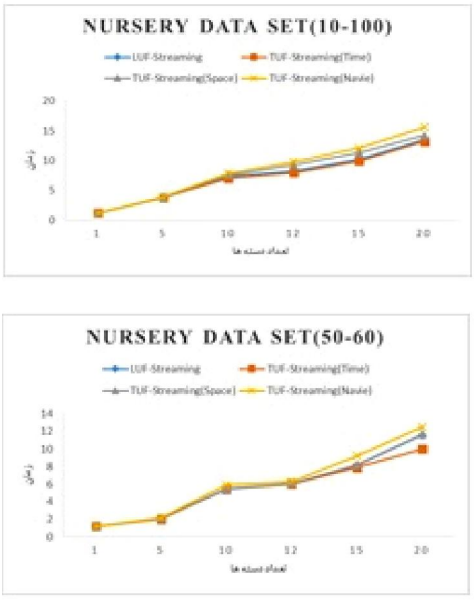
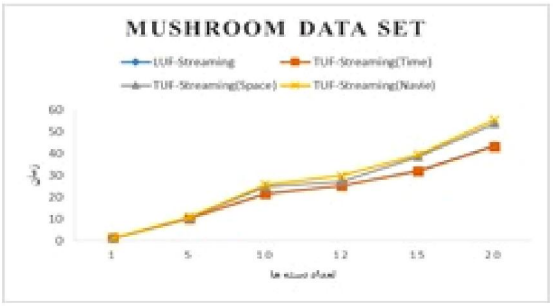
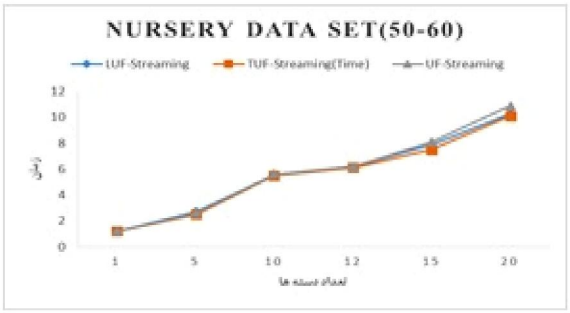


Figure 1 Evaluation of the various categories of the dataset Nursery (Experiment 1)

Experiment 2 Figure 2 shows the evaluation results of the Mushroom dataset. In this part, the running time of the algorithm for different number of clusters are

compared. Again, the threshold value is equal to 1.2 for all algorithms we consider a TUF-Streaming equal.





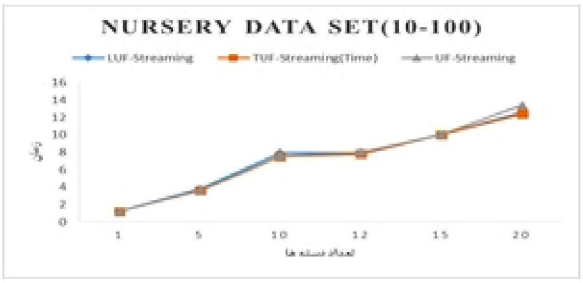
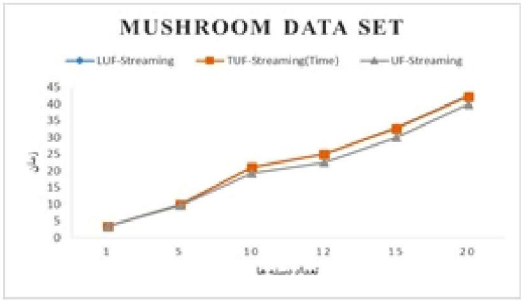


Figure 3 Evaluation of the various categories of the dataset Nursery (Experiment 3)

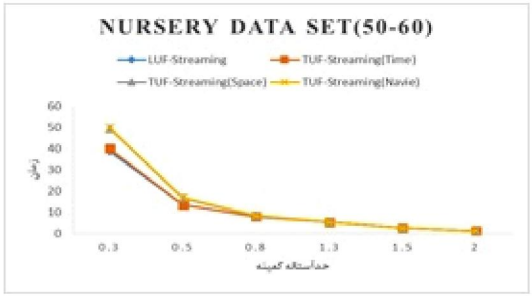
The whole TUF-Streaming (simplified) more time it takes. LUF-Streaming and TUF-Streaming (when) you have less time is considered. Both algorithms LUF-Streaming Experiment 3 and TUF-Streaming (time) show the best performance in terms of runtime. Same ways as for the previous experiments to compare these two algorithms with the existing algorithms were used. At first, we all algorithms for datasets Nursery (Figure 3) and the datasets Mushroom (Figure 4) to increase the number of batches run.

