A Relational Approach to Novelty Detection in Data Streams

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

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

Abstract. A data stream is a sequence of time-stamped data elements which arrive on-line, at consecutive time points. In this work we propose a multi-relational approach to mine complex data streams in order to identify novelty patterns which target new or unknown situations in the stream. Multi-relational data mining is motivated by the existence of several real-world data stream applications where data elements are complex data scattered in several database relations of a relational database. In our proposal a stream is mined according to a data block model, that is, the stream is segmented in a sequence of data blocks where each data block consists of complex data elements arriving in a user define period (e.g., daily or monthly). A relational pattern base is mined each time a new data block arrives in the stream and a time window is used to filter out novelty patterns. An application of the proposed algorithm to the problem of detecting anomalies in network traffic is described

1 Introduction

A data stream is an ordered sequence of data elements which is common to a variety of applications in the realm of telecommunications, networking and real-time monitoring. 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 [6]. 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 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 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, 4]. Most of these algorithms perform an incremental learning [9] which makes them able to face issues posed by continuous, unbounded, evolving characteristics of data streams. However, a common limitation is that these algorithms are designed to mine data elements which arrive as vectors of fixed attribute values. In many applications, the stream 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 packets in the connection. The structure of these complex data elements can be naturally modeled by several database relations, hence making methods of (multi-)relational data mining [2] more suitable for the discovery of useful patterns from these data streams.

In this work we focus on the novelty (or anomaly) detection task in complex data stream mining. Anomaly detection targets learning algorithms [5, 7, 12] being able to identify unknown situations which represent a change with respect to situations experienced before. We propose 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 data model. New iterations of mining results are built based on old mining results so that the results will not have to be recalculated each time data update is received.

The algorithm is based on the definition of time sensitive data block [5] that is the set of complex data which are periodically (e.g. daily, monthly) added to a stream. The data model is a base of relational patterns (i.e., patterns involving several database relations). Frequencies are computed on the data blocks falling in a sliding time window ending at current time. The time window is defined as a user-defined number of the lastly income blocks. Each time a data block flows in, the pattern base is updated in order to conserve only the patterns which are frequent in at least one block of the time window. Novelty patterns are those patterns in the base whose frequency on the last block significantly changes with respect to an observed "homogeneous" region of frequencies in the remaining blocks of the window.

The paper is organized as follows. Section 2 presents some preliminary concepts and defines novelty patterns. The novelty detection problem is defined in Section 3, while the algorithm that solves the problem is described in Section 4. Section 5 reports the evaluation 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 $\mathbf{a_i}$. Anyway, the 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 objects are stored in 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 properties (or attributes) of objects in S and a number of additional (non-target) relations T_{R_k}, such that 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 define the unit of analysis without being the main subject of 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.

Definition 1 (Unit of analysis). *A unit of analysis $D[s]$ consists of a reference object $s \in T_S$ and all 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 τ be a flow of continuous, consecutive and discrete time points and $\prec$ is a total order relation defined on, a unit of analysis $D[s_i]$ is associated with a time point t_i. A complex data stream is a continuous flow DS of time-stamped units of analysis, that is, $DS = \{\langle D[s_1], t_1 \rangle, \langle D[s_2], t_2 \rangle, \langle D[s_n], t_n \rangle, \ldots\}$, where $t_i \prec t_{i+1}$.*

Time intervals define data blocks in a complex data stream [5].

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. $t_i \in [t-p+1, t]$.*

Henceforth, the number of units of analysis in a basic data block B_i is denoted as $|B_i|$. It is noteworthy that two basic data blocks defined for time intervals of the same length may have a different number of units due to the variable data arrival rate.

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 (see Figure 1).

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 a pair of tables T_j and T_i, with T_j and T_i related by a foreign key FK in H. The name p denotes FK, while the term t (s) is a variable that represents the primary key of T_j (T_i).*

A relational pattern is defined as follows:

Definition 8 (Relational pattern). *A relational pattern P over the schema H is a conjunction of predicates:*

$$p_0(t_0^1), p_1(t_1^1, t_1^2), p_2(t_2^1, t_2^2), \dots, p_m(t_m^1, t_m^2)$$

where $p_0(t_0^1)$ is the key predicate associated with the table T_S and $p_i(t_i^1, t_i^2)$, $i = 1, \dots, m$, is either a structural predicate or a property predicate over H.

