REGULAR PAPER

Mining discriminative itemsets in data streams using the tilted-time window model

Majid Seyfi1 · Richi Nayak1 · Yue Xu1 · Shlomo Geva1

Received: 24 April 2018 / Revised: 21 January 2021 / Accepted: 24 January 2021 / Published online: 15 February 2021 © The Author(s), under exclusive licence to Springer-Verlag London Ltd. part of Springer Nature 2021

Abstract

A discriminative itemset is a frequent itemset in the target data stream with much higher frequency than that of the same itemset in the rest of the data streams in the dataset. The discriminative itemsets describe the distinguishing features between data streams. Mining discriminative itemsets in data streams is very important, where continuously arriving transactions can be inserted in fast speed and large volume. Compared with frequent itemset mining in single data stream, there are additional challenges in the discriminative itemset mining process as the Apriori property of subset is not applicable. We propose an efficient and high accurate method for mining discriminative itemsets in data streams using a tilted-time window model. The proposed single-pass H-DISSparse algorithm is designed particularly based on several well-defined characteristics aiming to improve the approximate frequencies of the itemsets in the tilted-time window model. The data structures are dynamically adjusted in offline time intervals to reflect the discriminative itemset frequencies in different time periods in unsynchronized data streams. Empirical analysis shows the efficient time and space complexity of the proposed method in the fast-growing big data streams.

Keywords Data stream mining · Discriminative itemsets · Prefix tree · Tilted-time window model

Majid Seyfi majid.seyfi@gmail.com
Richi Nayak r.nayak@qut.edu.au
Yue Xu yue.xu@qut.edu.au
Shlomo Geva s.geva@qut.edu.au

1 Data Science Discipline, Science and Engineering Faculty, Queensland University of Technology, Brisbane, QLD, Australia
1 Introduction

A data stream can be defined as an uninterrupted flow of transactions that transmit with fast speed over a period of time [25]. The last decade has witnessed the rise of frequent pattern mining methods for extracting useful information from a single data stream [1, 24]. The discriminative itemset mining from multiple data streams is another emerging research area [4, 11, 20, 21, 23, 27–30]. The discriminative itemsets in the tilted-time window model are defined as the frequent itemsets in the target data stream that have frequencies much higher than that of the same itemsets in other data streams in different time periods. Without loss of generality, we call other data streams a ‘general data stream’ for the sake of simplicity. The discriminative itemsets are relatively frequent in the target data stream and relatively infrequent in the general data stream in each time period during the history. An essential issue is to find the itemsets that can distinguish the target data stream from all other data streams in each time period.

The patterns in data streams are usually time sensitive, and in many applications, users are more interested in changes in patterns and their trends during the time than the patterns themselves [16]. The patterns appearing in the old time may not be dominant anymore and may have lost their attractions (e.g., patterns in the news delivery services). Particular groups of patterns appearing in a period of time should not affect the general trend of the patterns in data streams during the history or recent time (e.g., patterns related to the specific events). Time-related patterns and the changes in their trends during the history of data streams are of interest for recent patterns in the short time intervals and the old patterns in the larger time intervals. These patterns are represented in the tilted-time windows for answering the time sensitive queries. The tilted window model is made of multiple windows, which are in different sizes, and each one points to the specific period of time.

The landmark, damped and sliding window are the other types of window models. The landmark window model is from a special point in data stream as the start point until now [25]. The damped window model follows the landmark window model using the specified start point. However, a weighting functions are used for the incoming transactions by giving higher weight to the newly added transactions. These groups of algorithms usually use decaying factors for the old transactions [7]. The sliding window model has a dynamic start point moving by time. This window model has static size, which is defined either based on the number of transactions or a time slide [9].

There are many real-world scenarios that can show the significance of mining discriminative itemsets in data streams using the tilted-time window model. Monitoring the market basket transactions could have started a long time ago, and treating the old and new transactions in a same way equally may not be useful for tracing market fluctuation and guiding the business (e.g., as in the landmark window model [25]). The fading of old transactions (e.g., reducing the weight as in the damped window model [7]) may not be enough for those applications interested in finding the changes in the discriminative itemsets and their trends during the time. Discriminative itemsets in the data streams made of market basket dataset represented in a tilted-time window model are useful for identifying the specific set of items that are of high interest in one market compared to the other markets in different time periods. The target and general data streams are generated by the same sources (e.g., organizations) and distinguish based on discriminative itemsets. Discriminative itemsets are applicable for showing the relative changes in the data stream trends in different time periods and for answering the time-sensitive queries (e.g., in applications of personalization, anomaly detection and prediction). One interesting scenario in market basket is how
people in different suburbs feed their kids by differentiating based on the discriminative itemsets in the tilted-time window model.

Web page personalization can be optimized by changes in user preferences in different time periods or during specific events. User groups may have different preferences during the time by visiting the groups of the web pages much more frequently than other user groups in different time periods. The sequences of queries with higher support in one geographical area compared to another area are time related. The discriminative sequences of queries related to the specific events and changes in the relative trends are monitored separately in different time periods for better recommendation. Changes in the discriminative pattern trends during network monitoring in the last few minutes are more valuable for anomaly detection and network interference prediction than the discriminative patterns themselves. The discriminative itemsets are important in the classification and recommendation.

**FP-Stream** is a method proposed for mining frequent patterns from data streams using the tilted-time window model [16]. Frequent pattern mining in data streams has more challenges compared to the static datasets as the infrequent patterns can become frequent later and cannot be ignored. The data structures need to be adjusted regularly by pattern frequencies over times. The discriminative itemsets are a small subset of the frequent itemsets which were defined as a kind of contrast patterns [11]. The DISSparse algorithm is proposed in [30] as an efficient method for mining discriminative itemsets in one batch of transactions. There are also couple of methods proposed for mining discriminative items in data streams [23, 27]. The emerging patterns (EPs) [12] are the well-known contrasting patterns whose frequencies grow significantly higher in one dataset in comparison with another one.

Although the discriminative itemsets have similar definition to some types of emerging patterns [11, 34], there are fundamental differences. The discriminative itemsets are defined as single itemset with explicit supports in datasets, rather than group of patterns between two itemsets as border representatives in emerging patterns [12, 34]. The discriminative itemsets are useful in applications need single itemset and not group of itemsets between borders. For example, in rule mining based on support and confidence of each itemset, or itemset history in data stream mining. Every discriminative itemset must be considered separately with its supports in data streams. The group of itemsets between two borders may not be all discriminative in the new incoming batch of data streams.

As a research gap, there are differences compared to the emerging patterns. Firstly, the discriminative itemset is presented as single itemset with explicit supports in datasets. Each discriminative itemset is saved in the history based on its different frequencies. These discriminative itemsets may be discriminative in data streams by merging with their old frequencies or be ignored as non-discriminative itemsets. The emerging patterns are mainly discovered for one batch of transactions. Secondly, the discriminative itemsets are presented with different discriminative values (i.e., discrimination between support in target dataset vs support in general dataset). The highly and lowly discriminative itemsets are explicitly reported with their exact supports on the datasets. The itemsets with different discriminative values are clearly different from emerging patterns, which are discovered between two borders.

To the best of our knowledge, this is the first study in mining discriminative itemsets in data streams using the tilted-time window model. There are technical novelties in our proposed method compared to the existing methods for mining frequent itemsets in data streams [1, 24]. We apply the efficient DISSparse method [30] for offline updating the tilted-time window model. The DISSparse does not follow the Apriori property, which
is mainly used in mining frequent itemsets in data streams. We also define three corollaries for mining high accurate discriminative itemsets in the tilted-time window model. The DISSparse method [30] proposed for mining discriminative itemsets in static datasets. We apply this method with characteristics of the tilted-time window model for mining discriminative itemsets efficiently and with high accuracy in data streams.

The data streams are processed in one scan, and the tilted-time window model is updated in an offline state in different time periods. The discriminative itemsets are reported with exact support and exact ratio in one batch of transactions using the proposed method in [30]. The discriminative itemsets may happen between the borders of the tilted-time window frames. The relaxation ratio $\alpha$ is defined for saving the itemsets with approximate frequencies. The discriminative itemsets are reported with approximate support and approximate ratio between borders of the tilted-time window model. The multiple scans are not acceptable in data stream mining [1, 17], and the discriminative itemsets with exact minimum support and exact frequency ratio in data streams cannot be discovered. The approximation can become worse in the larger time periods by merging the approximate discriminative itemsets.

The number of false-positive and false-negative discriminative itemsets must be bounded for qualified answers. The discriminative itemsets do not follow the Apriori property defined for the frequent itemsets and a subset of discriminative itemsets can be non-discriminative. The tail pruning techniques proposed in FP-Stream [16] for efficient frequent itemset mining using the tilted-time window model by minimum support guarantee are not applicable to discriminative itemset mining using the tilted-time window model. The discriminative itemsets are a sparse subset of frequent itemsets [30]. Based on the properties of the discriminative itemsets in the tilted-time window model, three corollaries are defined. This guarantees the highest refined approximate support and approximates the ratio bound in the tilted-time window model.

In this paper, the problem of mining discriminative itemsets in data streams using the tilted-time window model is formally defined. The advanced high-efficient and high-accurate method called H-DISSparse utilizes the DISSparse algorithm [30] with the tilted-time window model. The proposed method is explained in detail with its novel data structures and offline updating of the tilted-time window model. In order to achieve the best approximation in mining discriminative itemsets in data streams, we use the properties of the discriminative itemsets in the tilted-time window model to propose a novel and efficient method. The proposed method guarantees the approximate support and approximate ratio bound in discriminative itemsets in the large and fast-growing data streams with the necessity of concise process fitting in real-world applications. The proposed method is extensively evaluated on data streams made of multiple batches of transactions exhibiting diverse characteristics and by setting different thresholds. Empirical analysis shows efficient time and space complexity with the highest refined approximate bound in discriminative itemsets gained by the H-DISSparse algorithm.

