Novelty Detection from Evolving Complex Data Streams with Time Windows

Michelangelo Ceci, Annalisa Appice, Corrado Loglisci, Costantina Caruso, Fabio Fumarola, and Donato Malerba

Dipartimento di Informatica, Università degli Studi di Bari
via Orabona, 4 – 70126 Bari, Italy
{ceci,appice,loglisci,caruso,ffumarola,malerba}@di.uniba.it

Abstract. Novelty detection in data stream mining denotes the identification of new or unknown situations in a stream of data elements flowing continuously in at rapid rate. This work is a first attempt of investigating the anomaly detection task in the (multi-)relational data mining. By defining a data block as the collection of complex data which periodically flow in the stream, a relational pattern base is incrementally maintained each time a new data block flows in. For each pattern, the time consecutive support values collected over the data blocks of a time window are clustered, clusters are then used to identify the novelty patterns which describe a change in the evolving pattern base. An application to the problem of detecting novelties in an Internet packet stream is discussed.

1 Introduction

A data stream is an ordered sequence of data elements generated at rapid rate. Differently from data in traditional static databases, data streams are continuous, unbounded, usually come with high speed and have a data distribution which may change with time because of fundamental changes in the underlying phenomena [8]. These characteristics of data stream pose specific computational issues which prevent the application of traditional data mining algorithms, which are designed to extract knowledge from static data only. First, the continuous, unbounded, and high speed characteristics of data streams require abilities to collect and process a huge amount of data. Anyway, there is neither enough time to rescan the whole stream each time an update occurs nor enough space to store the entire data stream for online processing. Second, the temporal evolution which typically characterizes the distribution of data in a stream demands for techniques that can capture the evolution of extracted patterns as well.

Several efficient and effective algorithms have been proposed in the literature to extract knowledge from data streams, mainly for clustering, association rules discovery, time series analysis and novelty detection [3,10,5,6]. Most of these algorithms perform an incremental learning [14] which makes them able to face issues posed by continuous, unbounded, evolving characteristics of data streams. However, a common limitation is that they are designed to mine data elements which arrive as vectors of fixed attribute values. In many applications, the stream

J. Rauch et al. (Eds.): ISMIS 2009, LNAI 5722, pp. 563–572, 2009.

is actually a sequence of complex data elements, composed of several objects of various data types which are someway related. For instance, network traffic in a LAN can be seen as a stream of connections, which are described by some properties (e.g., protocol) as well as by the sequence of one or more packets which flows consecutively in the network as part of the same connection. The structure of these complex data elements is naturally modeled by several relations of a relational databases, hence making methods of (multi-)relational data mining [4] more suitable for the discovery of useful patterns from these data streams. Currently, query processing engines [2,9] have been realized in order to query stream stored into relational database systems and deliver result sets on-the-fly. These systems integrate next generation query languages which extend SQL in order to extract data from the stream stored in the database, but they do not integrate multi-relational data mining systems to discover novel and unknown patterns from these data.

In this work we focus on the novelty (or anomaly) detection task in complex data stream mining. Anomaly detection targets learning algorithms [7,12,17] being able to identify unknown situations which represent a change with respect to situations experienced before. This work addresses the anomaly detection problem by irstly resorting to a multi-relational data mining algorithm, called Mr-NoDeS (*M*ulti-*R*elational *No*velty *De*tection in Data *S*tream), which is based on an incremental approach to mine a relational pattern base from the complex data lastly flowed in a stream and provides a human interpretable description of the changes which occur in this evolving pattern base. New iterations of mining results are built based on old mining results. The algorithm is based on the definition of time sensitive data block [7], that is, the set of complex data which are periodically (e.g., daily, monthly) added to a stream. The pattern base includes relational patterns (i.e., patterns possibly involving several database relations) which are frequent on at least one data block of a time window ending at current time. The time window is defined as a user-defined number of the lastly income blocks. Novelty patterns are those patterns in the base whose frequency on the last block significantly changes with respect to an "homogeneous" region of frequencies computed in the remaining window blocks.