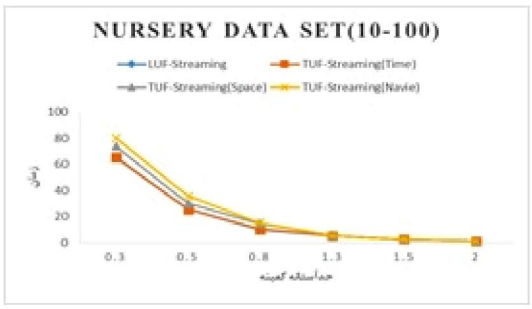
Possible reason is that these algorithms are used for

ry.
to
ble
ıen



the





set
Nursery (Experiment 4)

Experiment 4 In this experiment, all algorithms are presented for different threshold on both datasets Nursery (Figure 5) and the datasets Mushroom (Figure 6) are evaluated. In both figures, the x-axis and y-axis threshold increased running time. The number of data sets in each run of 10 put and again for a proper evaluation, the proposed values for all algorithms we consider a TUF-Streaming equal. We see the results of the

LUF-Streaming and TUF-Streaming (time) had the slowest time of the algorithms tested.

Experiment 5 this experiment, run LUF-Streaming and TUF-Streaming (time) and UF-Streaming on the dataset Nursery (Figure 7) and the datasets Mushroom (Figure 8) with a threshold of comparison. Experiment 3 because of that number returned by UF-Streaming repeating patterns of repetitive patterns returned by the algorithm is proposed.

This is because the UF-Streaming Slideshow Windows uses to delete data. However, we observe that our algorithm runs only slightly slower than the UF-Streaming.

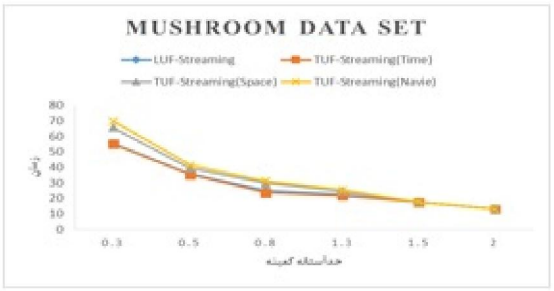
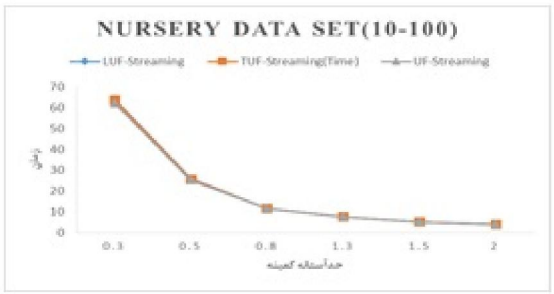


Figure 6 shows the evaluation threshold of the datasets Mushroom (Experiment 4)



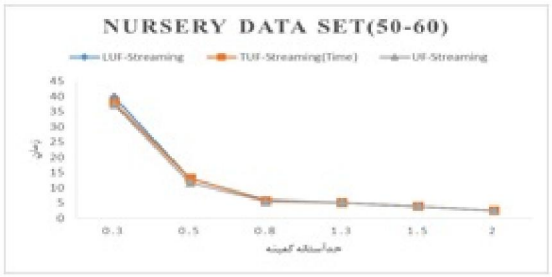


Figure 7 Evaluation of the dataset threshold Nursery (Experiment 5)

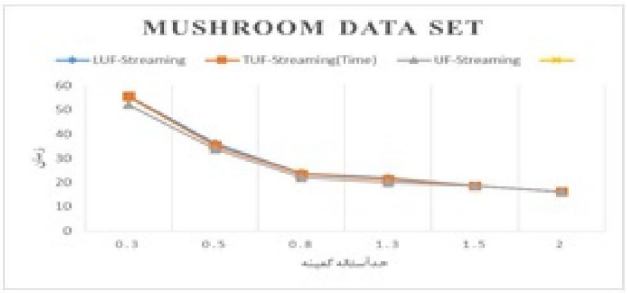


Figure 8 shows the evaluation threshold of the datasets Mushroom (Experiment 5)

Conclusion:

The results of our experiments on the data show that the most effective method of additive and non-AD Miner algorithm in the discovery of functional dependencies is more efficient. Another advantage of this method over other methods shows a dependency tuples that do not comply.