More specifically, the following contributions are made in this paper:

- Developing the efficient single-pass H-DISSparse algorithm for mining discriminative itemsets in data streams using the tilted-time window model.
- Introducing novel in-memory data structures for efficiently offline updating the tilted-time window model;
- Defining three corollaries for achieving the highest approximate support and approximate ratio in discriminative itemsets in the tilted-time window model;
• Evaluating the proposed algorithm in a wide range of datasets with different parameter settings;
• Showing strategies and principles for tuning parameters based on the application domains and dataset characteristics;

The rest of the paper is organized as follows: In Sect. 2, the existing works are presented. Section 3 defines the problem using formal notations, and the H-DISSparse algorithm with the tilted-time window model and its updating is proposed in detail. The experimental results are reported in Sect. 4. Section 5 includes the conclusion and future works.

2 Related works

There are many existing studies in contrast mining focus on different types of patterns [11]. Discriminative itemset mining problem in data streams addressed here focuses on the relative differences of supports in data streams explicitly. A related area to this research is emerging patterns (EPs) [12] as itemsets whose frequencies grow significantly higher in one dataset in comparison with another one. EPs are identified by extracting the maximal itemsets separately for each dataset using a defined minimum threshold. The left and right borders are then defined based on the lowest and highest thresholds, and a group of maximal itemsets are reported between the two borders [12]. In the proposed method in [12], which has been followed in [3, 5, 6, 13, 32–34] the degree of change in supports of itemsets is important, and the actual support of itemsets is not considered. Also, EPs are generally defined for static datasets.

Authors in Alhammady and Ramamohanarao [3] have attempted EPs mining in data streams based on the same idea of border definition. This method showed the EPs related to each block of transactions and discard the block from the process. Bailey and Loekito’s [5] method is proposed for mining contrast patterns in changing data based on the old and the current parts of a data stream. The method is focused on jumping emerging patterns as special type of contrast patterns. The minimal JEPs are discovered in data stream by adding new transactions and deleting the old transactions. This is different to the problem focused in this paper as the contrast patterns are discovered in the old part (i.e., old class) compared to the recent part (i.e., recent class) of a single data stream. The discriminative itemsets proposed in this paper are discovered in data streams changing at a same time. The emerging pattern mining with streaming feature selection [32] dynamically selects and maintains the effective features from feature stream.

The δ-discriminative emerging patterns [22] are determined based on a threshold δ. The DPMiner algorithm [22] can efficiently mine the δ-discriminative emerging patterns by skipping the subset of itemsets if their support in the general dataset is larger than δ. However, for the discriminative itemsets proposed in this paper, a subset of a non-discriminative itemsets can be discriminative and we cannot set a limit for the itemset frequency in the general dataset. The delta-discriminative emerging pattern is ignored as redundant if it is the superset of another delta-discriminative emerging pattern. The CDPM method (Conditional Discriminative Patterns Mining) [19] is also proposed for discovering a set of significant non-redundant discriminative patterns, which have no similar discrimination from their subsets. The DPM [22] and CDPM [19] discover the discriminative itemsets based on their statistical measures. The discriminative itemsets proposed in this paper are discovered with their explicit relative supports in data streams and are not redundant.
There are algorithmic differences between the proposed $H$-DISSparse method and the emerging pattern mining methods. The discriminative itemsets are small subset of frequent itemsets, and the proposed $H$-DISSparse method is based on fundamental of the FP-Growth method [18]. Compared to $H$-DISSparse method, emerging pattern mining methods are more efficient in the datasets with inherent discrimination (e.g., mushroom datasets with edible and poisonous classes, or chess datasets with win and lose classes). These datasets usually have one accepted class and one rejected class. In these datasets, mainly few attributes make permanent discrimination in the itemsets not considering the rest of the items involved in the itemsets. This is easy to represent between the borders as in emerging pattern mining methods [12, 22, 34]. In datasets without inherent discrimination (e.g., market basket datasets in different suburbs, or accident datasets in different regions) the data classes have similar credits as each other. When there is no inherent discrimination in the datasets, the number of borders between emerging patterns will become almost equal to the number of discriminative itemsets which is not efficient. In these datasets, the discriminative itemsets are random and sparse and must be discovered separately with explicit supports.

$FP$-Stream is a famous method proposed for mining frequent patterns from data streams [16]. The $FP$-Stream data structure is made of a frequent pattern tree for maintaining the frequent and sub-frequent patterns itemsets and a built-in tilted-time window model for each individual pattern. The time-sensitive frequent patterns with approximate support guarantee are maintained at multiple time granularities. Mining the frequent patterns over different time intervals in a data stream assumes that the transactions can be scanned in a limited size window at any moment. The pattern fragment method [18] is used for mining the frequent patterns in the current window frames. However, pattern fragment is not applied to the problem of discriminative itemset mining. The Apriori property defined for frequent itemsets is not applicable as the subset of discriminative itemsets can be non-discriminative. Due to the need of processing multiple data streams, a discriminative itemsets mining algorithm has to deal with the combinatorial explosion of itemsets generated from multiple streams. Also, the two types of pruning techniques proposed in Giannella et al. [16] cannot be used for mining discriminative itemsets from data streams using the tilted-time window model.

The discriminative item mining methods were proposed [23, 27] and do not have combinatorial explosion. The hybrid method [23] of discriminative item mining counts the frequent items in each stream. Items are assigned to different expandable buckets, and all the items in one bucket are counted together using the same counter. This method saves the summary of each bucket to discover the concealed discriminative items in non-discriminative buckets. The hierarchical counters method [27] of discriminative items mining utilizes different thresholds at different times by using a counter structure for identifying frequencies of all items. Discriminative itemset mining using these methods would be time- and space-consuming because of the explosion in the number of itemset combinations.

On the contrary, the proposed $H$-DISSparse method in this paper does not require the prior frequency calculation of all generated itemset combinations. It effectively utilizes a prefix tree structure for holding discriminative itemsets in the tilted-time window model. The logarithmic tilted-time windows model [16] is used to display the recent discriminative itemsets in fine granularities and the historical ones in coarse granularities. The $H$-DISSparse method uses several well-defined characteristics to improve the approximate supports. The empirical analysis of the proposed method reveals that it is able to produce complete set of discriminative itemsets in the current batch of transactions. We introduce sub-discriminative level to control the precision and recall of these itemsets in the tilted-time window model. Sub-discriminative
itemsets in the current timeframe are allowed to merge with previous frames, hoping they will collectively become discriminative itemsets.

3 Problem statement and algorithm

In this section, first we formulate the problem of mining discriminative itemsets in data streams using the tilted-time window model, and then, we propose the algorithm with the tilted-time window model.

3.1 Formal definition

Let $\sum$ be the alphabet set of items, a transaction $T = \{e_1, \ldots, e_i, e_{i+1}, \ldots, e_n\}$, $e_i \in \sum$, is defined as a set of items in $\sum$. The items in the transaction are in the alphabetical order by default for ease in describing the mining algorithm. The two data streams $S_i$ and $S_j$ are defined as the target and general data streams; each consists of a different number of transactions, i.e., $n_i$ and $n_j$, respectively. A group of input transactions from two data streams $S_i$ and $S_j$ in the pre-defined time period are set as a batch of transactions $B_n$, i.e., $n \geq 1$.

The tilted-time window model is composed of different window frames denoted as $W_k$, i.e., $k \geq 0$ as in Fig. 1. Each window frame $W_k$ refers to a different time period containing itemsets made of transactions from different number of batches in two data streams $S_i$ and $S_j$ with the lengths of $n^k_i$ and $n^k_j$, respectively. The current window frame is denoted as $W_0$.

An itemset $I$ is defined as a subset of $\sum$. The itemset frequency is the number of transactions that contains the itemset. The frequency of itemset $I$ in data stream $S_i$ in the window frame $W_k$ is denoted as $f^k_i(I)$ and the frequency ratio of itemset $I$ in data stream $S_i$ in the window frame $W_k$ is defined as $r^k_i(I) = f^k_i(I)/n^k_i$.

In this paper, if the frequency ratio of itemset $I$ in the target data stream $S_i$ in the window frame $W_k$ is larger than the frequency ratio in the general data stream $S_j$, i.e., $r^k_i(I)/r^k_j(I) > 1$, then the itemset $I$ can be considered as a discriminative itemset in the window frame $W_k$. Let $R^k_{ij}(I)$ be the ratio between $r^k_i(I)$ and $r^k_j(I)$, i.e., $R^k_{ij}(I) = r^k_i(I)/r^k_j(I)$. Obviously, the higher the $R^k_{ij}(I)$, the more discriminative the itemset $I$ is.

To more accurately define discriminative itemsets, we introduce a user-defined threshold $\theta > 1$, called a discriminative level threshold with no upper bound. An itemset $I$ is considered discriminative in the window frame $W_k$ if $R^k_{ij}(I) \geq \theta$. This is formally defined as:

$$R^k_{ij}(I) = \frac{r^k_i(I)}{r^k_j(I)} = \frac{f^k_i(I)n^k_j}{f^k_j(I)n^k_i} \geq \theta \quad (1)$$

The $R^k_{ij}(I)$ could be very large but with very low $f^k_j(I)$. In order to accurately identify discriminative itemsets that have reasonable frequency in the window frame $W_k$, and also in the case of $f^k_j(I) = 0$, we introduce another user-specified support threshold, $0 < \varphi < 1/\theta$, to

| $W_k$ | … | $W_3$ | $W_2$ | $W_1$ | $W_0$ |
|-------|----|-------|-------|-------|-------|

![Fig. 1 Tilted-time window frames](image)
eliminate itemsets that have very low frequency in the window frame $W_k$. In this paper, an itemset $I$ is considered as discriminative if its frequency in the window frame $W_k$ becomes greater than $\varphi \theta n_i$, i.e., $f^k_i(I) \geq \varphi \theta n_i$ and $R^k_{ij}(I) \geq \theta$.