The paper is organized as follows. Section 2 presents preliminary concepts and defines novelty patterns. The algorithm Mr-NoDeS is described in Section 3. Section 4 reports an application of the proposed algorithm on an Internet packet stream. Lastly, some conclusions are drawn.

2 Preliminary Concepts and Definitions

In the traditional streaming model, the input data elements $\mathbf{a_1}, \mathbf{a_2}, \ldots \mathbf{a_n}, \ldots$ arrive sequentially, item by item, as continuous fixed-length vectors of attribute values $\mathbf{a_i}$. This attribute-value representation appears unnatural and inadequate in several real world data stream applications where data elements are complex data which consist of several objects possibly belonging to heterogeneous data types. These objects may play different roles, hence it is necessary to distinguish

between the set S of reference (or target) objects, which are the main subject of analysis, and the sets R_k, $k = 1, \ldots, M$, of task-relevant (or non-target) objects, which are related to the former and can contribute to account for the variation. In the relational data model, both reference objects and task-relevant object can be naturally modeled as distinct relations (or tables) of a relational database D. Let H be the schema of D, H includes the definition of a (target) relation T_S which stores the properties (or attributes) of objects in S as well as an arbitrary number of additional (non-target) relations T_{R_k}, where each T_{R_k} stores attributes of objects in R_k. A reference object $s \in S$ defines a unit of analysis $D[s]$. Task-relevant objects which are someway related to the reference object contribute to defining the unit of analysis without being the main subject of the analysis. The "structure" of units of analysis, that is, the relationships between reference and task-relevant objects, is expressed in the schema H by foreign key constraints (FK). Foreign keys make it possible to navigate the data schema and retrieve all the task-relevant objects in D which are related to a reference object. In this way, a unit of analysis is retrieved by simply navigating the database structure without requiring any additional pretreatment of the complex data arriving in the stream.

Definition 1 (Unit of analysis). *A unit of analysis $D[s]$ consists of a reference object $s \in T_S$ and the finite set of task-relevant objects stored in some T_{R_k} which are related to s according to foreign key constraints of H.*

This notion of unit of analysis is coherent with the individual-centered representation [1], which has both theoretical (PAC-learnability) and computational advantages (smaller hypothesis space and more efficient search). In a streaming model, units of analysis are associated with time points.

Definition 2 (Complex data stream). *Let t_1, t_2, ..., t_i, ... a sequence of time points and let $\prec$ the order relationship defined among them. A complex data stream is a continuous flow DS of units of analysis associated with the time points, i.e., $DS = \{\langle D[s_1], t_1\rangle, \langle D[s_2], t_2\rangle, \langle D[s_n], t_n\rangle, \ldots\}$, where $t_i \prec t_{i+1}$.*

Time intervals define data blocks in a complex data stream [7].

Definition 3 (Data block). *Given a time point t and a time period p, a basic data block B is the set of units of analysis $D[s_i]$ such that their associated time t_i is in $[t - p + 1, t]$, i.e. $t_i \in [t - p + 1, t]$.*

The number of units of analysis in a basic data block B_i is denoted as $|B_i|$.

Given a time period p, a complex stream can be partitioned into consecutive data blocks B_1, B_2, ... such that B_i includes the units of analysis observed in the time interval $[t_0 + (i - 1) \cdot p, t_0 + i \cdot p]$. Since we are interested to capture evolutions (and novelties) from block to block, we extend the notion of time window to data blocks.

Definition 4 (Time window). *Let w be a window size. Then the time window $W(i, w)$ associated with each B_i is the set of basic blocks B_{i-w+1}, ..., B_i.*

Mining only data in a time window is a natural choice in data stream mining. Indeed, data distribution in a stream may change in time and we are interested in discovering a model which is descriptive of the recently income data only. In this work, the discovered model is intended as a base of "relational" patterns. In order to formally define a relational pattern, we introduce the concepts of key predicate, structural predicate and property predicate.

