Discovery of Events with Negative Behaviour Against Given Sequential Patterns

Discovery of events with negative behavior against given sequential patterns

Fahad Anwar, Ilias Petrounias, Tim Morris
University of Manchester
Manchester, UK
Fahad.Anwar@manchester.ac.uk
Ilias.Petrounias@manchester.ac.uk
Tim.Morris@manchester.ac.uk

Vassilis Kodogiannis
University of Westminister
London, UK
V.Kodogiannis@westminister.ac.uk

Abstract— **The dramatic drop in the prices of data collection and storage devices has not only enabled organisations to store almost every activity of their business processes, they can also retain every state of these activities as well. Availability of these masses of data also means that by implementing different data mining techniques we can yield more accurate and useful information to be used for important decision making. One of the key mining techniques on such data is to discover sequential patterns. Most of the existing sequential pattern mining approaches mainly deal with finding the positive behaviour of a sequential pattern that can help in predicting the next event after a sequence of events. In this paper we propose the concept of Negative Behaviour Against the Sequential Pattern (NBASP) that is to discover the events/event-sets which are unlikely to follow the given sequential pattern and discuss its applications in a variety of domains. A comprehensive problem definition and efficient algorithm to discover NBASP is presented.**

Keywords-component; data mining, post mining environment, sequential pattern

I. INTRODUCTION

One of the key desires of humanity is to access the knowledge, which can help them in predicting the future. As the thinking process evolves, it has become clearer to us that predictions based upon some facts are more accurate and useful than mere guesses not based on any real existing information. This combination of facts (data) and extraction of knowledge from these facts are the main focus of the knowledge discovery process. Data mining includes a wide range of knowledge discovery activities, such as association rules, classification and clustering. Based on those techniques Web mining and sequential pattern mining are also well researched. Most of the previous work presented in the area of sequential patterns is based upon finding the sequential patterns having "Positive Behaviour", i.e. they predict what will be the next event/event-set in a sequence. For example, "Customers who purchase a computers are likely to buy printers within the next 30 days", (computer→printer, "30 days") is a sequential pattern with positive behaviour. However, an aspect of sequential patterns which can be very helpful in decision support systems has not been fully explored. This is the discovery of item/event having negative behaviour against a given sequential pattern.

II. APPLICATION IN DIFFERENT DOMAINS

Discovery of the negative behaviour against a sequential pattern can be beneficial in different domains i.e. investigation of diseases for medical treatment, designing the flow of web sites, decision support systems in financial sectors, challenging the perceptions of domain experts, fraud detection in networking, telecommunication monitoring and designing outlooks.

A. Medical Treatment

Events having negative behaviour against given sequential pattern can be helpful to medical practitioners, where patients during the course of a (suspected) disease do not follow an expected pattern. At the same time it is useful when a disease changes and new patterns of symptoms may be established. Below are some examples where the finding of negative behaviour events can be useful:[1]

- A patient has bone pain, swelling on knee, swelling on wrist and fingers. If the doctors know that these symptoms are not going to lead to hypertension they can give corticosteroid medicines although they have side effects against hypertension.

- An elderly patient has pain in the bones and weight loss. The doctor may think he/she has rheumatoid disease but after a week the patient comes up with little pain on the upper right abdomen. If the doctors know that it is highly unlikely for pain on the right upper abdomen to follow the above mentioned 3 symptoms they will start investigating the disease again and may find out that the patient has cancer and bone metastasis.

978-1-4244-5164-7/10/$26.00 ©2010 IEEE

- Patient has palpitation, perspiration, loss of weight, T3 and T4 levels are high and TSH level is below normal. These symptoms and findings suggest over function of thyroid glands. The doctor needs to check the thyroid USG and scintigraphy to confirm. It seems less likely to have any symptoms of renal or liver disease (like pain on the right upper abdomen, jaundice). Therefore, there is no need to check renal or liver functions.

B. Financial Sector

In the financial sector, information/predictions of which event is not going to follow after a sequence of events can help in making important decisions. For example, if there is a sharp decline in stock prices of Microsoft and Oracle, there might be a perception in stock market that stock prices of other IT related companies will decline as well. However, if we know that this sequence of events will not be followed by a decline in stock prices of ebay.com, this will save shareholders of ebay.com from selling of their shares in panic under the wrong perception.