Definition 1 Discriminative itemsets in the tilted-time window model: Let $S_i$ and $S_j$ be two data streams, with the current size of $n^k_i$ and $n^k_j$ in a window frame $W_k$, i.e., $k \geq 0$ that contain varied length transactions of items in $\Sigma$, a user-defined discriminative level threshold $\theta > 1$ and a support threshold $\varphi \in (0, 1/\theta)$. The set of discriminative itemsets in $S_i$ against $S_j$ in the tilted-time window model in the window frames $W_k$, denoted as $DI^k_{ij}$, i.e., $k \geq 0$, is formally defined as:

$$DI^k_{ij} = \left\{ I \subseteq \sum_i \mid f^k_i(I) \geq \varphi \theta n^k_i \& R^k_{ij}(I) \geq \theta \right\}$$ (2)

The itemsets that are not discriminative in the current window frame $W_0$ can be discriminative in some larger window frames in the tilted-time window model (e.g., by merging the multiple window frames). In order to avoid missing potential discriminative itemsets in larger window frames, we propose to identify sub-discriminative itemsets in the tilted-time window model with a parameter specified by the user. The sub-discriminative itemsets are discovered by relaxation of $\alpha \in (0, 1)$ with more number sub-discriminative itemsets in smaller $\alpha$. The $\alpha$ is defined for approximate support and approximate ratio between borders of the tilted-time window model. The itemsets $I$ is sub-discriminative if it is not discriminative, but its frequency in target data stream $S_i$ is not less than $\alpha \varphi \theta n_i$ and its ratio is not less than $\alpha \theta$. The discriminative itemsets are in interest; however, the sub-discriminative itemsets also kept tracking during the process as they may be discriminative in larger window frames.

The logarithmic tilted-time window model is a compact data structure; for example, a batch of transactions in one minute is supposed as the smallest time period. The current window frame shows the discriminative itemsets in the last minute, and it is followed by the results in the remaining slots of the next 2 min, 4 min, 8 min, etc.

Example 1 The $H$-DISStream is graphically monitored using the running example presented in Table 1, which contains two simple datasets with the same number of transactions in $S_1$ and $S_2$ ($n_1 = n_2 = 15$). In this example, the discriminative level threshold is set to $\theta = 2$ and the support threshold is set to $\varphi = 0.1$.

The DISTree method [29] is proposed and tested based on single scan, but the size of the data structures and the processing time is highly affected. Data stream mining algorithms [10, 16, 31] use two scans for making the concise data structures and faster processing time in which items are ordered by decreasing frequencies as in Han et al. [18]. This ordering is adjusted based on the frequent items in the first batch of transactions and remains fixed for all remaining batches in the data streams. In this paper, we use two scans. In the first scan, the frequent items in the target data stream $S_1$ are found and sorted based on the descending order

| S/T | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|-----|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| $S_1$ | abcd | abcd | abcd | abc | ab | ace | bce | bc | bc | bde | bd | cde | cde | cde | Ce |
| $S_2$ | abcd | abcd | ac | ac | ac | ac | a | a | bcd | cde | cde | cde | cd | c | c |
of their frequencies (i.e., Desc-Flist order in Table 2 is constructed out of frequent items in Table 1). The Desc-Flist is used as the default order for all the prefix tree structures and also shows the processing order in the Header-Table (e.g., the Header-Table in Fig. 2 is processed from the least frequent item in the dataset, item a). The frequent items in each input transaction in the datasets are sorted based on the Desc-Flist order before adding to the prefix tree structure (e.g., the H-DISSStream in Fig. 2 is made of transactions with Desc-Flist order in their items).

The H-DISSStream is a prefix tree structure with the built-in tilted-time window model as defined below.

**H-DISSStream:** The H-DISSStream prefix tree structure is for holding the discovered discriminative and sub-discriminative itemsets as well. Both discriminative and sub-discriminative itemsets share the branches in the same H-DISSStream structure for their most common frequent items (e.g., as in Fig. 2). Each path in the H-DISSStream may represent a subset of multiple discriminative and sub-discriminative itemsets started from the root of the prefix tree structure. Each node in the H-DISSStream has two counters \(f_i\) and \(f_j\) for holding the frequencies of an itemset in the target data stream \(S_i\) and the general data stream \(S_j\), respectively, in the current window frame \(W_0\). The Header-Table is defined for fast traversing the prefix tree structure using the links holding the itemsets ending with identical items and the nodes are tagged as discriminative, sub-discriminative or non-discriminative (i.e., subset of discriminative or sub-discriminative itemsets). Each H-DISSStream node may have a built-in tilted-time window frame if the itemset is discriminative in the larger window frame \(W_k\) (i.e., \(k > 0\), sub-discriminative in the history summary of the window frames (i.e., \(W_{0..m}\)) or appeared as a non-discriminative subset of discriminative or sub-discriminative itemsets in any window frame \(W_k\) (i.e., \(0 < k \leq m\)).

### Table 2: Desc-Flist order of frequent items is target data stream \(S_1\)

| Item/order | \(a\) | \(b\) | \(c\) | \(d\) | \(e\) |
|------------|------|------|------|------|------|
| Frequency  | 6    | 10   | 12   | 8    | 7    |
| Order      | 4    | 1    | 0    | 2    | 3    |

Fig. 2 A sample H-DISSStream based on example with the built-in tilted-time window model
For the sake of clarity in this paper, the non-discriminative subset of discriminative or sub-discriminative itemsets in any window frame $W_k$ is called a non-discriminative subset; for example, in Fig. 2 $c_{12,13}$ is the non-discriminative itemset which is the subset of other discriminative itemsets in $W_0$, by considering the discriminative level $\theta = 2$, the support threshold $\varphi = 0.1$ and dataset length in two data streams $n_1^0 = n_2^0 = 15$. The $H$-DISStream is updated by its tilted-time window model in offline time intervals after processing a new batch of transactions in the pre-defined time intervals; for example, in Fig. 2 the discriminative itemset $cba_{4,2}$ is discovered in the current window frame $W_0$. The itemset $cba$ has a built-in table with four entries related to the older window frame $W_k$ i.e., $k > 0$.

The construction of the $H$-DISStream data structure (i.e., $W_0$) is started based on the discriminative itemsets discovered out of the first batch of transactions. By processing every new batch of transactions, the discriminative itemsets are shifted and the tilted windows merged together. The pruning happens in the $H$-DISStream if the itemset is non-discriminative in the tilted-time window model and stays as leaf node.

### 3.2 Tilted-time window model updating

For self-explaining the paper, the tilted-time window model is discussed with its structure and its updating process using some examples. These are modified here for two data streams with new notations. The details about this part can be found in Giannella et al. [16].

The data streams are defined as continuous batches, each containing a different number of transactions depending on the speed of data streams. This is showed as $B_1, ... B_{n'} B_{n'+1} \ldots, B_n$ with $B_n$ as the most recent one and $B_1$ as the oldest one. For each itemset $I$, the two counters $f_x(I) < x, y >$ and $f_y(I) < x, y >$ denote the frequencies of the itemset $I$ in data streams $S_x$ and $S_y$, respectively, in the group of continuous batches $B_x$ to $B_y$, with $x \geq y$. For the sake of clarity, the itemset $I$ and data stream indicatives are omitted from the context (i.e., $f_x(I) < x, y >$ and $f_y(I) < x, y >$ denoted as $f < x, y >$). The frequencies of an itemset in the logarithmic tilted-time window model are kept during the history as follows:

$$f < n, n > : f < n - 1, n - 1 > : f < n - 2, n - 3 > : f < n - 4, n - 7 > : \ldots \tag{3}$$

For updating the tilted-time window model by shifting and merging the older window frames, the intermediate windows denoted as $[f_x(I) < x, y >]$ and $[f_y(I) < x, y >]$ are used as extra memory spaces as shown below.

The $f < n, n >$ saves the frequencies of the itemsets discovered from the current batch of transactions in $W_0$ (i.e., $f_x^0(I)$ and $f_y^0(I)$ in $H$-DISStream prefix tree structure). By processing the new batch of transactions, the $f < n, n >$ is shifted and replaces the $f < n - 1, n - 1 >$ in $W_1$ (i.e., $f_x^1(I)$ and $f_y^1(I)$) and the recent frequencies set to the $f < n, n >$. Before shifting the $f < n - 1, n - 1 >$ to the next level, it is checked to see if its intermediate window is empty, then it is shifted to that, otherwise the frequencies in the $f < n - 1, n - 1 >$ and its intermediate window are added together and shifted to the next level $f < n - 2, n - 3 >$ in $W_2$ (i.e., $f_x^2(I)$ and $f_y^2(I)$). This process will continue until shifting stops.

Following Giannella et al. [16] $B_1$ is the oldest batch, but $W_0$ is the latest window, so, $B_1$ is in $W_m$ (if $m$ is the oldest window), $B_n$ is in $W_0$. If we put the batches and the windows together in one line, the indexes of batches are decreasing from current time to old time, while the indexes of windows are increasing from current time to old time.

The processing scenario for continued batches of transactions is presented in the example below:

Springer
**Example 2** The first batch of transactions $B_1$ is processed and the discovered discriminative itemsets are set to the $H\text{-DISStream}$ considered as $f < 1,1 >$ which is the most recent window frame ($W_0$) in the tilted-time window model. By processing the next batch of transactions $B_2$, the itemsets from $H\text{-DISStream}$ are shifted to the older window frame ($W_1$) represented as $f < 1,1 >$ in the window frame table and the new discovered itemsets are set to $H\text{-DISStream}$ as $f < 2,2 >$ (i.e., $W_0$). The process continues for another batch of transactions $B_3$, and the discovered itemsets in $H\text{-DISStream}$ and its tilted-time window model set as $f < 3,3 >, f < 2,2 >$[$f < 1,1 >$]. The $[f < 1,1 >]$ is the intermediate window frame, and by a new batch of transactions it will be merged with $f < 2,2 >$ and is represented in the tilted-time window model as $f < 4,4 >; f < 3,3 >; f < 2,1 >$. The full process is represented step by step for the first 10 batches of transactions as in Fig. 3.