Definition 5 (Key predicate). *The key predicate associated with the target table T_S in H is a unary predicate $p(t)$ such that p denotes the table T_S and the term t is a variable that represents the primary key of T_S.*

Definition 6 (Property predicate). *A property predicate is a binary predicate $p(t, c)$ associated with the attribute Att of the table T_i. The name p denotes the attribute Att, the term t is a variable representing the primary key of T_i and c is a constant which represents a value belonging to the range of Att in T_i.*

Definition 7 (Structural predicate). *A structural predicate is a binary predicate $p(t, s)$ associated with the foreign key constraint FK from the table T_i to the table T_j in H. The name p denotes FK, while the terms t and s are two variables which represent foreign key in T_i and the primary key in T_j according to FK*[1].

A relational pattern is defined as follows:

Definition 8 (Relational pattern). *A relational pattern P over the schema H is a conjunction of predicates $p_0(t_0^1), p_1(t_1^1, t_1^2), p_2(t_2^1, t_2^2), \ldots, p_m(t_m^1, t_m^2)$, where $p_0(t_0^1)$ is the key predicate associated with the table T_S and $p_i(t_i^1, t_i^2)$, $i = 1, \ldots, m$, is either a structural predicate or a property predicate over H.*

An example of relational pattern is reported in Example 1.

Example 1. Let us consider a stream of connections incoming a firewall, where each connection is a sequence of consecutive packets. An example of relational pattern P is the following:

connection(C),protocol(C,tcp),contain_packet(C,D),contain_packet(C,E),
D≠E, next(D,E)

where *connection* is a key predicate, *protocol* and *time* are property predicates, *contain_packet* is a structural predicate.

The support of a relational pattern P on a block B_i is computed as follows:

$$s_i(P) = \frac{|\{D[s] | \langle D[s], t\rangle \in B_i, \exists \theta : P\theta \subseteq D[s]\}|}{|\{D[s] | \langle D[s], t\rangle \in B_i\}|}, \tag{1}$$

where θ is a substitution of variables into constants and $P\theta$ denotes the application of the substitution θ to the pattern P. Therefore, we define a relational

[1] In database theory, a foreign key constraint allows certain attributes (foreign key) in one table to refer to attributes (primary key) in another table. In our formalization, both the primary key and the foreign key of a constraint are mapped into two variables which univocally identify the pairs of tuples related by the constraint.

pattern P as *frequent* with respect to a minimum support threshold *minSupp* if a block B_i exists, such that $s_i(P) \geq minSupp$.

A novelty pattern on a time window can be formally defined as follows.

Definition 9 (Novelty pattern). *Let (1)* $W(i,w) = \langle B_{i-w+1}B_{i-w+2}\ldots, B_i\rangle$ *be a time window with length* w *and an ending block* B_i*; (2)* P *be a pattern and* $\langle s_{i-w+1}, s_{i-w+2}, \ldots, s_i\rangle$ *the list of support values of* P *on each data block of* $W(i,w)$*; (3)* $\Theta_P : [0,1] \rightarrow \Psi$ *be a discretization function which associates a support value of* P *in the interval* $[0,1]$ *with a discrete values* $\psi \in \Psi$*. Then,* P *is a* novelty pattern *for the time window* $W(i,w)$ *if and only if* $\Theta(s_{i-w+1}(P)) = \ldots = \Theta(s_{i-1}(P)) \neq \Theta(s_i(P))$.

3 The Algorithm

The algorithm Mr-NoDeS is a two step data stream algorithm that is triggered each time a p sized data block arrives in the stream. The algorithm is designed to record only data blocks falling in a w sized time window. In the first step, the relational pattern base $M(i,w)$ is updated each time a data block B_i arrives, while in the second phase patterns in $M(i,w)$ are filtered out in order to keep only those patterns which represent novelty patterns within the time window $W(i,w)$. Details on relational pattern discovery, pattern base maintenance and novelty pattern detections are reported in the next subsections.

3.1 Relational Pattern Discovery

The pattern discovery is performed by exploring level-by-level the lattice of relational patterns ordered according to a generality relation ($\geqslant$) between patterns. Given two patterns P_1 and P_2, $P_1 \geqslant P_2$ denotes that $P1$ (P_2) is more general (specific) than P_2 (P_1). The search proceeds from the most general pattern and iteratively alternates the candidate generation and candidate evaluation phases as in the levelwise method [13]. Candidate generation assumes that the space of pattern is structured according to the θ-subsumption generality order [15].

