



Mining anomalous events against frequent sequences in surveillance videos from commercial environments

Fahad Anwar^a, Ilias Petrounias^{b,*}, Tim Morris^c, Vassilis Kodogiannis^d

^a Wolfson Molecular Imaging Centre, University of Manchester, Manchester M20 3LJ, UK

^b Manchester Business School, University of Manchester, Manchester M15 6PB, UK

^c School of Computer Science, University of Manchester, Manchester M13 9PL, UK

^d School of Electronics and Computer Science, University of Westminster, London W1W 6UW, UK

ARTICLE INFO

Keywords:

Knowledge discovery
Data mining
Sequential pattern mining
Periodicity mining
Surveillance videos
Business intelligence
Video mining
Anomalous events mining

ABSTRACT

In the UK alone there are currently over 4.2 million operational CCTV cameras, that is virtually one camera for every 14th person, and this figure is increasing at a fast rate throughout the world (especially after the tragic events of 9/11 and 7/7) (Norris, McCahill, & Wood, 2004). Security concerns are not the only factor driving the rapid growth of CCTV cameras. Another important reason is the access of hidden knowledge extracted from CCTV footage to be used for effective business decision making, such as store designing, customer services, product marketing, reducing store shrinkage, etc.

Events occurring in observed scenes are one of the most important semantic entities that can be extracted from videos (Anwar & Naftel, 2008). Most of the work presented in the past is based upon finding frequent event patterns or deals with discovering already known abnormal events. In contrast, in this paper we present a framework to discover unknown anomalous events associated with a frequent sequence of events (A_{EASP}); that is to discover events, which are unlikely to follow a frequent sequence of events. This information can be very useful for discovering unknown abnormal events and can provide early actionable intelligence to redeploy resources to specific areas of view (such as PTZ camera or attention of a CCTV user). Discovery of anomalous events against a sequential pattern can also provide business intelligence for store management in the retail sector. The proposed event mining framework is an extension to our previous research work presented in Anwar et al. (2010) and also takes the temporal aspect of anomalous events against frequent sequence of events into consideration, that is to discover anomalous events which are true for a specific time interval only and might not be an anomalous events against frequent sequence of events over a whole time spectrum and vice versa. To confront the memory expensive process of searching all the instances of multiple sequential patterns in each data sequence an efficient dynamic sequential pattern search mechanism is introduced. Different experiments are conducted to evaluate the proposed anomalous events against frequent sequence of events mining algorithm's accuracy and performance.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

The proliferation of TV channels and video-based surveillance systems has enabled us to store almost every activity that mirrors our world. This generates huge volumes of data, too much for human operators to process; therefore there is a great need for automated multimedia content analysis (Norris, McCahill, & Wood, 2004). Motivated by the success of sequential pattern mining approaches in analysing transactional data, a significant amount of

research effort has been devoted to apply these techniques to multimedia data for unearthing hidden interesting information. In multimedia surveillance videos the sequential patterns are frequent sequences of events with temporal/sequential ordering. We can define the sequential pattern as below.

A sequential pattern is a set of events $SP = (e_1, e_2, e_3 \dots e_n)$ with temporal and sequential ordering having more representation in a database than the user-defined parameter of min_supp (minimum support) (Agrawal et al., 1995). Support for a sequential pattern is the fraction of data-sequences from the database supporting the given sequential pattern

$$SP = \langle e_1, e_2 \dots e_n, min_supp \rangle$$

$\langle \text{"Car in entrance area"} \rightarrow \text{"Car in parking road"} \rightarrow \text{"Car turn left"} \rangle, 60\%$

* Corresponding author.

E-mail addresses: Fahad.Anwar@manchester.ac.uk (F. Anwar), Ilias.Petrounias@manchester.ac.uk (I. Petrounias), Tim.Morris@manchester.ac.uk (T. Morris), V.Kodogiannis@westminster.ac.uk (V. Kodogiannis).

The significance of the above mentioned sequential pattern depends on the time it takes to complete the given sequential pattern. This feature is defined by the user-defined parameter of sequence duration (SD). Therefore, for a sequential pattern to be supported by a data-sequence it has to occur within the user-defined sequence duration.

$$SP = \langle e_1, e_2 \dots e_n, min_supp, SD \rangle$$

$\langle \text{"Car in entrance area"} \rightarrow \text{"Car in parking road"} \rightarrow \text{"Car turn left"} \rangle, 60\%, 2 \text{ minutes} \rangle$.