The sub-discriminative itemsets have potential to be discriminative by merging the window frames as presented in example below.

**Example 3** Considering $\theta=3$ and support threshold $\varphi = 0.1$, the itemset $I$ with frequencies $f^1_1(I) = 5$ and $f^2_1(I) = 2$ in window frame $W_0$ with the lengths of $n^0_1 = 10$ and $n^0_2 = 10$ is non-discriminative. The itemset $I$ with frequencies $f^1_1(I) = 5$ and $f^1_2(I) = 1$ in the window frame $W_1$ with the lengths of $n^1_1 = 15$ and $n^1_2 = 15$ is discriminative. By setting relaxation of $\alpha = 0.8$, the itemset $I$ is not omitted and discovered as discriminative itemset in the larger window frame by shifting and merging the window frames $W_0$ and $W_1$.

### 3.3 Discriminative itemsets approximate bound

The pruning techniques proposed in Giannella et al. [16] are not applicable to the problem of mining discriminative itemsets using the tilted-time window model, as explained briefly in below.

The first pruning technique in Giannella et al. [16] is related to the tail pruning in the tilted-time window model by accepting an error threshold boundary of the false-positive frequent itemsets. The tail sequences of the oldest tilted-time window frames related to the itemset are pruned if the itemset is not frequent in any of those window frames and not sub-frequent in the history of the data stream from the current time period to any of those window frames, by the defined error threshold. Based on the claim in Giannella

![Fig. 3](Tilted-time window model updating)
et al. [16], the number of false-positive answers is reasonable if the error threshold set small enough. However, setting a small error threshold is in contradiction with efficiency as the large number of sub-frequent itemsets with very low support (i.e., almost zero) has to be generated and saved in the tilted-time window model. Empirical analysis shows the number of frequent itemsets grows exponentially in smaller supports.

The discriminative itemsets are a subset of frequent itemsets, and in many applications the discriminative itemsets are in interest with low support and low ratio (e.g., in anomaly detection). Also, in the research problem of discriminative itemset mining, the itemsets in each window frame have at least two frequencies related to the target data stream and general data stream. By dropping tail sequences of the oldest tilted-time window frames, the discriminative itemsets will be missed during merging the older window frames, causing false negatives and less recall; for example, because of lack of itemsets in window frames with possible high ratio between the lengths of target data stream and general data stream (i.e., $R_k^{ij}(I) \gg 1$). This can also cause false positive and less accuracy; for example, because of lack of itemsets in window frames with possible low ratio between the lengths of the target data stream and the general data stream (i.e., $R_k^{ij}(I) \ll 1$).

The second pruning technique in Giannella et al. [16] is based on the anti-monotone Apriori property in the frequent itemsets. The superset of the frequent itemset has a frequency equal to or less than its subset, that is hold in all the window frames in the tilted-time window model. Hence, if the itemset is not frequent in the current batch, then none of its supersets need be examined. Following this if the tail of itemsets can be dropped based on the explained tail pruning techniques, then the possible existed similar tail in its all supersets can be dropped. The Apriori property is not valid for the discriminative itemset mining and an itemset can be discriminative in different window frames with non-discriminative subsets in the same window frames.

In this paper, the properties of the discriminative itemsets in the tilted-time window model are applied within efficient time and space usage for the highest refined approximate support and approximate ratio bound, by minimizing the number of false-positive and false-negative discriminative itemsets in data streams. The insight behind using these properties is to have more itemsets with exact frequencies in the different window frames, and at the same time make the window frames small enough during the history of the data streams.

The $FP-Tree$ is defined based on the basics of $FP-Growth$ [18] out of frequent items of transactions by sharing the branches for their most common frequent items. This was adapted in [29, 30] by two counters in each node for holding the frequencies of itemsets in the target dataset and general dataset. In the proposed method in this paper, the $FP-Tree$ is generated for the current batch of transactions as a similar prefix tree structure made for one batch of transactions in [29, 30], but without pruning infrequent items (i.e., $FP-Tree$ includes all items). The $FP-Tree$ includes the infrequent items in the target data stream $S_i$ in the current batch of transactions, so it can be used for mining the frequencies of the itemsets that are non-discriminative subsets of the discriminative itemsets in the older window frames. It should be noted that although the $FP-Tree$ includes all the items, each conditional $FP-Tree$ that is generated during offline batch processing follows the basics of $FP-Growth$ [18] by pruning items that are infrequent in the target data stream $S_i$. This ensures that the $FP-Tree$ including all the items does not add high complexity to the batch processing using $DISSparse$ algorithm. The discovered discriminative itemsets are saved in a pattern tree with a built-in tilted-time window model as we explain in section below.
3.3.1 Maintaining discriminative itemsets in the tilted-time window model

In the proposed $H$-DISSparse method in this paper, the itemsets that are discriminative, or appeared as non-discriminative subset (of discriminative itemsets) at least in one window frame $W_k$ (i.e., $k \geq 0$) are saved in the tilted-time window model during the history of data streams. The exact set of discriminative itemsets in the current batch of transactions is discovered using DISSparse algorithm during batch processing, respectively, and is held in the current window frame $W_0$. The exact frequencies of non-discriminative subsets in the current window frame $W_0$ are obtained by traversing the $FP$-Tree through Header-Table links for their appearances in the current batch of transactions. The tilted-time window frames are updated by shifting and merging the itemset frequencies in the larger window frames after processing each batch of transactions in the offline state.

**Property 1** Exact set of discriminative itemsets in the current batch of transactions is held in the current window frame $W_0$.

This property says that the entire discriminative itemsets in current window frame $W_0$ are saved with their exact frequencies. The discriminative itemsets in the current window frame $W_0$ are discovered using the DISSparse algorithm [30] with 100% accuracy and recall.

**Property 2** Exact set of non-discriminative subsets (of discriminative itemsets) in the current batch of transactions is held in the current window frame $W_0$.

This property says that the entire non-discriminative itemsets stay as internal node in current window frame $W_0$ (i.e., subset of discriminative itemsets) is saved with their exact frequencies. The exact frequencies of the non-discriminative itemsets (subsets) in the current window frame $W_0$ are obtained by traversing the $FP$-Tree through Header-Table links for their appearances in the current batch of transactions.

The first corollary is formally defined below.

**Corollary 1** An exact set of itemsets including discriminative and non-discriminative subsets in the current batch of transactions is held in the current window frame $W_0$. Where the current window frame $W_0$ is the $H$-DISStream prefix tree structure and the non-discriminative subsets are the subset of discriminative itemsets at least in one window frame $W_k$ (i.e., $k \geq 0$) in the tilted-time window model during the history of data streams.

**Rationale 1** ($H$-DISStream holds exact frequencies of itemsets from the time they are maintained in the tilted-time window model) Corollary 1 ensures that any itemset in the $H$-DISStream structure and its built-in tilted-time window model have the exact frequencies in each window frame $W_k$ (i.e., $0 \leq k \leq m$ where $m$ is the oldest window frame related to the itemset).

**Proof** The authenticity of the DISSparse algorithm for mining discriminative itemsets in a batch of transactions has been proved in Seyfi et al. [30]. Property ensures the exact set of discriminative itemsets is held in the current window frame $W_0$. Property 2 ensures to hold the exact frequencies of non-discriminative subsets in the current window frame $W_0$. The $FP$-Tree holds all the items including the infrequent items in the target data stream $S_t$. 

\[ \text{Springer} \]
This implies that the frequencies of non-discriminative subsets that in the current batch of transactions are infrequent in the target data stream \( S_i \), are not missed. The above two properties imply that any itemset in \( W_0 \) holds the exact frequencies in the current batch of transactions. The tilted-time window updating is done by shifting and merging the itemset frequencies in the smaller window frames, which are holding the exact frequencies of itemsets, to the larger window frames. These ensure the exact frequencies of the itemset in every tilted window frame \( W_0, W_1, ..., W_m \) (i.e., \( W_m \) is the oldest window frame related to itemset).

Based on Corollary 1, the exact itemset frequencies are held in each window frame by considering the oldest window frame where the itemset has appeared in the \( H\text{-DISStream} \) structure. However, the itemset frequencies may have been ignored during batch processing in the oldest window frames (i.e., as non-discriminative itemset without any discriminative superset). The frequencies of an itemset are less than or equal to its actual frequencies in the history of data streams considering all input batches \( B_1, B_2, ..., B_n \). Let \( W_x \) be the oldest possible window frame in \( H\text{-DISStream} \), and \( W_m \) be the oldest window frame related to the itemset \( I \) in \( H\text{-DISStream} \) (i.e., \( m \leq x \)), the following statements hold.

\[
\sum_{k=0}^{m} f^k(I) \leq \sum_{k=0}^{x} f^k(I), \quad \sum_{k=0}^{m} f^k(I) \leq \sum_{k=0}^{x} f^k(I)
\]

\[
\sum_{k=0}^{m} n^k_j \leq n_j, \quad \sum_{k=0}^{m} n^k_j \leq n_j
\]

As above, the frequencies of the itemsets in the tilted-time windows are equal to or smaller than their true frequencies in the data streams. The length of the data streams in the tilted-time windows is also equal to or smaller than their true lengths in the data streams. These caused approximate frequencies and approximate ratios in discriminative itemsets in the tilted-time window model. The discriminative itemsets with approximate frequencies less than their exact frequencies may be missed in the tilted-time window model. The approximate ratio can be less or more than the actual ratio considering the ratios in the oldest window frames \( W_k \), i.e., \( m < k \leq x \) (e.g., \( R^k_{ij}(I) \gg 1 \) or \( R^k_{ij}(I) \ll 1 \), respectively). Based on the above conditions, one of the following two statements holds.