Definition 10 (θ-subsumption generality order). *Let* P_1 *and* P_2 *be two relational patterns.* P_1 *is more general than* P_2 *under* θ*-subsumption, denoted as* $P_1 \geqslant_\theta P_2$*, if and only if a substitution* θ *exists such that* $P_2\theta \subseteq P_1$.

Example 2. Let us consider the patterns: (P_1) connection(C). (P_2) connection(C), packet(C,P). (P_3) connection(C), service(C,'http'). These patterns are ordered as follows: $P_1 \geqslant_\theta P_2$, $P_1 \geqslant_\theta P_3$.

The θ-subsumption generality order satisfies the monotonicity property with respect to support (i.e., a specialization of an infrequent pattern cannot be frequent) and defines a quasi-ordering which can be searched according to a downward refinement operator which computes the refinements for a relational pattern [11]. The downward refinement operator ρ' used in this work is defined below.

Definition 11 (Downward refinement operator). *Let P be a relational pattern. Then $\rho'(P) = \{P \cup \{p(t_1, t_2)\} | p(t_1, t_2)$ is a structural or property predicate that shares at least one term with one predicate already occurring in P.*

ρ' is a refinement operator under θ-subsumption, i.e., $P \geqslant_\theta Q$ for all $Q \in \rho'(P)$. Due to the monotonicity property of θ-subsumption generality order with respect to support, ρ' allows for a level-wise exploration of the quasi-ordered set of relational patterns. Indeed, the implemented algorithm starts from a set $\wp$ containing only the most general pattern, i.e. the pattern that contains only the key predicate, and then updates $\wp$ by repeatedly applying ρ' to all patterns in $\wp$. In generating each level of the quasi-ordered set, the candidate pattern search space is represented as a set of enumeration trees (SE-trees)[18]. The idea is to impose an ordering on atoms such that all patterns in the search space are enumerated. Practically, a node g of a SE-tree is represented as a group comprising: the *head* ($h(g)$), i.e. the pattern enumerated at g, and the *tail* ($t(g)$) that is the ordered set consisting of all atoms which can be potentially appended to g by ρ' in order to form a pattern enumerated by some sub-node of g. A child g_c of g is formed by taking an atom $q \in t(g)$ and appending it to $h(g)$. Therefore, $t(g_c)$ contains all atoms in $t(g)$ that follows q (see Figure 1). In the case q is a structural predicate, $t(g_c)$ contains both atoms in $t(g)$ that follows q and new atoms directly linkable to q according to ρ' not yet included in $t(g)$. Given this child expansion policy, without any pruning of nodes or pattern, the SE-tree enumerates all possible patterns and prevents the generation and evaluation of candidate equivalent under θ-subsumption to some other candidate.

As pruning criterion, the monotonicity property of the generality order $\geqslant_\theta$ with respect to the support value is exploited to avoid refinement of infrequent relational patterns. An additional pruning criterion stops the search when a maximum number of literals ($MaxNumLiterals$) have been added to a pattern.

3.2 Pattern Base Maintenance

The pattern base $M(i, w)$ is the set of relational patterns which are frequent on at least one basic block of the time window $W(i, w)$. A pattern $P \in M(i, w)$ is associated with a support list $\langle s_{i-w+1}(P), s_{i-w+2}(P) \ldots, s_i(P)\rangle$, that is, the list of support values computed for P on each data block of $W(i, w)$. A pattern base $M(i, w)$ is mined starting from $M(i-1, w)$ when the data block B_i arrives in the stream. Operations to maintain the pattern base are inserting frequent patterns,

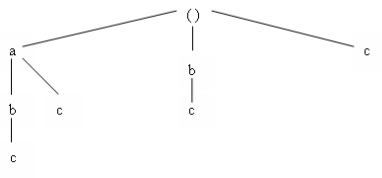

Fig. 1. The enumeration tree to search the patterns a, b, c, ab, ac, bc, abc

deleting unfrequent patterns and updating the support list of a pattern already belonging to the base. We can distinguish between three cases based upon the serial number i of the data block B_i which arrives in the stream.