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

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

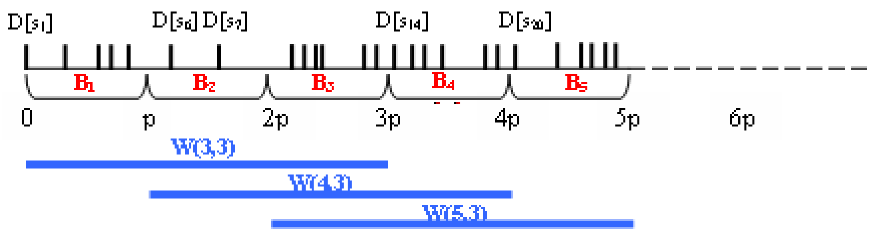

Fig. 1. Data block and sliding time windows in a data stream.

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 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. *$W(i, w)$ be a time window with length w and an ending block B_i, i.e., $W(i, w) = \langle B_{i-w+1}B_{i-w+2}\ldots, B_i\rangle$ be a time window;*
2. *P be a pattern and $s_{i-w+1}, s_{i-w+2}, \ldots, s_i$ is the support of P on 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:*

$$\Theta(s_{i-w+1}(P)) = \ldots = \Theta(s_{i-1}(P)) \neq \Theta(s_i(P)). \quad (2)$$

3 Problem Definition

In this work the novelty detection problem is formulated as the problem of capturing change in evolving data block model of a data stream. Patterns which describe a novelty in the stream are identified within a time based window of basic blocks. Based upon Definition 9, a pattern P can be marked as a novelty pattern on the block B_i if P has "approximately" the same support for all data blocks in time window $W(i, w)$, except for the last one. In this formulation, novelty detection depends on two user-defined parameters: the minimum support (*minSupp*) and the size (w) of the time window. Formally,

Given:

1. a continuous,unbounded stream of time-stamped units of analysis:
$$DS = \{\langle D[s_1], t_1\rangle, \langle D[s_2], t_2\rangle, \ldots, \langle D[s_n], t_n\rangle, \ldots\},$$
where the units of analysis are segmented in data blocks $B_1, B_2, \ldots, B_i$ according to a time based data block size p;
2. a time window size w;
3. a minimum support threshold *minSupp*;

Find the sets $NP_{i,w}$ of novelty patterns associated to the data block B_i based on the time window $W(i, w)$, $i \geq w$.

An algorithmic solution to the problem of detecting novelty patterns in a complex data stream is presented in the next section.

4 The Algorithm

The algorithm Mr-NoDeS (*M*ulti-*R*elational *No*velty *De*tection in Data *S*tream) is a two stepped data stream algorithm. The algorithm is triggered each time a complete data block with time size p arrives in data stream and it is designed to record only the time window formed by the last incoming w data blocks. In the first step, the relational pattern base $M(i, w)$ on the time window $W(i, w)$ is incrementally mined each time a data block B_i arrives, while in the second phase patterns are filtered out in order to keep only those which represent novelty patterns within $W(i, w)$. Details on relational pattern discovery, pattern base maintenance and novelty pattern detections are reported in the next subsections.

4.1 Relational Pattern Discovery

The relational pattern discovery is performed by exploring level-by-level the lattice of relational patterns ordered according to a generality relation ($\geqslant$) between patterns. Formally, 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). Hence, the search proceeds from the most general pattern and iteratively alternates the candidate generation and candidate evaluation phases as in the levelwise method [8]. Candidate generation assumes that the space of pattern is structured according to the θ-subsumption generality order [10].

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 1. Let us consider the relational patterns: (P_1) connection(C). (P_2) connection(C), packet(C,P). (P_3) connection(C), service(C,'http'). (P_4) connection(C), packet(C,P), starting_time(P,8). Then it can be proved that the patterns are ordered as follows: $P_1 \geqslant_\theta P_2$, $P_1 \geqslant_\theta P_3$, $P_1 \geqslant_\theta P_4$, $P_2 \geqslant_\theta P_4$.

θ-subsumption generality order defines a quasi-ordering, since it satisfies the reflexivity and transitivity property but not the anti-symmetric property. This quasi-ordering can be searched according to a downward refinement operator which computes the set of refinements for a relational pattern.

The downward refinement operator ρ' used in this work is defined as follows.

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 $P\}$.*

It can be proved that ρ' is a refinement operator under θ-subsumption, i.e., $P \geqslant_\theta Q$ for all $Q \in \rho'(P)$. The refinement operator ρ' allows for a levelwise 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)[13]. 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 2). In the case q is a structural predicate (i.e., a new relation is introduced in the pattern), $t(g_c)$ 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 (i.e., a superset of an infrequent pattern cannot be frequent) can be 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, where $MaxNumLiterals$ is a user-defined parameter.