This feature can be used to detect inconsistent data in a data set used. First, our evaluation focused on four algorithms that we have presented, the evaluation shows that among the four algorithms presented LUF-streaming algorithm that uses algorithms Landmark windows TUF-streaming (time) the window uses has the best performance.

Then these two algorithms have the best performance, the UF-streaming algorithms, in that it uses the slide windows have evaluations show that the presented algorithm is only slightly slower than in the UF-streaming algorithms run and if that UF-streaming algorithm that uses a slide window and it means that the removal of older data uses.

If the algorithms are presented using the techniques of a window, that window size increases the increases. This means that more information is available online from the data stream mining and users using older data and applies the data to their strategies.

References.

1. Mundher, M. Muhamad, D. Rehman, A. Saba, T. Kausar, F. (2014) Digital watermarking for images security using discrete slantlet transform, Applied Mathematics and Information Sciences Vol 8(6), PP: 2823-2830, doi.10.12785/amis/080618.
2. Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. SIGMOD Rec., 29:1{12, May 2000.
3. Rehman, A. and Saba, T. (2014). Evaluation of artificial intelligent techniques to secure information in enterprises, Artificial Intelligence Review, vol. 42(4), pp. 1029-1044, doi. 10.1007/s10462-012-9372-9.
4. Norouzi, A. Rahim, MSM, Altameem, A. Saba, T. Rada, A.E. Rehman, A. and Uddin, M. (2014) Medical image segmentation methods, algorithms, and applications IETE Technical Review, Vol.31(3), doi. 10.1080/02564602.2014.906861.