C. Web Site Flow

Discovery of negative behaviour events against a sequential pattern can be useful in challenging the perceptions of system designers/domain experts. System designers may have developed knowledge based on prior experiences over many years, whereas, data reflect patterns on current conditions/environments which are different from previous environments. Let us take an example of an E-Commerce Web site designer who perceived the following flow of a web site (as depicted in Figure 1)

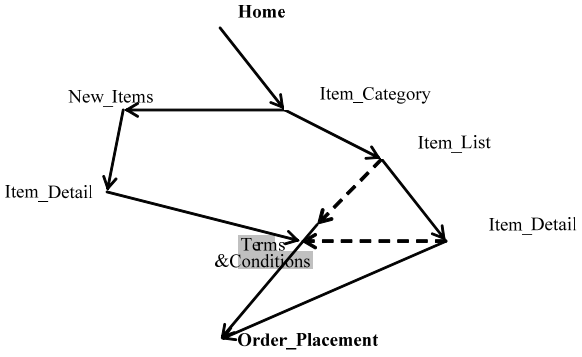

Figure 1. Web site flow diagram

If after finding the negative behaviour against the sequence of events (Home→Item_Category→Item_List) flow, the result shows that Web users, who follow this sequence pattern may go to pages (Item Detail, Order Placement). However, they are unlikely to navigate to the "Terms & Condition Page" although a sufficient number of people access this page but from a different route (Home→New_Items→Item_Detail). Doted lines in Figure 1 show that Web users are not navigating the site according to the system designer's perception. The discovered negative behaviour events (NB_E) against the perceived site flow challenge the perception of system designers. They prompt him to redesign the flow of the site, so Web users should go to "Terms & Conditions Page"; since the "Terms & Conditions Page" contains important information.

D. Customer Buying Cards

In recent years there has been an increasing trend of issuing combined customer buying cards by different stores. These cards can be used by customers to buy items from multiple stores. Finding the negative behaviour from the customer database can be useful for decision makers of these stores in order to find out which combination of stores should be avoided while introducing combined customer buying cards. For example if decision makers of SAINSBURY stores know that customers who buy items from TESCO and next week from ASDA are unlikely to buy items from SAINSBURY stores later on they can avoid introducing combined customer buying cards with these stores.

III. RELATED WORK

Mining sequential patterns is one of the most extensively researched areas in the data mining research community. Agrawal and Srikant first introduced the problem of discovering sequential patterns in 1995 [2]. They proposed three different algorithms: AprioriAll, AprioriSome and AprioriDynamic. Since then, a lot of work and improvements to the original algorithm have taken place. Most of the previous work presented in the area of sequential patterns is based upon finding the frequent sequential patterns having "Positive Behaviour", i.e. However, an aspect of sequential patterns which can be very useful in variety of domain has not been fully explored. That aspect is to find the events which have negative behaviour (NB_E) against the given sequential pattern, i.e. discovering the events which are unlikely to follow a sequence of events. There have been some research efforts to mine both positive and negative association rules together [3-5]. However they ignore the temporal aspects of events. The work presented in [1, 6], concentrated on discovering the negative conclusion for a given sequential pattern in post mining environment, which is similar to our concept of discovery of NB_{ASP}; however, these approaches discover the items/item-sets having negative conclusion for only one given sequential pattern at a time, whereas, the algorithm presented in paper can discover NB_E against multiple sequential patterns simultaneously. Confronting multiple sequential patterns simultaneously increases the complexity of NB_E discovery process considerably. The complexity of the algorithm is based upon the fact that the suggested algorithm has to find the existence of each given

sequential pattern within the specific sequence duration. Therefore, the process of searching the existence of each sequential pattern and then discovering events with negative behaviour can span into multiple search spaces, with each given sequential pattern having its own different search spaces to be confronted.

IV. PROBLEM DEFINITION FRAMEWORK

One of the most important ingredients of any data mining application is the provision of a flexible and comprehensive problem definition framework in which users can express the problem statement comprehensively and easily. In a broader view, the problem of discovering events which have negative behaviour against the all the given sequential patterns can be defined as:

$$NB_{ASP} = SP \rightarrow \sim Event$$

We are given a time-stamped database of events *D* over a time domain *T* and a set of sequential patterns SP_{SET} and a list of candidate negative behaviour events (NB_E) ($^{CL}NB_E$). The goal is to discover all NB_E with reference to each given sequential pattern. $^{CL}NB_E$ can be all the events in the database or a user-defined list of events.

$$< SP_{SET}, {}^{CL}NB_E >$$
$$SP_{SET}=\{SP_1, SP_2, SP_3 \ldots.. SP_n\}$$
$$^{CL}NB_E = \{ e_1,e_2 \ldots e_n \}$$

As we are dealing with two concepts, sequential pattern and NB_E, the suggested problem definition framework needs to be flexible enough to define both these concepts comprehensively and easily. Let us first discuss the frequent sequence of events (sequential patterns) and see how we can define the problem of searching for all instances of given sequential patterns.