\[
\forall l, 0 \leq l \leq m \leq x, \quad \frac{\sum_{k=l}^{m} f^k(l)}{\sum_{k=l}^{m} n^k_l} \leq \frac{\sum_{k=l}^{x} f^k(l)}{\sum_{k=l}^{x} n^k_l} \leq \frac{\sum_{k=l}^{x} f^k(l)}{\sum_{k=l}^{x} n^k_l}
\]

\[
\forall l, 0 \leq l \leq m \leq x, \quad \frac{\sum_{k=l}^{m} f^k(l)}{\sum_{k=l}^{m} n^k_l} > \frac{\sum_{k=l}^{x} f^k(l)}{\sum_{k=l}^{x} n^k_l} > \frac{\sum_{k=l}^{x} f^k(l)}{\sum_{k=l}^{x} n^k_l}
\]

Based on the above conditions, the frequency ratios in the tilted-time windows are either smaller or larger than the true frequency ratios in the data streams. To improve the accuracy, we define a relaxation ratio for discovering the discriminative itemsets between borders of the tilted-time window model as in section below.
3.3.2 Improving the accuracy using relaxation ratio

The discriminative itemsets with approximate ratios less than their exact ratios may be missed in the tilted-time window model, i.e., false-negatives. The non-discriminative itemsets with approximate ratios greater than their exact ratios may be reported in the tilted-time window model as discriminative itemsets, i.e., false-positives. Data stream mining algorithms basically must be designed based on a single scan [1, 17] as the multiple scans of datasets is often too expensive. The approximation can be refined using a relaxation of $\alpha$ by sub-discriminative itemsets. The sub-discriminative itemsets are discovered during batch processing by modifying the DISSparse algorithm and are saved in the current window frame $W_0$. The two heuristics proposed in DISSparse method [30] are modified by the relaxation of $\alpha$, for holding the sub-discriminative itemsets during the batch processing. The sub-discriminative itemsets are saved in the tilted-time window model as the potential discriminative itemsets and based on the relaxation of $\alpha$.

**Property 3** By modifying Corollary 1 using relaxation of $\alpha$ a set of non-discriminative itemsets in the current batch is held in $W_0$ are discovered as sub-discriminative itemset.

This property says that the sub-discriminative itemsets are discovered by choosing the relaxation of $\alpha$ from non-discriminative itemsets. The sub-discriminative itemsets in the current window frame $W_0$ and the history of data streams are discovered for better approximation in itemset frequencies and itemset frequency ratios.

**Property 4** By using smaller relaxation of $\alpha$ a better approximation will be in the discriminative itemsets in the tilted-time window model.

This property says that the more sub-discriminative itemsets are discovered by choosing the smaller relaxation of $\alpha$. This is a trade-off between better approximations in discriminative itemsets in the tilted-time window model and computation cost.

The second corollary is formally defined below.

**Corollary 2** A refined approximate bound in discriminative itemsets in the tilted-time window model is obtained by modifying the Corollary 1 based on relaxation of $\alpha$.

Where $\alpha$ is the relaxation threshold for sub-discriminative itemset and Corollary 1 is for holding the exact set of discriminative itemsets and non-discriminative subsets in the current batch of transactions in the current window frame $W_0$.

**Rationale 2** (Highest refined approximate bound in discriminative itemsets in the tilted-time window model) Corollary 2 ensures that the approximation in discriminative itemsets in the tilted-time window model may be improved by holding the sub-discriminative itemsets in the tilted-time window model.

**Proof** The sub-discriminative itemsets have potential to be discriminative by merging in the larger window frames. The sub-discriminative itemsets improve the approximate bound in discriminative itemsets by increasing the number of window frames that hold the exact frequencies of itemsets in the tilted-time window model. This caused a smaller number of false-positives and false-negatives in the discriminative itemsets in the tilted-time window model. Considering two relaxations of $\alpha$ and $\alpha'$, i.e., $\alpha \leq \alpha'$ the approximate bound for
itemset $I$ in the tilted-time window model is obtained by its exact frequencies from the time it is maintained in the window model $W_m$ and $W_{m'}$, respectively, i.e., $m \geq m'$. □

The size of the pattern tree (i.e., $H$-DISStream) and its built-in tilted-time window model can be large in the history of data stream. We propose a tail pruning technique for holding the in-memory data structures in a reasonable size as we explain in section below.

### 3.3.3 Tail pruning in the tilted-time window model

The discriminative itemsets are a sparse subset of frequent itemsets. The $H$-DISStream with built-in tilted-time window frames, in principle, is much smaller than the $FP$-Stream used in Giannella et al. [16] for frequent itemset mining in data streams using the tilted-time window model. However, without effective pruning techniques, the $H$-DISStream structure can still become unnecessarily large during the history of data streams. Using a reasonable relaxation of $\alpha$, a non-discriminative itemset in the $H$-DISStream structure is non-potential to be discriminative, with the approximate bound. The tail pruning in $H$-DISStream is applied for space saving by pruning the non-discriminative itemsets.

**Property 5** The set of non-discriminative itemsets are defined and can be tagged for deletion for space saving.

This property says that the large number of non-discriminative itemsets can be deleted for space saving in the tilted-time window model in data streams.

**Property 6** The set of non-discriminative itemsets stay as leaf node is deleted from tilted-time window model for space saving.

This property says that the large number of non-discriminative itemsets is deleted if they stay as leaf node. This tail pruning will cause for large space saving in the tilted-time window model in data streams.

The third corollary is formally defined below.

**Corollary 3** An itemset in the $H$-DISStream and its built-in tilted-time window model is pruned if it is non-discriminative itemsets and stays as tail itemset.

Where the tail itemset is not a subset of any discriminative or sub-discriminative itemsets in any $W_k$, i.e., $k \geq 0$ and the non-discriminative itemsets are defined in data streams in the tilted-time window model.

**Rationale 3** (Concise $H$-DISStream structure) Corollary 3 ensures that any itemset that is held in the tilted-time window model is a discriminative, sub-discriminative or non-discriminative subset in the history of data streams.

**Proof** The $H$-DISStream is made up of a compact prefix tree structure by sharing branches for their most common frequent items. The logarithmic built-in tilted-time window model is also a very compact data structure. Property 5 ensures that the discriminative and sub-discriminative itemsets are not pruned. Property 6 ensures that the non-discriminative subsets are not pruned. These imply that the itemsets are pruned if they have the least potential to be discriminative in the recent trends in the data streams. The more space is saved using
Corollary 3 by pruning the non-discriminative itemsets staying as leaf node and their possible direct non-discriminative subsets iteratively. In the tilted-time window model, consider itemsets $I \subset I'$ which are both in the $H\text{-DISStream}$ structure at the end of the batch processing. Let $W_0, W_{\frac{1}{2}}, ..., W_m$ and $W'_0, W'_1, ..., W'_{m'}$ be the window frames that are maintained in the tilted-time window model for the itemsets $I$ and $I'$, respectively. The number of window frames related to the itemset $I$ is equal to or more than the number of window frames related to the itemset $I'$ (i.e., $m \geq m'$).

The periodic changes in discriminative itemsets are happened by concept drifts in data streams and the discriminative itemsets in the neighbor window frames become considerably different. The pruned non-discriminative itemsets as explained are basically the least potential discriminative itemsets. The principles are proposed for setting the relaxation of $\alpha$ based on data stream characteristics.

**Claim 1** Based on Rationales 1, 2 and 3, the highest refined approximate bound in discriminative itemsets is achieved efficiently by setting relaxation of $\alpha$ to a reasonably small value and applying the tail pruning.

Claim 1 essentially says the approximation in discriminative itemset mining in data streams using the tilted-time window model is improved by saving a greater number of sub-discriminative itemsets. The sub-discriminative itemsets have potential to be discriminative itemsets in the recent history of data streams. We call this the highest refined approximate bound in the discriminative itemsets in the tilted-time window model, which is obtained by the smaller number of false-positives and false-negatives during the recent history of data streams.

In the next sections, a single-pass algorithm is proposed for mining discriminative itemsets in data streams using the tilted-time window model. The prefix tree structure $H\text{-DISStream}$ with the built-in tilted-time window model is used in the algorithm following the defined corollaries in this section.

### 3.4 H-DISSparse method

The $H\text{-DISSparse}$ method utilizes the $DISSparse$ algorithm [30] with the tilted-time window model. The discriminative (and sub-discriminative) itemsets are directly updated to the $H\text{-DISStream}$ structure (i.e., current window frame $W_0$), and the tilted-time window model is updated by shifting and merging the itemsets in the older window frames $W_k$ (i.e., $k > 0$). The $H\text{-DISSparse}$ method continues by discovering discriminative and sub-discriminative itemsets for the next batch of transactions.

#### 3.4.1 H-DISSparse algorithm

The $H\text{-DISSparse}$ algorithm is presented by incorporating three corollaries defined for the efficient discriminative itemset mining using the tilted-time window model and within the approximate bound guarantee. The $H\text{-DISStream}$ structure is an offline structure and is updated in offline time intervals when the current batch of transactions $B_n$ is full (i.e., $n \geq 1$). The first batch of transactions $B_1$ is treated differently by calculating all the item frequencies and making the $Desc\text{-Flist}$ based on the descending order of the item frequencies. The $Desc\text{-Flist}$ order is used for saving space by sharing the paths in the prefix trees...
with most frequent items on the top. This Desc-Flist remains the same for all the upcoming batches in data streams. The $H$-DISSparse algorithm is single-pass for the rest of the batches of transactions. The input parameters discriminative level $\theta$, support threshold $\varphi$ and relaxation of $\alpha$ are defined based on the application domain, data stream characteristics and sizes or by the domain expert users, as discussed in experiments. The $H$-DISSparse algorithm is represented in Algorithm 1.