Most of the work presented in the past was based upon finding sequential patterns to provide a hierarchical structure of events for video retrieval and indexing. Research into surveillance videos mainly deals with discovering already known abnormal events. In contrast, the work presented in this paper is focused on the problem of discovering unknown anomalous events against known sequential patterns (A_{EASP}). That is to discover events that are unlikely to follow a frequent sequence of events. This information can be very useful for discovering unknown abnormal events and can provide early actionable intelligence to redeploy resources to specific areas of view (such as PTZ camera or the attention of a CCTV user). Discovery of anomalous events against a sequential pattern can also provide business intelligence for store management in the retail sector. The importance of anomalous events against a frequent sequence of events can be seen from the following examples.

Suppose we have the frequent sequence of events: 'a vehicle on the road' \rightarrow 'a vehicle on the parking road' \rightarrow 'a vehicle in the parking place' (sequence duration 03 minutes). This sequence becomes more interesting if it is followed by an unlikely event such as 'vehicle on the parking road again', this might be a hit and run accident. Therefore, the attention of the CCTV user needs to be diverted to the specific video stream. It is important to note that an anomalous event isolated from the specific sequence of events could be just a normal event (Fig. 1).

In a retail store environment, suppose we have following frequent sequence of events: 'section D is crowded' \rightarrow 'high customer activity in section A' \rightarrow 'long queues on tills 05 and 06'. Although this information provides business intelligence to depute resources to tills 05 and 06, it does not provide information regarding what resources can be spared to supply tills 05 and 06. But, if we had additional information that the above sequence of events is unlikely to follow 'high activity on tills 1 & 2' then information about the anomalous event can be used for effective store management (Fig. 2).

In a till scanning process at a retail store a well known sequential pattern of events is (scanning of all the items \rightarrow End Transaction \rightarrow Total). This normal frequent sequence of events will become abnormal (an interesting event) if it is followed by event (scanning an item), as it can be an attempted fraud. Here, again it is important to note that "scanning an item" is very much a normal event in isolation to the above mentioned sequential pattern of events.

We can define the A_{EASP} as an event that is unlikely to follow a specific frequent sequence of events (a sequential pattern).

Definition 1

$A_{EASP} = SP \rightarrow \sim \text{Event}$.

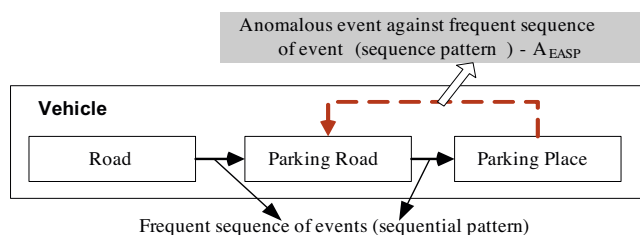


Fig. 1. Anomalous event against frequent sequence of events.

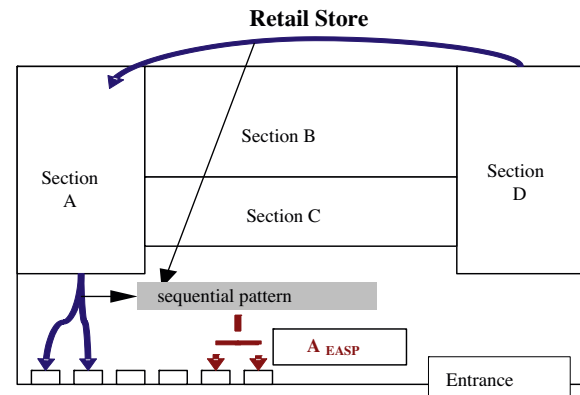


Fig. 2. Anomalous event against a frequent sequence of event.

The remaining of the paper is structured as follows. Section 2 presents related work and identifies how it differs from the work presented in this paper. In Section 3 a comprehensive problem definition framework is presented. Section 4 focuses on the discovery process of anomalous events against the given sequential patterns, which is followed by algorithm implementation and validation in Section 5. Section 6 summarises the research efforts carried out in this paper, followed by a discussion on future work. Section 7 lists of all the abbreviations in presentation order for the ease of reader. A list of the abbreviations and acronyms used in the paper can be found in Appendix.