A sequential pattern can be described as a set of event $SP=(e_1,e_2,e_3 \ldots e_n)$ with temporal/sequential ordering satisfying the sequence duration (SD). SD is the time limit during which the sequential pattern must exist.

$$SP_{SET}=\{(SP_1, SD_1), (SP_2, SD_2), \ldots.. (SP_n, SD_n)\}$$

For a better understanding of problem definition of NB_E we introduce three user-defined parameters, namely: Time Period (TP), Maximum Time Interval (MT_{ITVL}) and Maximum Tolerance (M_{TOL}).

A. Time period (TP)

It is quite possible that the user is very much interested in discovering NB_E against a given sequential patterns which may only exist during a particular time period rather than in the complete time spectrum. Hence the user-defined parameter TP can be used to reflect this concept. TP represents the time period during which NB_E needs to be discovered. For example, we can say an event *E* is NB_E against sequential pattern *X* during the time period of 1st Jan 2009 to 30th March 2009. This information can not only reduce the extra burden on the mining process by concentrating only on the user-defined segmented event database, but TP can also be used to observe the changes in the existing patterns by comparing the mining results based upon different time-periods.[7]

$$<TP,\ SP_{SET},\ {}^{CL}NB_E>$$

B. Maximum time interval (MT_{ITVL})

The maximum time interval (MT_{ITVL}) is a time interval after the sequential pattern during which we have to find the presence of NB_E. This parameter is imperative since we are not interested in an event which is unlikely to follow the given sequential pattern after a relatively long period of time. As each sequential pattern nature can be different therefore MT_{ITVL} needs to be defined for each given sequential pattern.

$$<TP,\ SP_{SET},\ {}^{CL\ CL}NB_E>$$
$$SP_{SET}= (SP_n, <SD,GR>, <MT_{ITVL},GR>)$$

C. Maximum tolerance (M_{TOL})

NB_E against a given sequential pattern mean that they have no or very limited existence between the end time of the sequential pattern and the end time of the user-defined parameter of MT_{ITVL}. The user-defined parameter of M_{TOL} is introduced here to determine the maximum presence allowed for a detected event in order to be considered as an NB_E against the given sequential pattern.

$$<TP,\ SP_{SET},\ {}^{CL}NB_E\ ,\ M_{TOL}>$$
$$SP_{SET}= (SP, <SD,GR>, <MT_{ITVL},GR>)$$

V. V. DISCOVERY PROCESS

The discovery process of NB_E is consists of three main phases; Data Pre-Processing, Searching the instances of multiple sequential patterns and Discovery of the presence of $^{CL}NB_E$ (Figure 2). The last two phases work alternately. The algorithm finds the instance of a given sequential pattern in the data-sequence. Then the algorithm passes the searched sequential pattern instance end time along with a pointer to $^{CL}NB_E$ and user-defined parameters of MT_{ITVL} and M_{TOL} to the next phase in which it attempts to discover the support of all $^{CL}NB_E$ during the user-defined time limit (MT_{ITVL}).

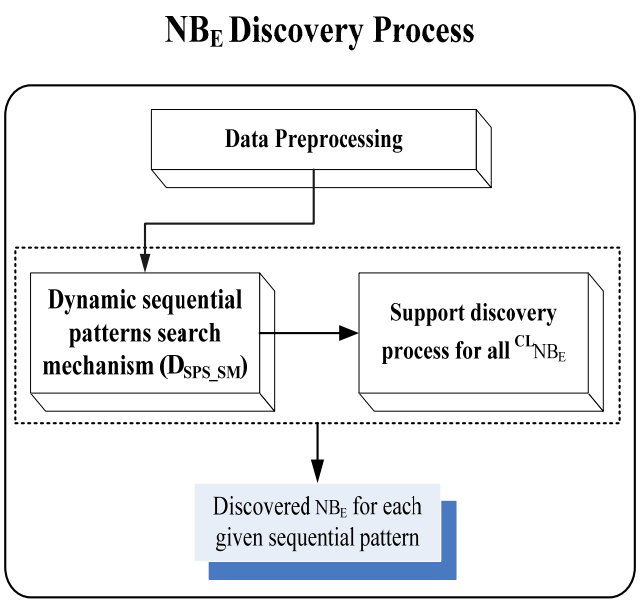

Figure 2. **Discovery Process**

A. Data pre-processing

The data is filtered according to user-defined parameter of Period Time (TP). For example if TP is defined as 1st Jan, 2009 to 30th March, 2009 then all the events detected during this period will be filtered out for on-word data processing. Next, we convert the transaction data-set into a sequence database. For this, we sort the database on Sequence_ID and transaction date/time and number of events in each data sequence. Sorting the sequence database on the number of events in each data sequence (descending order) will help algorithm work faster. This is because scanning the longer data sequence first will help in eliminating the events having positive behaviour quickly**.**