($\mathbf{i=1}$). The pattern base $M(1,w)$ is mined from scratch: $M(1,w)=\{P_i|s_1(P_i) \geq minSupp\}$.

($\mathbf{i=2,\ldots,w}$). Relational patterns which are frequent on B_i are discovered and used to construct M_i^+, that is the set of patterns P which are frequent on B_i, but do not belong to $M(i-1,w)$. $M(i,w)$ is constructed by adding patterns of M_i^+ to $M(i-1,w)$. For each pattern $P \in M(i-1,w)$, the associated support list $\langle s_1(P),\ldots,s_{i-1}(P)\rangle$ is updated by adding $s_i(P)$. Differently, for each pattern $P \in M_i^+$, the entire support list $\langle s_1(P),\ldots,s_i(P)\rangle$ is built from scratch.

($\mathbf{i>w}$). The sliding time window moves from $B_{i-w},\ldots,B_{i-1}$ to $B_{i-w+1},\ldots,B_i$ and the data block B_{i-w} is removed from memory. Relational patterns which are frequent on B_i are discovered and used to construct both M_i^+ and M_i^-.

1. M_i^+ that is the set of patterns P which are frequent on B_i but do not belong to $M(i-1,w)$;
2. M_i^- that is the set of patterns P which are unfrequent on B_i, belong to $M(i-1,w)$ but are unfrequent on $B_{i-w+1},\ldots,B_{i-1}$.

$M(i,w)$ is then constructed from $M(i-1,w)$ by adding patterns of M_i^+ and removing patterns of M_i^-. The support list of each pattern $P \in M(i,w)$ is updated or created from scratch. An update operation is performed when the pattern P is already included in $M(i-1,w)$ ($P \in M(i,w)-M_i^+$). In this case, the support list associated to P is updated by removing $s_{i-w}(P)$ and by adding $s_i(P)$. A creation operation is performed when $P \in M_i^+$ and the entire support list $\langle s_{i-w+1}(P),\ldots,s_i(P)\rangle$ is built.

3.3 Time-Window Based Novelty Pattern Detection

Mr-NoDeS post-processes the pattern base $M(i,w)$ in order to identify novelty patterns for B_i. The function Θ_P is a discretization function based on a density based clustering algorithm. Several clustering algorithms have been designed in the literature. In this paper we use the density-base clustering algorithm DBSCAN [16] which is devised to discover arbitrary-shaped clusters which are discovered without providing a-priori the number of clusters. The complexity is quadratic with respect to the size w of the window ($\mathbf{O}(w^2)$). Clusters are intended as dense, timely consecutive, areas.For each pattern P, the cluster construction starts from a data block b (seed) and constructs the neighborhood $N(b)$ that includes b and the data block incoming before b as well as the data block incoming after b in the stream. The neighborhood is labeled as a cluster c only if it satisfies the condition of dense region. Density is estimated by means of the standard deviation of the support values of P falling in $N(b)$. Standard deviation must not exceed a user-defined threshold ($maxStd$). The cluster is then expanded by merging partially overlapping neighborhoods. The merge is performed only

in the case that the cluster which is output of the merge operation satisfies the condition of dense region. To avoid to evaluate the possible expansion of a cluster by merging an heterogeneous neighborhood, Mr-NoDeS evaluates only the merge of neighborhoods including support values whose standard deviation does not exceed a local user-defined threshold ($localMaxStD$). If a cluster cannot be further expanded, a new seed is selected and a new cluster is constructed. The strategy adopted to select the seed is the sequential one. The cluster c is labeled with the interval $[minC, maxC]$ where $minC$ ($maxC$) is the minimum (maximum) support value falling in c.

Each time, the clustering algorithm segments the time window $W(i, w)$ of P in only two clusters, namely $c1$ and $c2$, such that $c1$ includes the support values for data blocks $B_{i-w+1}, \ldots, B_{i+1}$ and $c2$ includes the support value for the data block B_i, then P is marked as a novelty pattern for B_i over $W(i, w)$. In other words, patterns whose support value passes from a cluster to another, are marked as novelty patterns.