2. Related work

When we see a video, we are basically interested in the visual and audio parts of the video; such as one or more entities in the video that are acting, or some sound that is created by some entities either saying or making some sounds. Moreover, when something is written on the screen, we are interested in the textual part of the video as well. All of these visual, audio and textual strings form multimedia events. The main focus of multimedia event mining approaches is to apply data mining techniques to explore the characteristics and relationships of these different multimedia contents and to discover interesting events in multimedia data (Öztaarak and Yazici, 2006).

Due to the ever-growing number of installations of visual surveillance systems to monitor traffic flow on roads many researchers have developed methods to detect events in such videos. The main objectives of these effects are to yield unknown knowledge, such as vehicle classification/identification, traffic flow and the spatio-temporal relationships of different objects in the field of view. The techniques presented in Cucchiara, Piccardi, and Mello (2000), Dailey, Cathey, and Pumrin (2000), Huang et al. (1994), Kamijo, Matsushita, Ikeuchi, and Sakauchi (2000) and Dieter et al. (1994) follow a two-step process; firstly objects from the videos were segmented through low level vision algorithms. Then the behaviour of these objects was analysed to gain information for important decision marking. Most of the work to analyse traffic video in the past was based on low level feature extraction. However, in recent years researchers have focused on unsupervised image segmentation and object modelling to capture spatio-temporal relationships among objects as well (Shu-Ching et al., 2000, 2001a, 2001b). Shu-Ching et al. in Chen et al. (2001) proposed a framework to discover and capture the spatio-temporal relationships among vehicles through unsupervised video segmentation and tracking. Multimedia augmented transition network (MATN) and multimedia input strings were used to model these

relationships and explore the hidden information, such as accident events, vehicles making a U-Turn, etc. Fig. 3 provides the MATN and multimedia string based modelling example. Here, G represents a target object and C means car. Hence, $G_1 \& C_{13} \& C_{10}$ represents that the first car is on the top left of G and C_{10} means second car is on the left of the target object/vehicle.

Oh and Bandi (2002) identify the importance of data mining activity for video indexing and propose a framework for indexing unstructured video contents. In the first step of their proposed indexing framework a background frame is extracted from a given sequence for preprocessing and its colour histogram is computed. Then for each subsequent frame the foreground region is identified via background subtraction. Next, these frames are categorised according to the level of motion detected in them. In the last step temporal information is used to index frames within each category. This process provides a hierarchical structure from a sequence based on these categories, which are not independent from each other. Work presented in Oh and Bandi (2002) was later extended in Jung-Hwan et al. (2003) by introducing a technique for automatic measurement of the overall motion in not only two consecutive frames but also of the entire shot (collection of frames). In their proposed technique, the location of motion in each frame is used for better categorisation of video contents. The Average Motion Matrix (AMM) was calculated for each shot, then a multi-level hierarchical clustering approach to group segments in terms of category and motion of segments is implemented. The algorithm is implemented in a top-down fashion, where the feature category is utilised at the top level. In other words, the algorithm groups segments into K_1 clusters according to the categories. Ghanem et al. proposed a system for mining surveillance video in Ghanem et al. (2004). The main focus was to define a high level query language for reasoning about spatial and temporal relations of background regions and moving entities, and about human activities. Moreover another useful contribution was to provide a powerful graphical interface where users can formulate visual queries.

In Turaga et al. (2007) a linear dynamic system (LDS) is presented on optical flow features for surveillance action events. The LDS iterates between model learning and sequence segmentation. Their proposed algorithms learn the model parameters from a video stream and then segment a single video sequence into different clusters where each cluster represents an event; moreover they also present a technique to build affine, view and rate invariance of the activity into the distance metric for clustering. In Hamid et al. (2005, 2007) n -grams and suffix trees are used to mine movement patterns from the CCTV footage collected using ceiling-

mounted cameras. They proposed the representation for an activity as bags of event n -grams that can capture the global structure of an activity using its local event statistics. The detected patterns enable the detection of frequent events such as “Fedex delivery” as well as anomalous events such as “truck driving away with its back door open”. Petrushin in Valery (2005) proposed an effective approach to detect frequent and rare events in a multicamera surveillance environment by using multilevel self organizing map (SOM) clustering on the foreground pixel distribution in color and spatial location. Another important element of their approach was the anchoring of visualisation and event browsing with the summary frame. Hanning and Kimber (2006) presented a multi-camera framework for detecting unusual events in surveillance videos. Their proposed framework uses a two-stage training to generate a probabilistic model for the usual events, and extract the unusual event by thresholding the likelihood of a test event generated by the usual event model (Xie, Sundaram, & Campbell, 2008).