4.2 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. Possible operations to maintain the pattern base are inserting frequent patterns, deleting unfrequent patterns or updating the support list of a pattern. We can distinguish between three cases based upon the serial number i of the data block B_i which arrives in the stream.

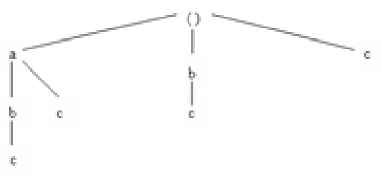

Fig. 2. The enumeration tree over the atoms $A = \{a, b, c\}$ to search the atomsets a, b, c, ab, ac, bc, abc.

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

$(i = 2, \ldots, w)$. Relational patterns which are frequent on B_i are discovered and used to construct M_i^+, where M_i^+ 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.

$(i > w)$. 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 relational 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 relational 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 both adding patterns in M_i^+ and removing patterns in M_i^-. The support list of each pattern $P \in M(i, w)$ is then updated or created from scratch. A list update happens 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 list creation happens when $P \in M_i^+$ and the entire support list $\langle s_{i-w+1}(P), \ldots, s_i(P) \rangle$ is built.

4.3 Time-window based novelty pattern detection

Once data blocks of a time window arrive in the stream $(i \geq w)$, Mr-NoDeS post-processes the pattern base $M(i, w)$ in order to identify novelty patterns according to Definition 9. In this work, function Θ_P is a discretization function defined on the basis of a clustering algorithm. Several clustering algorithms have been designed in the literature. In this paper we use the density-base clustering algorithm DBSCAN [11] which is devised to discover arbitrary-shaped clusters. Clusters are intended as dense areas. Density is estimated in terms of homogeneity of the support values falling in a "timely" area. This method permits to cluster only support values which are similar and observed at consecutive time points. We resort to an implementation of the algorithm that is opportunely adapted to the problem of clustering a timely ordered sequence of support values.

Definition 12. *A cluster is a region of timely consecutive support values whose standard deviation does not exceed a user-defined threshold.*

A top-level description of the clustering algorithm is reported in Algorithm 1. For each pattern P, the cluster construction starts from an arbitrary data block b (seed) and constructs the neighborhood $N(b)$ that includes data blocks in a circle of radius 1 and center b (see Figure 3). The neighborhood is then 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 neighborhood including support values whose standard deviation does not exceed a local a 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 a window of size w.

5 Experiments

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) which has only two values (udp and tcp); the starting time (integer), that is, the starting time of the connection; the destination (nominal), that is, the IP of department public servers; the service (nominal), that is, the requested service (http, ftp, smtp and many other ports); the number of packets (integer), that is, the number of packets transferred within the connection; the average packet time distance (integer), that is, the average distance between packets within the connection; the length (integer), that is, the time length of

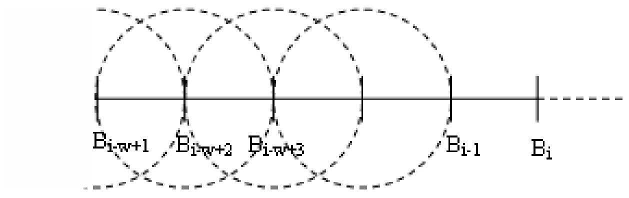

Fig. 3. Density based clustering: neighborhood construction and cluster expansion.

Algorithm 1 Density-based clustering of support values of a pattern over the time window.

```
1: input: $S = \langle s_{i-w+1}, \ldots, s_i \rangle$;
2: output: $C = \langle c_1, \ldots, c_n \rangle$;
3: repeat
4:    $c$=generateClassLabel();
5:    Choose $b$ from $S$ as seed for $c$ ($b$ must be unclassified);
6:    $N(b)$=neighborhood($b$,$S$) ;
7:    if standardDeviation($N(b)$)<$maxStD$ then
8:       label $N(b)$ as $c$;
9:       for $b_i$ labeled as $c$ do
10:         $N(b_i)$=neighborhood($b_i$,$S$);
11:         if standardDeviation($c \cup N(b_i)$)<$localMaxStD$ then
12:            if standardDeviation(c)<$maxStD$ then
13:               Label $N(b_i)$ as $c$;
14:            end if
15:         end if
16:      end for
17:   else
18:      Label $b$ as $c$;
19:   end if
20:   Add $c$ to $C$;
21: until all elements of $S$ are classified
```

the connection; the nation code (nominal), that is, the nation the source IP belongs to; the nation time zone (integer), that is, time zone description of the source IP. The source IP is represented by four groups of tree digits and each group is stored in a separate attribute (nominal). 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 $minsup = 0.1$, $MaxNumLiterals = 5$, $maxStD = 0.02$, $localMaxStD = 0.02$ and $w = 3, 4, 5, 6$. Values of $minsup$, $maxStD$ and $localMaxStD$ are chosen by a tuning process. Results of this process are not reported here due to space limitations. Number of anomaly patterns discovered is plotted in Figure 4. 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