4 The Application

Mr-NoDeS is applied to detect anomaly patterns in the Internet packet stream incoming the firewall of the Department of Informatics in Bari from June 1st to June 28th, 2004. Units of analysis are time-stamped ingoing connections (reference objects) which are composed by several packets (task-relevant objects).

Dataset Description. A connection is described by the identifier (integer); the protocol (nominal); the starting time (integer); the IP destination (nominal); the service (nominal); the number of packets (integer); the average packet time distance (integer); the time length (integer); the source nation code (nominal); the source nation time zone (integer). Each packet is described by the identifier (integer) and the starting time (number) of the packet within the connection. The interaction between consecutive packets is described by the time distance. Numeric attributes are discretized through an equal-width discretization that partitions each range of values into 10 bins.

Analysis of results. In these experiments, the data block size is 24 hours, while starting point is at 00:00 on June 1st, 2004. Novelty pattern discovery is triggered each time a new data block arrives in the stream by setting $minSupp = 0.1$, $MaxNumLiterals = 5$, $maxStD = 0.02$, $localMaxStD = 0.05$ and $w = 3, 4, 5, 6$. The number of anomaly patterns is plotted in Figure 2. Interestingly, the number of patterns extracted for each time window is rather large. This is due to the high number of similar extracted patterns. In fact, in most of cases, Mr-NoDeS extracts the patterns that are related each other according to the θ-subsumption generality order (one is the specialization of the other). However, the number of discovered novelty patters significantly decreases for $w = 6$, where the average number of patterns extracted for each data block is 59.48. This makes it possible to manually analyze patterns. In addition, by observing the smoothing of peaks in the number of novelty patterns per

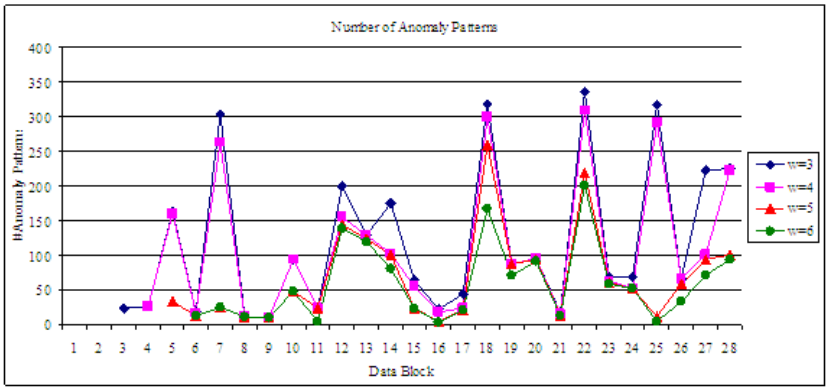

Fig. 2. Number of anomaly patterns discovered on each data block by varying w

data blocks we observe that the cardinality of anomaly pattern base presents a high variance over the different data blocks when $w = 3$, while this variance is somehow mitigated by increasing values of w. This would help the user to identify and analyze critical days, when attacks may have occurred. There are several critical data blocks (days) when $w = 3$ and less when $w = 6$. In particular, days where the number of extracted novelty patters is greater than 200 decreases from B_7, B_{11}, B_{18}, B_{22}, B_{25}, B_{27}, B_{28} when $w = 3$ to B_{22} when $w = 6$. An example of novelty pattern P_1 detected for the data block B_{22} (June 22th) by using $w = 5$ is "$conn(C), packet(C, P), proto(C, udp)$", where $s_{18}(P_1)$=0.0420, $s_{19}(P_1)$=0.078, $s_{20}(P_1)$=0.095 and $s_{21}(P_1)$=0.0422 are automatically clustered in a single region labeled with $[0.0420, 0.095]$, while $s_{22}(P)$=0.71 is clustered alone. This pattern describes as an anomaly on June 22th, 2004 the sharp increase of percentage of udp connections incoming the firewall. An example of novelty pattern P_2 which takes into account the relational nature of data is extracted on June 20th, 2004 with $w = 5$, that is, "$conn(C), packet(C, P), packToPack(P, Q), dist(P, Q, [0, 0.280])$, $service(C,$ $unknown)$". P_2 has a support value of 0.213 on the June 20th 2004 (B_{20}), while its support is clustered in the region $[0.0, 0.0]$ in the previous days of the window $W[20, 5]$. This pattern detects as anomalous the high number of connections C which use an *unknown* service and include at least two packets P and Q, where P is sent after Q with a time distance that is in the interval [0, 280] ms.