In Toshev et al. (2006), Alexander et al. proposed an Apriori based model to discover frequent complex events from surveillance video. Simple events, such as a man standing or a car parked, were used as input to the model. Two main challenges, which were confronted in their model, were: how to determine the similarity between two events/classes of events and how to handle uncertainty in video data. To calculate the similarity between two patterns they introduce the concept of similarity measure. Here, $\text{sim}^{(m)}(p_1, p_2)$ means the similarity measure of order m between two m -patterns p_1 and p_2 . To confront the problem of uncertainty in video data, they introduce a Weak-Apriori property which decreases the support threshold for shorter patterns in order to prevent losing sub-patterns of frequent patterns. A manually created description of the data and results are presented in Fig. 4.

“Each flow displays a sequence of primitive events of ‘vehicle or person in a zone’ with the zone name and object type given. The occurrence refers to data descriptions. The discovered complex events are marked bold with their rank to the right” (Toshev et al., 2006).

Due to the importance of news broadcasts there has been much research effort on news video analysis. Zhang, Tan, Smoliar, and Yihong (1995) base their work on the anchorperson position in order to split the news into independent subjects. This model fits a type of news where the anchorperson and camera position do not change much. Mohan (1996) proposes to segment TV news by synchronising images with the associated close-captions or tele-text. Chen and Faudemay (1997) presents multi-criteria video segmentation based on image and sound analysis. In Chen et al. (2003), an effective data mining framework for automatic extraction of goal events in soccer video was presented. The proposed framework fully exploits the rich semantic information contained in visual and audio features for soccer video data and incorporates the data mining process for effective detection of soccer goal events. In Zhu et al. (2005), recognise the significance of semantic information for the video summarisation process. A framework was introduced to confront the challenges due to undefined relationships among video contents. In their approach different processing techniques were used to find visual and audio in sports videos (e.g., court field, camera motion activities, and applause) and then associations among those clues were explored. Discovery of associations were based on the domain expert knowledge or by manual observation, such as “after a match point score there are applause”.

In Chen et al. (2007) confronted the problem of video event detection by proposing a hierarchical temporal association mining model. Their proposed model consists of three main components: “feature extraction for video and audio streams”, “hierarchical temporal association mining” and “sequential pattern mining”. Firstly, video is divided into shots then visual and audio features from these shots are extracted. Extracted shot level features are

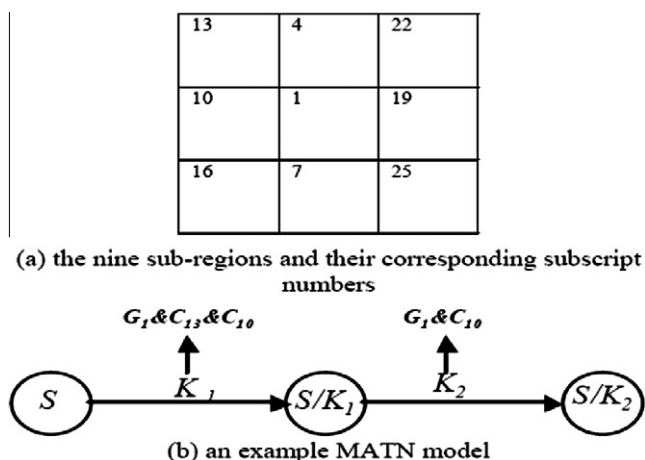


Fig. 3. MATN and multimedia input strings for modelling the key frames of traffic video shots (Chen et al., 2001).