B. B. Dynamic sequential patterns search mechanism (D_{SPS_SM})

Any sequential pattern can exist multiple times during each data sequence (SD); therefore it is likely that the algorithm will have to find multiple instances of SP in each SD. As the algorithm tries to discover NB_E against multiple sequential patterns, it increases the complexity of searching all the instances of each sequential pattern in one database scan. The complexity of the algorithm is based upon the fact that the suggested algorithm has to find the instances of each given sequential pattern within the specific sequence duration. Therefore, the process of searching the instances of each sequential pattern can span into multiple search spaces, with each given sequential pattern having its own different search spaces to be confronted. We call these search spaces search windows (SW). By SW we mean the limited search space in which we have to find the instance of each given sequential pattern. For example, if we have three given sequential patterns SP_1, SP_2, SP_3 with associated SD of 3, 8, 5 minutes respectively and the data-sequence length is 10 minutes, we will have the following multiple SWs for each given sequential pattern (Table 1) .

SP	Search Windows	SW#
1	{(1-3),(2,4),(3-5), (4-6), (5-7), (6-8), (7-9), (8-10)}	8
2	{(1-8),(2,9),(3-10)}	3
3	{(1-5),(2,6),(3-7), (4-8), (5-9), (6-10)}	6

Table 1 Search windows for each sequential pattern

In light of the above mentioned challenges, it is important to formulate a mechanism which can minimise the expensive process of searching for instances of multiple sequential patterns in each data sequence. To confront the above mentioned complexities, we introduce a Dynamic sequential patterns search mechanism (D_{SPS_SM})**.** D_{SPS_SM} facilitates the discovery of all NB_E against all the given sequential patterns in a single database scan and optimises the NB_E discovery process considerably. To elaborate the concept of D_{SPS_SM}, we first need to discuss how we will perceive the status of sequential pattern events during the D_{SPS_SM} and what we mean by a valid discovered event.

In D_{SPS_SM} the given sequential pattern events are divided into three statuses:

- Current Event (C_{Event}) C_{Event} is a sequential pattern event for which D_{SPS_SM} is currently searching the support.
- Left Hand Side Events (E_{LHS}) All the events of the given sequential pattern which are on the left side of C_{Event} are known as E_{LHS}.
- Right Hand Side Events (E_{RHS}) All the events of the given sequential pattern which are on the right side of C_{Event} are known as E_{RHS}.

For an event to be considered as a valid discovered event, it has to satisfy the following conditions:

- The event time should be within the boundaries of the current search window (C_{SW}).

$$C_{SW(ST)} \geq E_T \leq C_{SW(ET)}$$

- The event time should be greater than the last discovered left hand end event (E_{LHS}).

$$E_T > E_{LHS\_LDT}$$

The main idea of D_{SPS_SM} is to explore the sequence

duration property of the given sequential patterns and utilise its nature during the search process of all the given sequential patterns in single database scan. Moreover, D_{SPS_SM} also utilises the temporal/sequential nature of sequential pattern to scan only the minimum required data-set from the event database during the search process of given SPs. In the following sub-sections we will discuss both these concepts in more detail.

1) SD properties for different sequential patterns

Due to the sequence duration (SD) property of sequential patterns, the SP search process needs to be divided into different search windows (SWs). Each sequential pattern can have different SD (which means a different size and number of SWs for each sequential pattern). This enhances the complexity of discovering support of all sequential patterns in one database scan. However, the following properties of SDs enable us to utilise the scanning result of each SW for multiple sequential patterns, which reduces the burden of database scanning considerably.

- SD Property 1: The first search window (^{F}SW) of the given SP which has the largest SD overlaps all the first SWs of all other given sequential patterns (Figure 3(a)).

- SD Property 2: The sequential pattern which has the largest *SD* may overlap multiple SWs of other SPs with shorter SD as shown in Figure 3(b) and (c).