5. Carson Kai-Sang Leung, Christopher L. Carmichael, and Boyu Hao. Efficient mining of frequent patterns from uncertain data. In Proceedings of the Seventh IEEE International Conference on Data Mining Workshops (ICDM 2007), Washington, DC, USA, pages 489{494. IEEE Computer Society, 2007.
6. Chun-Kit Chui, Ben Kao, and Edward Hung. Mining frequent itemsets from uncertain data. In Proceedings of the 11th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD 2007), Nanjing, China, pages 47{58. Springer-Verlag, 2007.
7. Carson Kai-Sang Leung, Mark Anthony F. Mateo, and Dale A. Brajczuk. A tree-based approach for frequent pattern mining from uncertain data. In Proceedings of the 12th Paci_c-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD 2008), Osaka, Japan, pages 653{661. Springer-Verlag, 2008.
8. A. Selamat, C. Phetchanchai, T. Saba, A. Rehman (2010). Index financial time series based on zigzag-perceptually important points, Journal of Computer Science, vol. 6(12), 1389-1395, doi. 10.3844/jcssp.2010.1389.1395.
9. Xiangyuan Dai, Man Lung Yiu, Nikos Mamoulis, and Michail Vaitis. Probabilistic spatial queries on existentially uncertain data. In Proceedings of the 9th International Symposium on Spatial and Temporal Databases (SSTD 2005), Angra dos Reis, Brazil, pages 400{417.Springer-Verlag, 2005.
10. Saba, T. and Altameem, A. (2013) Analysis of vision based systems to detect real time goal events in soccer videos, Applied Artificial Intelligence, vol. 27(7), pp. 656-667, doi. 10.1080/08839514.2013.787779.
11. Nan Jiang and Le Gruenwald. Research issues in data stream association rule mining. SIGMOD Rec., 35:14{19, March 2006.
12. A. Rehman and T. Saba (2012). Off-line cursive script recognition: current advances, comparisons and remaining problems, Artificial Intelligence Review, Vol. 37(4), pp.261-268. doi. 10.1007/s10462-011-9229-7.
13. Mahmood Deypir, Mohammad Hadi Sadreddini, An Efficient Algorithm for Mining Frequent Itemsets Within Large Windows Over Data Streams, International Journal of Data Engineering (IJDE), Volume (2) : Issue (3) : 2011.
14. Li H, Lee S, Shan M. “An efficient algorithm for mining frequent itemsets over the entire history of data streams”. Proceedings of the first international workshop on konwledge discovery in data streams, Pisa, Italy, 2004.
15. Carson Kai-Sang Leung and Fan Jiang, Frequent Pattern Mining from Time-Fading Streams of Uncertain Data, LNCS 6862, pp. 252–264, 2011.
16. Mannila, H. and R¨aih¨a, K.-J. (1991).The design of relational databases. Addison-Wesley.
17. Brockhausen, P. (1994). Discovery of functional and unary inclusion dependencies in relational databases. Master’s thesis, University Dortmund, Informatik VIII. In German.
18. Anforderungen An, Lehrstuhl Viii, Lehrstuhl Viii, Siegfried Bell, Siegfried Bell Peter Brockhausen, Peter Brockhausen, Fachbereich Informatik, Fachbereich Informatik, Fachbereich Informatik, Fachbereich Informatik, Fachbereich Informatik, Discovery of Data Dependencies in Relational Databases, Machine Learning: ECML-95: 8th European Conference on Machine Learning, Vol. 8, 1995.
19. P. Bosc, L. Lietard, and O. Pivert, Functional dependencies revisited under graduality and imprecision, pp. 57 - 62, NAFIPS, 1997.
20. F. Berzal, I.Blanco, D. Sanchez, J.M. Serrano and M.A. Vila, A definition for fuzzy approximate dependencies, Fuzzy Sets and Systems, Vol. 149, No. 2, pp. 105 – 129, 2005.
21. J. Kivinen and H. Mannila, Approximate inference of functional dependencies from relations, Theoretical Computer Science, Vol. 149, No. 1, pp. 129 – 149, 1995.
22. J.M. Morrissey, Imprecise information and uncertainty in information systems, ACM Transactions on Information Systems, Vol.8, No. 2, pp. 159–180, 1990.
23. U. Nambiar, and S. Kambhampati, Answering Imprecise Queries over Autonomous Web Databases, In: Proc. ICDE 2006, 22nd International Conference on Data Engineering, 2006.
24. S.L. Wang, J.W. Shen, and T.P. Hong, Discovering functional dependencies Incrementally from Fuzzy Relational Databases, in: Proc. English National Conference on Fuzzy Theory and Its Applications, pp. 17 – 24, 2000.
25. S.L. Wang, J.S. Tsai, B.C. Chang, Mining Approximate Dependencies using partitions on Similarity-Relation based Fuzzy databases, in: Proc. IEEE SMC'99, Vol. 6, PP. 871_875, 1999.
26. S.L. Wang, J.S. Tsai and T.P. Hong, Discovering functional dependencies from Similarity-based Fuzzy Relational Databases, Intelligent Data Analysis, Vol. 5, No. 2, pp. 131 – 149, 2001.
27. P. A. Flach and I. Savnik, Database dependency discovery: a machine learning approach, AI communications, Vol. 12, No. 3, pp. 139 – 160, 1999.
28. H. Mannila and K.J. Raiha. Algorithms for inferring functional dependencies from relations. DKE, Vol. 12, No. 1, pp. 83-99, 1994.
29. S. Lopes, J.M. Petit, L. Lakhal, Efficient Discovery of Functional Dependencies and Armstrong Relations, in: Proc. ICDT 2000, the 7th International Conference on Extending Database Technology: Advances in Database Technology, Vol. 1777, pp. 350 – 364, 2000.