5 Conclusions

In this paper, we present a multi-relational data mining algorithm to discover novelty patterns from evolving data blocks of complex data streams. Complex data are stored in several relations of a relational database. A relational pattern base is maintained for the stream data falling in the current a time window. A clustering algorithm is employed to detect sharp change of support values. The algorithm is applied for the anomaly detection in an Internet Packet stream.

Acknowledgment

This work is partial fulfillment of objective of ATENEO-2008 project "Scoperta di conoscenza in domini relazionali" and Strategic Project PS121: "Telecommunication Facilities and Wireless Sensor Networks in EmergencyManagement".

References

1. Blockeel, H., Sebag, M.: Scalability and efficiency in multi-relational data mining. SIGKDD Explorations Newsletter 5(1), 17–30 (2003)
2. Brenna, L., Demers, A., Gehrke, J., Hong, M., Ossher, J., Panda, B., Riedewald, M., Thatte, M., White, W.: Cayuga: a high-performance event processing engine. In: International Conference on Management of Data, pp. 1100–1102. ACM, New York (2007)
3. Domingos, P., Hulten, G.: Mining high-speed data streams. In: the 6th International Conference on Knowledge Discovery and Data Mining, KDD 2000, pp. 71–80. ACM, New York (2000)
4. Džeroski, S., Lavrač, N.: Relational Data Mining. Springer, Heidelberg (2001)
5. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: Mining data streams: a review. SIGMOD Record 34(2), 18–26 (2005)
6. Gama, J.: Issues and challenges in learning from data streams. In: Kargupta, H., Han, J., Yu, P.S., Motwani, R., Kumar, V. (eds.) Data Mining and Knowledge Discovery Series on Next Generation of Data Mining, pp. 209–222. Chapman and Hall, CRC Press, Taylor and Francis Group (2009)
7. Ganti, V., Gehrke, J., Ramakrishnan, R.: Mining data streams under block evolution. SIGKDD Explorations 3(2), 1–10 (2002)
8. Guha, S., Koudas, N., Shim, K.: Data-streams and histograms. In: the 33th Symposium on Theory of Computing, STOC 2001, pp. 471–475. ACM, New York (2001)
9. `http://www.streambase.com/`
10. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: the 7th International Conference on Knowledge Discovery and Data Mining, KDD 2001, pp. 97–106. ACM, New York (2001)
11. Lisi, F.A., Malerba, D.: Inducing multi-level association rules from multiple relations. Machine Learning 55(2), 175–210 (2004)
12. Ma, J., Perkins, S.: Online novelty detection on temporal sequences. In: the 9th International Conference on Knowledge Discovery and Data Mining, KDD 2003, pp. 613–618. ACM, New York (2003)
13. Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. Data Mining and Knowledge Discovery 1(3), 241–258 (1997)
14. Mitchell, T.: Machine Learning. McGraw Hill, New York (1997)
15. Plotkin, G.D.: A note on inductive generalization. Machine Intelligence 5, 153–163 (1970)
16. Sander, J., Ester, M., Kriegel, H.-P., Xu, X.: Density-based clustering in spatial databases: The algorithm gdbscan and its applications. Data Mining and Knowledge Discovery 2(2), 169–194 (1998)
17. Spinosa, E.J., de Carvalho, A.P.d.L.F., Gama, J.: Cluster-based novel concept detection in data streams applied to intrusion detection in computer networks. In: The Symposium on Applied Computing, SAC 2008, pp. 976–980. ACM, New York (2008)
18. Zhang, X., Dong, G., Kotagiri, R.: Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets. In: Knowledge Discovery and Data Mining, pp. 310–314 (2000)