- SD Property 3: All search windows (SWs) can expand with single granularity interval, for example if the sequence duration granularity is defined as minutes then all SWs can expend with a single minute (Figure 3(d)).

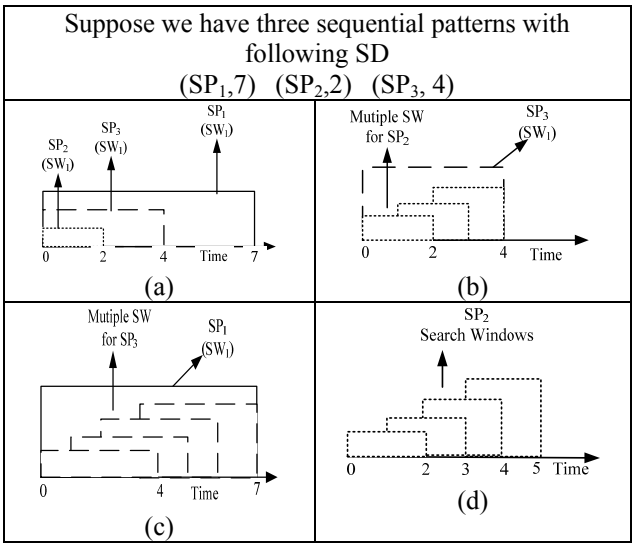

Figure 3. Multiple search windows with SD properties

2) Sequential pattern temporal/sequential properties

During the sequential pattern search process a significant number of different SWs overlap each other. Therefore, the scanned result of the overlapped SW portion can be utilised during the search process of sequential pattern instances in the next SW. This property enables the algorithm to expand the search dynamically; that means it will only search the next event of SP in C_{SW} if it is required. To utilise the search results of the previous SW, D_{SPS_SM} utilises the following temporal/sequential properties of sequential patterns:

- SP Temporal/Sequential Property 1: If D_{SPS_SM} does not find the support for the current event (C_{Event}) of a given sequential pattern then all the right hand side events (E_{RHS}) have no significance in the current search window (C_{SW}). This is due to the fact that for a valid support of a given SP, every event has to be discovered within the $C_{SW.}$ Hence, the process of searching E_{RHS} in C_{SW} will be terminated and D_{SPS_SM} moves to the next search window (N_{SW}). Suppose we have a sequential pattern SP= (A→C→M→Y→C→B) and support of the event A and C is discovered in $C_{SW,}$ however the support of event M is not discovered then there is no need to search the remaining events (YCB) in C_{SW} and D_{SPS_SM} moves to the $N_{SW.}$

- SP temporal/sequential property 2: If an already first discovered left hand side event (E_{LHS}) from the previous search window (P_{SW}) is valid in C_{SW} then all previously discovered E_{RHS} of P_{SW} are valid in C_{SW} as well.

C. Data Structure

By utilising the above mentioned properties of sequential patterns and the temporal/sequential nature of a given sequential pattern, the proposed algorithm can search all sequential patterns instances with only one database scan. To accomplish this, the algorithm needs to have a data structure which can hold the status of all events during the SP support discovery process (this is because we want to utilise the already discovered events of different sequential pattern). The D_{SPS_SM} processes the data with the following data structure elements.

- Sequential pattern information ($SP_{\mathbf{Info}}$): This data structure is used to hold information about all the sequential patterns for which support needs to be searched. The $SP_{\mathbf{Info}}$ is a mainly static data structure (populated at the start of D_{SPS_SM} discovery process and only the SP^{SP} and SP^{EP} fields are updated once an instance of a specific SP is found

- Search window information ($SW_{\mathbf{Info}}$): This data structure is used to hold information about SWs of each sequential

pattern. The SW_{Info} is updated according to the need of new SWs during the search process of all given sequential pattern.

- Sequential pattern events information ($SPE_{\mathbf{Info}}$): SPE_{Info} is a vertical transformation of the original database for unique events of all the given sequential patterns. SPE_{Info} expands dynamically as the process shifts to the next search window

- Current search event (C_{SE}): C_{SE} holds the list of events for all the given SPs which need to be discovered next. C_{SE} basically informs D_{SPS_SM} which events need to be searched next in the C_{SW}

- SP discovered events information ($SP_{\mathbf{DE}}$): For each given sequential pattern a separate SP_{DE} is created. SP_{DE} holds the position of all the events which have been discovered so far for that specific SP. Once all the events of a sequential pattern are searched then SP_{DE} is reset to its initial state with no searched event