The FP-Tree is made by adding the transactions from $B_n$ (i.e., the most current batch of transactions) without pruning infrequent items. The tilted-time window model is updated for larger window frames $W_k$ (i.e., $k > 0$) by shifting and merging as explained, based on the basic of the logarithmic tilted-time window frames as in Giannella et al. [16]. Following the DISSparse algorithm proposed in Seyfi et al. [30], the itemset combinations from potential discriminative subsets for each Header-Table item are generated. The relaxation of $\alpha$ for the approximate bound guarantee in discriminative itemset in the tilted-time window model is defined. The itemsets are checked based on the Definition 1 to be saved as discriminative and sub-discriminative itemsets or be deleted as non-discriminative itemsets if they are leaf nodes, respectively.

The $H$-DISStream structure as the current window frame $W_0$ is updated instantly by discriminative and sub-discriminative itemsets as in DISSparse algorithm [30]. By full discovery of the discriminative itemsets in $B_n$, the exact frequencies of the non-discriminative subsets not updated in $W_0$ are tuned using the FP-Tree following Corollary 1. By the end of updating the window frame $W_0$, the tail pruning is applied to the $H$-DISStream and its built-in tilted-time window model following Corollary 3. The discriminative itemsets in target data stream $S_i$ against general data stream $S_j$ in each window frame $W_k$ (i.e., $k \geq 0$) are reported in offline time intervals in each window frame in $DI_{ij}^k$ and the $H$-DISSparse algorithm is continued for the new incoming batch of transactions $B_{n+1}$. The main contribution in this algorithm is applying the tilted-time window model [16] to the DISSparse algorithm [30]. The properties of the discriminative itemsets in the tilted-time window model are used as three corollaries for more accurate approximate frequencies and more accurate approximate ratio bound. Without applying these corollaries to the algorithm, the approximations are not very accurate which caused for more false-positive and false-negative answers.
Algorithm 1 \textit{H-DISSparse}

\textbf{Input:} (1) The discriminative level $\theta$; (2) The support threshold $\varphi$; (3) The relaxation of $\alpha$; and (4) The Input batches $B_n$ i.e., $n \geq 1$ made of transactions with alphabetically ordered items belonging to data streams $S_i$ and $S_j$.

\textbf{Output:} $DI_{ij}^k$ i.e., $k \geq 0$, different set of discriminative itemsets in $S_i$ against $S_j$ in the tilted-time window model ($H$-DISSStream structure)

\textbf{Begin}
1. \textbf{While} not end of \textit{streams} \textbf{do}
2. \hspace{1em} Read current batch of transactions $B_n$;
3. \hspace{1em} Order the items in transactions based on \textit{Desc-List} made of $B_1$;
4. \hspace{1em} Make \textit{FP-Tree} for $B_n$ based on expansion of \textit{FP-Growth} (i.e., includes all items);
5. \hspace{1em} Update \textit{H-DISSStream} as $W_0$ using \textit{DISSparse} algorithm modified based on \textbf{Corollary 2}; (\textit{DISSparse} heuristics modified by \textbf{Corollary 2});
6. \hspace{1em} Update window frames $W_k$ (i.e., $k > 0$) by shifting and merging using \textit{FP-Stream} algorithm in (Giannella, Han et al. 2003);
7. \hspace{1em} Tune non-discriminative subset in $W_0$ using \textit{FP-Tree} (\textbf{Corollary 1});
8. \hspace{1em} Apply tail pruning in \textit{H-DISSStream} (\textbf{Corollary 3});
9. \hspace{1em} Report discriminative itemsets $DI_{ij}^k$ for each window frame $W_k$;
10. \textbf{End while};
\textbf{End.}

3.4.2 \textit{H-DISSparse} algorithm complexity

In the \textit{H-DISSparse} algorithm, the significant part attracting considerable complexity is related to the \textit{DISSparse} algorithm by generating the potential discriminative itemsets. Updating the tilted-time window model by shifting and merging, tuning the frequencies of the non-discriminative subsets and applying the tail pruning in the \textit{H-DISSStream} structure have less complexity compared to the \textit{FP-Stream}, by considering the sparsity property of discriminative itemsets. In \textit{H-DISSparse}, the tilted-time window model is updated after finding every pattern out of current batch of transactions. The efficiency of the \textit{H-DISSparse} algorithm is discussed in detail by evaluating the algorithm with the input data streams in experiments. Empirical analysis shows the performance of the proposed method by testing with different parameter settings (e.g., relaxation of $\alpha$). The efficiency of the \textit{H-DISSparse} algorithm is discussed on large and fast-growing data streams for mining discriminative itemsets with the approximate bound guarantee.
4 Performance evaluation

The algorithms were implemented in C++, and the experiments were conducted on a desktop computer with an Intel Core (TM) Duo E2640 2.8 GHz CPU and 8 GB main memory running 64-bit Microsoft Windows 7 Enterprise. The synthetic datasets were generated using the IBM synthetic data generator [2]. The $T : I : D$ format shows the datasets with $T$ as the average transaction length, $I$ as the average length of the maximal potentially large itemsets and $D$ as the number of transactions. We used the same $T$ for $S_1$ and $S_2$ to indicate that both datasets belong to the same domain. For simplicity, we defined the combination of $\theta n_1$ as minimum support. Using different synthetic data, we can control the number of discriminative itemsets in the output.

We evaluate the $H$-$DISSparse$ algorithm using data streams modelled as multiple batches of transactions. The main dataset is generated with $S_1$ as $T25 : I10 : D320K$ and $S_2$ as $T25 : I15 : D1600K$ limited to $1K$ unique items. The data streams are modelled as $32$ continuous batches in the same sizes (i.e., for the sake of clarity) with $T25 : I10 : D10K$ and $T25 : I15 : D50K$ belong to the target data stream $S_1$ and general data stream $S_2$, respectively. The ratio between size of $S_1$ and $S_2$ is also the same for all $32$ batches (i.e., $n_2/n_1 = 5$). The susy dataset from the UCI repository provided in Fournier-Viger et al. [15] is also used for evaluation with real datasets.

4.1 Benchmarking

The precision of the $DISTree$ method [29] has been confirmed by the completeness of itemset combination generation following the basics of the $FP$-$Growth$ [18] and correctness of discriminative and non-discriminative itemsets by full traversing the $DISTree$. The $DISSparse$ method [30] is proposed based on determinative heuristic for efficient mining of discriminative itemsets by limiting the itemset generation to the potential discriminative subsets. The precision of the $DISSparse$ method has been confirmed, based on its completeness of potential discriminative itemset combination generation and correctness of discriminative itemsets and non-discriminative itemsets. There is no such state of the art for comparing with the $H$-$DISSparse$ algorithm. Therefore, in the evaluation below, the method utilized $DISTree$ algorithm called $H$-$DISTree$ is chosen as a baseline model to compare with the proposed method utilized $DISSparse$ called $H$-$DISSparse$.

The differences between $H$-$DISSparse$ and existing methods in [8, 14, 31] prevents comparison between the discovered patterns in these methods. The discriminative itemsets discovered by the $H$-$DISSparse$ and the patterns discovered by these methods have different structural definitions. The discovered patterns in Tanbeer et al. [31] are frequent patterns in sliding window model. Frequent patterns in [8, 14] are discriminative in one dataset w.r.t class labels. They are discovered out of frequent patterns (classification rules) by feature selection (rule ranking). Discriminative patterns are extracted for each class vs whole dataset and not based on class differences. The discriminative measure of the pattern is based on information gain [26]. However, in our proposed $H$-$DISSparse$ method the itemsets are discriminative between two data streams (i.e., target vs general data streams). The discriminative measure is the relative frequencies of the itemsets in data streams. As a reason, these methods cannot be used as benchmark with $H$-$DISSparse$ method.
4.2 Evaluation on synthetic datasets

The scalability of \textit{H-DISSparse} is presented within offline updating of the tilted-time window model after processing each batch of transactions. In this section, during all experiments the discriminative level $\theta = 10$ and support threshold $= 0.01\%$, but the scalability of the algorithm is tested based on different relaxations of $\alpha$. It is assumed while the new batch is loaded with transactions the \textit{H-DISStream} updating can be done by processing the current batch of transactions. This works well as far as the algorithms are faster than the rate of incoming data streams.

The number of discriminative itemsets in the batches (i.e., presented in $W_0$) is different because of distributions of the transactions. The embedded knowledge and the trends in data streams change through time by the concept drifts. This has high effects on the algorithm scalability, as in Figs. 4 and 6. The discriminative itemsets in the larger tilted window frames are usually sparser. The data streams in the larger window frames $W_k$ (i.e., $k > 0$) have higher lengths and the discriminative itemsets appeared with high concept drifts are neutralized. For the sake of clarity, the scalability of the algorithm is first represented by time and space complexities for processing the batch of transactions (i.e., not considering tilted-time window model updating) as in Fig. 4. We used the \textit{DISTree} algorithm [29] and \textit{DISSparse} algorithm [30] for processing the batch of transactions.

![Fig. 4 Scalability of batch processing not considering the tilted-time window model updating](image1)

![Fig. 5 Tilted-time window model updating time complexity](image2)

![Fig. 6 Time complexity of \textit{H-DISTree} and \textit{H-DISSparse} algorithms](image3)
The variations in batch processing time and space are caused by the concept drifts in transaction distribution in the batches. The variations are not big in the DISSparse algorithm compared to the DISTree algorithm. However, the DISSparse algorithm also has high time and space complexities for processing the batches $B_1$ and $B_9$ with a high number of discriminative itemsets.