30. C. Wyss, C. Giannella, and E. Robertson, FastFDs, A Heuristic-Driven Depth-First Algorithm for Mining Functional Dependencies from Relation Instances, Data Warehousing and Knowledge Discovery, Vol. 2114, No. 1, pp. 101 – 123, 2001.
31. Y. Huhtala, J. Karkkainen, P. Porkka and H. Toivonen, TANE: An Efficient Algorithm for Discovering Functional and Approximate Dependencies. The Computer Journal. Vol. 42, No. 2, pp. 100 – 111, 1999.
32. Xuan Hong Dang, Kok-Leong Ong, and Vincent Lee, An Adaptive Algorithm for Finding Frequent Sets in Landmark Windows, Springer-Verlag Berlin Heidelberg, pp. 590–597, 2012.
33. S.M. Fakhr Ahmad, M. Zolghadri Jahromi, M.H. Sadreddini, A new incremental method for discovery of minimal approximate dependencies using logical operations, Journal of Intelligent Data Analysis, Volume 12 pages.607-619, 2008.
34. Carson Kai-Sang Leung. Mining uncertain data. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1(4):316,329, 2011.
35. T Saba, A Rehman, M Elarbi-Boudihir (2014). Methods And Strategies On Off-Line Cursive Touched Characters Segmentation: A Directional Review, Artificial Intelligence Review vol. 42 (4), pp. 1047-1066. doi 10.1007/s10462-011-9271-5.
36. A. Rehman and T. Saba (2011). Document skew estimation and correction: analysis of techniques, common problems and possible solutions Applied Artificial Intelligence, Vol. 25(9), pp. 769-787. doi. 10.1080/08839514.2011.607009
37. T. Saba, A. Rehman and G. Sulong (2011) Improved statistical features for cursive character recognition" International Journal of Innovative Computing, Information and Control (IJICIC) vol. 7(9), pp. 5211-5224.
38. Z. F. Muhsin; A. Rehman; A. Altameem; T. Saba; M. Uddin (2014). Improved quadtree image segmentation approach to region information. the imaging science journal, vol. 62(1), pp. 56-62, doi. http://dx.doi.org/10.1179/1743131X13Y.0000000063.
39. Neamah, K. Mohamad, D. Saba, T. Rehman, A. (2014). Discriminative features mining for offline handwritten signature verification, 3D Research vol. 5(3), doi. 10.1007/s13319-013-0002-3
40. Saba T, Al-Zahrani S, Rehman A. (2012) Expert system for offline clinical guidelines and treatment Life Sci Journal, vol.9(4), pp. 2639-2658.
41. Rehman, A. Alqahtani, S. Altameem, A. Saba, T. (2014) Virtual machine security challenges: case studies, International Journal of Machine Learning and Cybernetics vol. 5(5), pp. 729-742, doi. 10.1007/s13042-013-0166-4.
42. K. Meethongjan, M. Dzulkifli, A. Rehman, A. Altameem, T. Saba (2013) An intelligent fused approach for face recognition, Journal of Intelligent Systems, vol. 22(2), pp. 197-212, doi. 10.1515/jisys-2013-0010,
43. S.Joudaki, D. Mohamad, T. Saba, A. Rehman, M. Al-Rodhaan, A. Al-Dhelaan (2014) Vision-Based Sign Language Classification: A Directional Review, IETE Technical Review, Vol.31 (5), 383-391, doi. 10.1080/02564602.2014.961576.
44. T. Saba, A. Rehman, A. Altameem, M. Uddin (2014) Annotated comparisons of proposed preprocessing techniques for script recognition, Neural Computing and Applications Vol. 25(6), pp. 1337-1347 , doi. 10.1007/s00521-014-1618-9.
45. JWJ Lung, MSH Salam, A Rehman, MSM Rahim, T Saba (2014) Fuzzy phoneme classification using multi-speaker vocal tract length normalization, IETE Technical Review, vol. 31 (2), pp. 128-136, doi. 10.1080/02564602.2014.892669.
46. T Saba, A Rehman, A Al-Dhelaan, M Al-Rodhaan (2014) Evaluation of current documents image denoising techniques: a comparative study , Applied Artificial Intelligence, vol. 28(9), pp. 879-887, doi. 10.1080/08839514.2014.954344.
47. A Nodehi, G Sulong, M Al-Rodhaan, A Al-Dhelaan, A Rehman, T Saba (2014) Intelligent fuzzy approach for fast fractal image compression, EURASIP Journal on Advances in Signal Processing, doi. 10.1186/1687-6180-2014-112.
48. MSM Rahim, SAM Isa, A Rehman, T Saba (2013) Evaluation of Adaptive Subdivision Method on Mobile Device 3D Research 4 (2), pp. 1-10.
49. AM Ahmad, G Sulong, A Rehman, MH Alkawaz, T Saba (2014) Data Hiding Based on Improved Exploiting Modification Direction Method and Huffman Coding, Journal of Intelligent Systems, vol. 23 (4), pp. 451-459, doi. 10.1515/jisys-2014-0007.
50. M Harouni, MSM Rahim, M Al-Rodhaan, T Saba, A Rehman, A Al-Dhelaan (2014) Online Persian/Arabic script classification without contextual information, The Imaging Science Journal, vol. 62(8), pp. 437-448, doi. 10.1179/1743131X14Y.0000000083.
51. T. Saba, A.Rehman, S. Al-Zahrani (2013) Character Segmentation in Overlapped Script using Benchmark Database, pp. 140-143, ISBN: 978-1-61804-233-0
52. Uddin, M. Shah, A. Rehman, A. (2014) Metrics for Computing Performance of Data Center for Instigating Energy Efficient Data Centers, Journal of Scientific and Industrial Research, vol. 73 (1), 11-15.
53. MSM Rahim, A Rehman, R Kumoi, N Abdullah, T Saba (2012) FiLeDI framework for measuring fish length from digital images, International Journal of Physical Sciences, vol. 7 (4), pp. 607-618.

10/28/2014