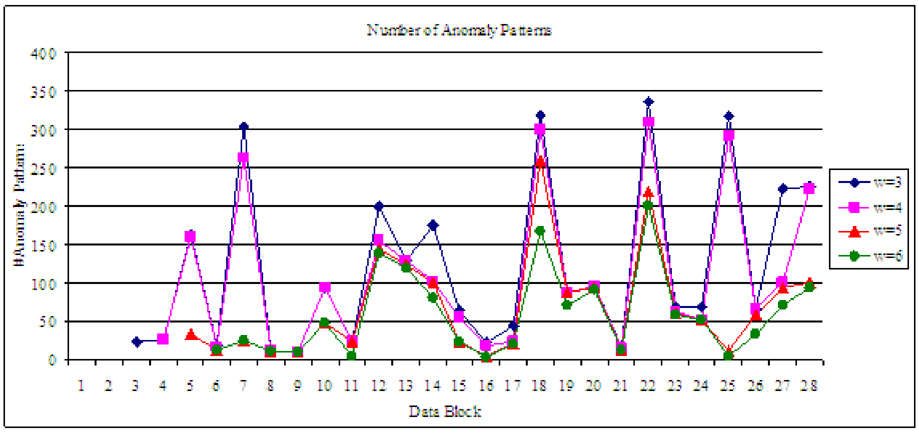

Fig. 4. Number of anomaly patterns discovered on each data block by varying w.

the smoothing of peaks in the number of novelty patterns per 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:

$$conn(C), pack(C, P), proto(C, udp),$$

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. Another example of novelty pattern P_2 extracted on June 20th, 2004 with $w = 5$ is:

$conn(C), pack(C, P), pack\_Pack(P, Q), dist(P, Q, [0.280]), service(C, unknown)$.

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 describes 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 between P and Q that is in the interval [0, 280] ms.

6 Conclusions

In this paper, we face the problem of discovering novelty patterns from stream of complex data elements which are scattered in several relations of a relational database. A relational pattern base is incrementally mined on a time window each time a new data block arrives in the stream and a density based clustering algorithm is employed to detect sharp change of support values. The algorithm is applied to Packet data stream in order to solve a problem of anomaly detection and then support the control activity of a network administrator. As future work we plan to investigate scalability of the algorithm with respect to window size w.

Acknowledgments

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

References

1. H. Blockeel and M. Sebag. Scalability and efficiency in multi-relational data mining. *SIGKDD Explorations Newsletter*, 5(1):17–30, 2003.
2. S. Džeroski and N. Lavrač. *Relational Data Mining*. Springer-Verlag, 2001.
3. M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy. Mining data streams: a review. *SIGMOD Record*, 34(2):18–26, 2005.
4. J. Gama. Issues and challenges in learning from data streams. In H. Kargupta, J. Han, P. Yu, R. Motwani, and V. Kumar, editors, *Book chapter In Data Mining and Knowledge Discovery Series on Next Generation of Data mining*, pages 209–222. Chapman and Hall, CRC Press, Taylor and Francis Group, 2009.
5. V. Ganti, J. Gehrke, and R. Ramakrishnan. Mining data streams under block evolution. *SIGKDD Explorations*, 3(2):1–10, 2002.
6. S. Guha, N. Koudas, and K. Shim. Data-streams and histograms. In *In the 33thannual ACM Symposium on Theory of Computing, STOC 2001*, pages 471–475, New York, NY, USA, 2001. ACM.
7. J. Ma and S. Perkins. Online novelty detection on temporal sequences. In *In the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003*, pages 613–618. ACM, 2003.
8. H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.
9. T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
10. G. D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5:153–163, 1970.
11. J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Min. Knowl. Discov.*, 2(2):169–194, 1998.
12. E. J. Spinosa, A. P. d. L. F. de Carvalho, and J. Gama. Cluster-based novel concept detection in data streams applied to intrusion detection in computer networks. In *In the Symposium on Applied Computing, SAC 2008*, pages 976–980. ACM, 2008.
13. X. Zhang, G. Dong, and R. Kotagiri. Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets. In *Knowledge Discovery and Data Mining*, pages 310–314, 2000.