The tilted-time window model updating time for the algorithm is represented in Fig. 5. The H-DISSparse algorithm mainly has less time usage for the tilted-time window model updating compared to the time usage for batch processing. The fluctuations are mainly because of the tilted-time window updating by a different number of discriminative itemsets discovered in the batches. The high growths in the time usage of the algorithms are because of the wide tail pruning in the H-DISSstream structure caused by high concept drifts in the old batches; for example, after processing $B_3$, a wide number of non-discriminative itemsets is pruned during tail pruning process. These are mainly appeared in the tilted-time window model after processing $B_1$ with high concept drifts, as in Fig. 5. The same thing happens after processing $B_{11}$. This is caused by wide tail pruning for the large number of discriminative itemsets discovered in $B_9$.

We ran the full algorithm with the DISTree method for the batch processing, called H-DISTree. Obviously, the H-DISSparse algorithm is more efficient and scalable for batch of transactions with different characteristics. The full-time complexity of the H-DISTree and H-DISSparse algorithms is represented in Fig. 6. The H-DISTree time complexity is highly affected even by small concept drifts (e.g., the batch processing for $B_{28}$ is completely out of tolerable range). For the rest of the experiments in this section, the scalability of the H-DISSparse algorithm is represented by the full-time complexity in mining discriminative itemsets in the tilted-time window model.

The H-DISSstream size as the biggest data structure in the designed algorithms is presented in Fig. 7. Despite the batches with high concept drifts (e.g., the batch of transactions with large number of discriminative itemsets), the H-DISSstream size tends to become stable with very small growth by processing a larger number of batches in the data streams. The high growth in the size of H-DISSstream caused by concept drifts is quickly neutralized by processing the new batches and applying tail pruning (e.g., the growth in H-DISSstream size caused by $B_1$ and $B_9$ is neutralized by processing the next few batches). Following the compact logarithmic tilted-time window model and by applying the tail pruning as in Corollary 3, the H-DISSstream size stays small as in-memory data structure. The periodic drops in the size of H-DISSstream are caused by merging the tilted window frames. This can be seen more clearly in Fig. 7 after processing $B_8$, $B_{16}$, $B_{24}$, $B_{27}$ and $B_{32}$.

![Fig. 7 H-DISSstream structure size](image-url)
4.2.1 Approximation in discriminative itemsets in the tilted-time window model

In this section, the scalability of the \textit{H-DISSparse} algorithm, as a highly accurate and highly efficient method for mining discriminative itemsets using the tilted-time window model with the highest refined approximate bound, is evaluated with different parameter settings. The \textit{H-DISTree} algorithm is not scalable for processing large data streams with the highest approximate bound and is not evaluated.

Three corollaries are defined for mining discriminative itemsets using the tilted-time window model with the highest refined approximate bound. The relaxation of $\alpha$ in Corollary 2 is set for the highest refined approximate bound in discriminative itemsets in the tilted-time window model. The \textit{H-DISSparse} time usage and the \textit{H-DISSStream} size are represented in Fig. 8 by a different setting for the relaxation of $\alpha$, i.e., $\alpha = 1$, $\alpha = 0.9$ and $\alpha = 0.75$. The \textit{H-DISSparse} scales well by relaxation of $\alpha = 0.9$ with improvement in approximate discriminative itemsets as compared to relaxation of $\alpha = 1$. The \textit{H-DISSparse} scalability with smaller relaxation of $\alpha$ (e.g., $\alpha = 0.75$) is more sensitive to the concept drifts in data streams as with higher variations in time and space complexity in Fig. 8.

Figure 9 shows the number of sub-discriminative itemsets by setting relaxation of $\alpha = 0.9$ and $\alpha = 0.75$, respectively. The sub-discriminative itemsets are considered as overhead for the algorithms and can be increased exponentially by setting very small relaxation of $\alpha$ (e.g., by relaxation of $\alpha = 0.75$ the average number of sub-discriminative itemsets in the batches is greater than 1 million).

All the itemsets in the tilted-time window model, including the discriminative itemsets, are with full accuracy from the time they are saved in the window model. Therefore, the \textit{H-DISSparse} method discovers all the true discriminative itemsets in any tilted-time windows in the current history of the tilted-time window. Also, there is no false discriminative itemsets in the current history of the tilted-time window. The non-discriminative itemsets appears as subsets of discriminative itemsets are with true frequencies in data streams as well. These all ascertain the effectiveness of the proposed \textit{H-DISSparse} algorithm.
4.2.2 Discriminative itemsets in the tilted-time window model without tail pruning

The tail pruning defined in Corollary 3 is applied to the \textit{H-DISSStream} structure for space saving by pruning the least potential discriminative itemsets. Corollary 1 is defined for obtaining the exact frequencies of the itemsets from their maintaining time in the tilted-time window model. The exact frequencies of the non-discriminative subsets are obtained by traversing the \textit{FP-Tree} through \textit{Header-Table} links for their appearances in the current batch of transactions. Figure 10 shows the scalability of the original \textit{H-DISSSparse} algorithm eliminating Corollaries 1 and 3, respectively. Eliminating Corollary 1 adds more time complexity to the few batches (e.g., \(B_3, B_9, B_{11}\) and \(B_{17}\)), mainly because of adding wrong discriminative itemsets to the process. The \textit{H-DISSStream} size as compared to the original algorithm has small decreases by eliminating Corollary 1. This is caused by more tail pruning and higher approximation in discriminative itemsets in the tilted-time window model.

The \textit{H-DISSSparse} algorithm shows higher time complexity by eliminating Corollary 3 as a consequence of bigger data structures. However, in two points (i.e., after processing \(B_3\) and \(B_{11}\)) the time complexity decreases caused by eliminating the wide tail pruning of discriminative itemsets appeared by concept drifts. The \textit{H-DISSStream} size becomes much bigger during the time. The periodic drops in \textit{H-DISSStream} size after processing \(B_8, B_{16}, B_{24}, B_{27}\) and \(B_{32}\) is caused by merging the larger window frames in the tilted-time window model.

4.3 Evaluation on real datasets

To evaluate the proposed \textit{H-DISSSparse} algorithm on real applications, we ran the experiments on the real datasets. The susy dataset from the UCI repository provided in Fournier-Viger et al. [15] was used. The selected dataset is dense (i.e., transactions have values for each attribute) with less sparsity characteristics compared to the synthetic market basket datasets. For this reason, we set the parameters in a way to show the scalability in the best scales. The susy dataset is related to high-level features derived by physicist to help discriminate between two classes defined as signal and background. It is related to the particles detected using particle accelerator based on Monte Carlo simulations. This dataset is made of five million instances, and the first column is the class label followed by eighteen features. The transactions are made of about hundred ninety unique items. We selected the 32 batches each of them made of fifty thousand instances for the scale of the experiments. The \textit{H-DISSSparse} algorithm is evaluated with different parameter settings for the highest refined approximate bound in discriminative itemsets in the tilted-time window model.

![Fig. 10 Scalability of \textit{H-DISSsparse} algorithm by eliminating Corollaries 1 and 3](image-url)
4.3.1 Scalability on datasets with less concept drifts in the tilted-time window model

The scalability of $H$-DISSparse is presented within offline updating of the tilted-time window model after processing each batch of transactions. In this section, during all experiments the discriminative level $\theta = 2$ and support threshold $\phi = 1\%$ and the scalability of the algorithms are tested based on different relaxations of $\alpha$. It is assumed while the new batch is loaded with transactions the $H$-DISStream updating can be done by processing the current batch of transactions. This works well as far as the algorithms are faster than the rate of incoming data streams.

The number of discriminative itemsets in the batches (i.e., presented in $W_0$) is different because of distributions of the transactions. The embedded knowledge and the trends in data streams do not change through time by the concept drifts.

For the sake of clarity, the scalability of the algorithm is first represented by time and space complexities for processing the batch of transactions (i.e., not considering tilted-time window model updating) as in Fig. 11. We used the $DISTree$ algorithm [29] and DISSparse algorithm [30] for processing the batch of transactions as discussed.

The small variations in batch processing time and space are caused by the difference in the number of discriminative itemsets in the batches. The algorithm mainly has less time usage for the tilted-time window model updating compared to the batch processing. In the tilted-time window updating, there are no fluctuations mainly because of the similar number of discriminative itemsets discovered in the batches. There is no wide tail pruning compared to the experiments with synthetic datasets. This real dataset does not have high concept drifts in the batches.

We ran the full algorithm with the $DISTree$ method for the batch processing, called $H$-DISTree. Obviously, the $H$-DISSparse algorithm is more efficient and scalable for batch of transactions with different characteristics. The full-time complexity of the $H$-DISTree and $H$-DISSparse algorithms is represented in Fig. 12. The $H$-DISTree time complexity is affected even by small concept drifts. For the rest of the experiments in
In this section, the scalability of the \textit{H-DISSparse} algorithm, as a highly accurate and highly efficient method for mining discriminative itemsets using the tilted-time window model with the highest refined approximate bound, is evaluated with different parameter settings. The \textit{H-DISTree} algorithm is not scalable for processing large data streams with the highest approximate bound and cannot be evaluated in this section.

Three corollaries are defined for mining discriminative itemsets using the tilted-time window model with the highest refined approximate bound. The relaxation of $\alpha$ in Corollary 2 is set for the highest refined approximate bound. The \textit{H-DISSparse} time usage and the \textit{H-DISStream} size are represented in Fig. 14 by a different setting for the relaxation of $\alpha$, i.e., $\alpha = 1$ and $\alpha = 0.9$. The \textit{H-DISSparse} scales double by relaxation of $\alpha = 0.9$ with improvement in approximate discriminative itemsets compared to relaxation of $\alpha = 1$.