D. Support discovery process for all $^{CL}NB_E$ *(*NB_{E_DS}*)*

Once D_{SPS_SM} finds the instance of a specific given SP, it passes the end time/position of discovered SP support to NB_{E_DS} along with user-defined parameters of M_{TOL} and MT_{ITVL}. The main purpose of NB_{E_DS} is to count the support of all the candidate events given in $^{CL}NB_E$ during the limited time space (based on user-defined parameter of MT_{ITVL}). We call this time space the interested time (I_{TIME}); which can be defined as follows:

Interested Time (I_{TIME}) is a time interval between SP_{ET} (a searched sequential pattern's instance end time) and MT_{ITVL_ET} (the end time of the user-defined parameter of MT_{ITVL}). For each sequential pattern instance found there will be a specific I_{TIME}. I_{TIME} is always equal to or greater than the end time of the sequential pattern instance found in the data-sequence and less or equal to the MT_{ITVL_ET}. Since each given sequential pattern has its own MT_{ITVL} and their SP_{ET} might be different, therefore NB_{E_DS} may have to deal with different I_{TIME} simultaneously.

Just like search windows of different sequential patterns, I_{TIME} of different SPs can also overlap each other. Due to this property of I_{TIME} we can utilise the support discovery of $^{CL}NB_E$ between I_{TIME}s of different SPs. In Figure 4 I_{TIME} of SP_2 overlaps the I_{TIME} of SP_1 SP_3 and SP_4. Therefore, events support counted in I_{TIME} of SP_2 can also be used for SP_1, SP_3 and SP_4.

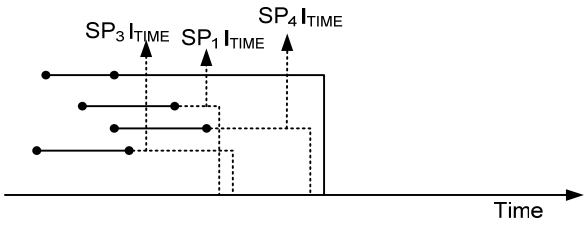

Figure 4. Overlapping ITIME'S for different SP's

Based on the NB_E discovered in all data sequences the proposed algorithm then calculates the average support (as a percentage) of $^{CL}NB_E$ events against each sequential pattern and outputs the discovered negative behaviour events against each sequential pattern. That is to identify the events with average support less than the user defined parameter of M_{TOL}.

VI. CONCLUSION

In the paper, we proposed the concept of events having negative behaviour against the sequential pattern (NB_{ASP}), which has applications in a variety of domains. We first proposed a comprehensive and flexible problem definition framework and then presented a data mining framework to discover all the events having negative behaviour against all the given sequential patterns in one database scan. To confront the process/memory expensive process of searching all the instances of multiple sequential patterns in each data sequence a dynamic sequential pattern search mechanism (D_{SPS_SM}) is also presented.

REFERENCES

[1] F. Anwar and I. Petrounias, "An Algorithm for Identifying Patient's Negative Behaviour against a Sequence of Symptoms," in First East European Conference on Health Care Modeling and Computation (HCMC 2005) Craiova, Romania, 2005.

[2] R. Agrawal and R. Srikant, "Mining Sequential Patterns," in Eleventh International Conference on Data Engineering, Taipei, Taiwan, 1995, pp. 3-14.

[3] A. Maria-Luiza and R. Z. Osmar, "Mining positive and negative association rules: an approach for confined rules," in Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, 2004, pp. 27-38.

[4] C. Cornells, Y. Peng, Z. Xing, and C. Guoqing, "Mining Positive and Negative Association Rules from Large Databases," in IEEE Conference on Cybernetics and Intelligent Systems, 2006, pp. 1-6.

[5] W. Xindong, Z. Chengqi, and Z. Shichao, "Efficient mining of both positive and negative association rules," ACM Transactions on Information Systems (TOIS), vol. 22, pp. 381-405, 2004.

[6] P. Kazienko, "Mining Sequential Patterns with Negative Conclusions," in Proceedings of the 10th international conference on Data Warehousing and Knowledge Discovery (DaWaK '08) Turin, Italy: Springer-Verlag, 2008, pp. 423-432.

[7] F. Anwar, I. Petrounias, V. S. Kodogiannis, V. Tasseva, and D. Peneva, "Efficient periodicity mining of sequential patterns in a post-mining environment," in 4th International IEEE Conference on Intelligent Systems, (IS '08). vol. 2, 2008, pp. 2-11.