Figure 15 shows the number of sub-discriminative itemsets by setting relaxation of $\alpha = 0.9$. The sub-discriminative itemsets are considered as overhead for the algorithms and can be increased exponentially by setting very small relaxation of $\alpha$. 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{Fig_13.png}
\caption{\textit{H-DISStream} structure size}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{Fig_14.png}
\caption{Scalability of \textit{H-DISSparse} algorithm by relaxation of $\alpha = 1$ and $\alpha = 0.9$}
\end{figure}
4.4 Discussion

The general outcome from experiments on \textit{H-DISSparse} algorithm discussed in previous subsections is explained here. The \textit{H-DISSparse} algorithm exhibits efficient time and space complexity for mining discriminative itemsets using the tilted-time window model. The highest refined approximate bound in discriminative itemsets in the tilted-time window model is obtained efficiently based on Corollaries 1, 2 and 3 by the smaller relaxation of $\alpha$. Setting the relaxation of $\alpha$ in accompaniment with other parameters (i.e., support threshold $\varphi$, discriminative level $\theta$ and input batch size $n_i$) is very important in the real applications. Proper size has to be considered for the current window frame (i.e., $W_0$) for updating the tilted-time window model in reasonable time intervals. This is highly related to the application and domain experts by considering limited computing and storage capabilities and the approximate bound in the false-positive discriminative itemsets.

The changes in trend based on the concept drifts in batches are neutralized quickly in the tilted-time window model, and the in-memory \textit{H-DISStream} structure is held efficiently during the life time of data streams. The in-memory \textit{H-DISStream} without tail pruning (i.e., Corollary 3) is not efficient even by considering a compact logarithmic tilted-time window model. The \textit{FP-Tree} structure made of transactions in one batch considering all items is used efficiently in \textit{H-DISSparse} algorithm. The \textit{H-DISTree} algorithm is more sensitive to the concept drifts and not efficient even without considering sub-discriminative itemsets (i.e., relaxation of $\alpha = 1$). The main part of time and space complexity in the algorithms is related to the batch processing, although the tail pruning of non-discriminative itemsets appeared in the old batches can add complexity to the algorithms as well.

5 Conclusion and future works

 Discriminative itemsets show the distinguishing features of the target data stream in comparison with the general trends existed in the general data stream. This paper proposes a method for mining discriminative itemsets from fast-growing large data streams using the \textit{H-DISStream} structure. Three corollaries have been defined for improving the approximation in discriminative itemsets in the tilted-time window model. This paper presents the \textit{H-DISSparse} algorithm to extract discriminative itemsets accommodating several batches of streams data using the tilted-time window model. All the structures generated and used during the mining process are attempted to consume least time and space. The proposed
method has been extensively evaluated with datasets exhibiting distinct characteristics. The algorithm reports the discriminative itemsets with number of false-positive answers. The historical data structures generated during the process were able to be fitted in the main memory. Results ascertain that mining discriminative itemsets in multiple data streams is realistic in fast-growing data streams. In this paper, discriminative itemsets are updated in offline time intervals in the tilted-time window model. In the future, we propose the algorithm for mining discriminative itemsets in data streams using the sliding window model.

**References**

1. Aggarwal CC (2007) Data streams: models and algorithms. Springer, Berlin
2. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th international conference on very large data bases VLDB.
3. Alhammady H, Ramamohanarao K (2005) Mining emerging patterns and classification in data streams. In: The proceedings of IEEE/WIC/ACM international conference on web intelligence, pp 272–275
4. Amagata D, Hara T (2017) Mining top-k co-occurrence patterns across multiple streams. IEEE Trans Knowl Data Eng 29(10):2249–2262
5. Bailey J, Loekito E (2010) Efficient incremental mining of contrast patterns in changing data. Inf Process Lett 110(3):88–92
6. Bailey J, Manoukian T, Ramamohanarao K (2002) Fast algorithms for mining emerging patterns. In: Proceedings of the 6th European conference on principles of data mining and knowledge discovery
7. Chang JH, Lee WS (2003) Finding recent frequent itemsets adaptively over online data streams. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM
8. Cheng H, Yan X, Han J et al (2008) Direct discriminative pattern mining for effective classification. In: 2008 IEEE 24th international conference on data engineering, IEEE
9. Chi Y, Wang H, Philip SY et al (2004) Moment: maintaining closed frequent itemsets over a stream sliding window. In: Fourth IEEE international conference on data mining ICDM ’04
10. Chi Y, Wang H, Philip SY et al (2006) Catch the moment: maintaining closed frequent itemsets over a data stream sliding window. Knowl Inf Syst 10(3):265–294
11. Dong G, Bailey J (2012) Contrast data mining: concepts, algorithms, and applications. CRC Press, Boca Raton
12. Dong G, Li J (1999) Efficient mining of emerging patterns: discovering trends and differences. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining
13. Fan H, Ramamohanarao K (2002) An efficient single-scan algorithm for mining essential jumping emerging patterns for classification. In: Proceedings of the 6th Pacific-Asia conference on advances in knowledge discovery and data mining
14. Fan W, Zhang K, Cheng H et al (2008) Direct mining of discriminative and essential frequent patterns via model-based search tree. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining
15. Fournier-Viger P, Lin JC-W, Gomariz A et al (2016) The SPMF open-source data mining library version 2. In: Machine learning and knowledge discovery in databases: European conference, ECML PKDD 2016, Riva del Garda, Italy, 19–23 Sept 2016, Proceedings, part III. Springer, Cham, pp 36–40
16. Giannella C, Han J, Pei J et al (2003) Mining frequent patterns in data streams at multiple time granularities. Next Gener Data Min 212:191–212
17. Han J, Pei J, Kamber M (2011) Data mining: concepts and techniques. Elsevier, Amsterdam
18. Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. ACM sigmod record. ACM, New York
19. He Z, Gu F, Zhao C et al (2017) Conditional discriminative pattern mining. Inf Sci 375(3):1–15
20. He Z, Zhang S, Gu F et al (2019) Mining conditional discriminative sequential patterns. Inf Sci 478:524–539
21. Leonardo P, Fabio V (2018) Efficient mining of the most significant patterns with permutation testing. In: Proceedings of the 24th ACM sigkdd international conference on knowledge discovery & data mining. London, United Kingdom. ACM, pp 2070–2079
22. Li J, Liu G, Wong L (2007) Mining statistically important equivalence classes and delta-discriminative emerging patterns. In: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM
23. Lin Z, Jiang B, Pei J et al (2010) Mining discriminative items in multiple data streams. World Wide Web 13(4):497–522
24. Manku GS (2016) Frequent itemset mining over data streams. In: Garofalakis M, Gehrke J, Rastogi R (eds) Data stream management: processing high-speed data streams. Springer, Berlin, pp 209–219
25. Manku GS, Motwani R (2002) Approximate frequency counts over data streams. In: Proceedings of the 28th international conference on very large data bases, VLDB endowment
26. Quinlan JR (2014) C4.5: programs for machine learning. Elsevier, Amsterdam
27. Seyfi M (2011) Mining discriminative items in multiple data streams with hierarchical counters approach. In: Fourth international workshop on advanced computational intelligence (IWACI), 2011, IEEE
28. Seyfi M (2018) Mining discriminative itemsets in data streams using different window models. Queensland University of Technology, Brisbane
29. Seyfi M, Geva S, Nayak R (2014) Mining discriminative itemsets in data streams. In: International conference on web information systems engineering. Springer
30. Seyfi M, Nayak R, Xu Y et al (2017) Efficient mining of discriminative itemsets. In: Proceedings of the international conference on web intelligence, Leipzig, Germany. ACM, pp 451–459
31. Tanbeer SK, Ahmed CF, Jeong B-S et al (2009) Sliding window-based frequent pattern mining over data streams. Inf Sci 179(22):3843–3865
32. Yu K, Ding W, Simovici DA et al (2015) Classification with streaming features: an emerging-pattern mining approach. ACM Trans Knowl Discov Data 9(4):1–31
33. Yu K, Ding W, Wang H et al (2013) Bridging causal relevance and pattern discriminability: Mining emerging patterns from high-dimensional data. IEEE Trans Knowl Data Eng 25(12):2721–2739
34. Zhang X, Dong G, Kotagiri R (2000) Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets. In: Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining

Majid Seyfi is a researcher in the areas of data mining, machine learning and databases, and has made contributions to related research areas. His current research interests include data stream mining, contrast data mining, pattern mining, classification, and big data.
Richi Nayak is an associate professor in the School of Computer Science, Queensland University of Technology, Australia. She is an active researcher in the areas of data mining and machine learning. She has developed a number of novel algorithms to deal with diverse data sets such as text, time-series, stream as well as originating from multiple domains. She has published over 200 refereed papers, most of her journal papers have been published in top Q1 journals and major conferences. Her current research interests include deep learning and matrix/tensor factorization techniques to deal with diverse datasets.

Yue Xu is an associate professor in the School of Computer Science, Queensland University of Technology, Australia. She is an active researcher in the areas of data mining, machine learning and web intelligence, and has made important contributions to related research areas. She has published over 150 refereed papers over the past 10 years, some of which have been published in top journals such as Information Science, IEEE Transactions on Knowledge and Data Engineering, European Journal of Information Systems, ACM Transactions on Intelligent Systems and Technology, Knowledge-Based Systems, etc., and major conferences such as ICDM, CIKM, WWW and ICIS. Her current research interests include text mining, recommender systems, web intelligence, and user behavior modeling.

Shlomo Geva is an adjunct professor in the School of Computer Science, Queensland University of Technology, Australia. He is an active researcher in the areas of information retrieval, bioinformatics, text mining, machine learning, recommender systems and web intelligence, and has made contributions to related research areas. He has published over 150 refereed papers, some of which have been published in top journals such as BMC bioinformatics, information research, information processing and management, IEEE Transactions on Knowledge and Data Engineering, Neural Computation, IEEE Transactions on Neural Networks, etc., and major conferences such as ICDM, CIKM, WWW and ICIS. His current research interests include bioinformatics, text mining, recommender systems and web intelligence.