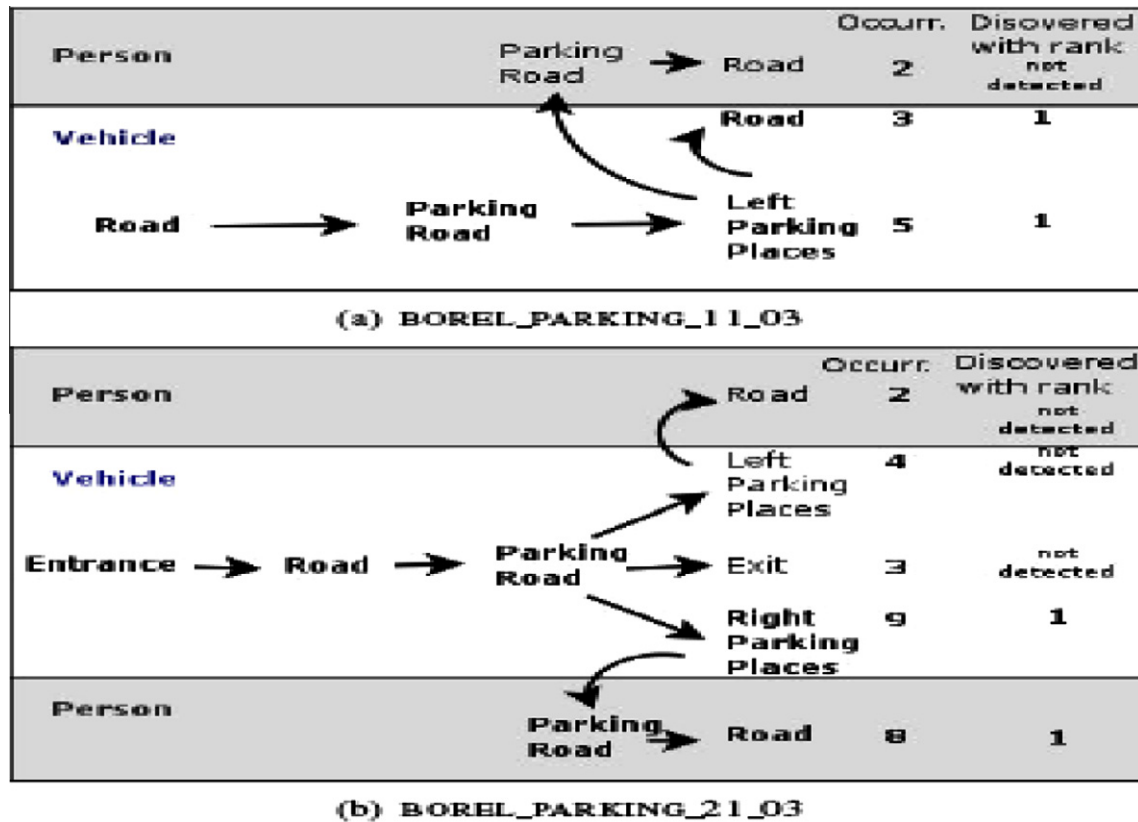


Fig. 4. Manually created description of the data and results (Toshev et al., 2006).

then provided as input to the extended association rule mining algorithm to discover the temporal patterns (important for characterising the events). In the last step, already discovered patterns are used for the sequential pattern mining process to discover the events of interest. Another important contribution in their work was the adaptive mechanism to determine the thresholds of minimum support and confidence level during the discovery of association rules and sequential patterns.

Lawrence and Yi-Jen (2008) proposed a video content management system for sports related videos. The system consists of three main modules: shot ontology definition, feature extraction and video indexing. For the classification of shots into two main categories of long shot and short shot they utilised the ratio of the skin coloured and court coloured portions of video shots. After extracting low level features they applied statistical analysis on 'motion', 'score board change' and 'shot type' to segment the videos into events such as goal, foul, free throw, etc. They validate their system on different basketball related videos.

In the Informedia project (Hauptmann and Witbrock, 1998; Hauptmann and Smith, 1995) speech recognition and image analysis were used to extract content information and to build indices and a summary. Xingquan and Xindong (2003) identify that associations among video contents can provide the basis for video summarisation. Their approach consists of three steps: video pre-processing, association mining and summary creation. The main concept was that in structured video contents (such as sport broadcasts, movies and news videos), certain sequential patterns exist. As shown in the dialog scene of Fig. 5, if we denote the actor by "A", the actress by "B" and the shot that contains both of them by "C", then all shots in the first row of Fig. 5 form a sequence "ABACAB" and "AB" is a sequential pattern; therefore by exploring this sequential pattern, effective summarisation can be done.

In Shirahama et al. (2007) a sequential pattern mining approach was used for extracting semantic events in structured videos (movie). Two types of temporal constraints (semantic event boundaries and temporal localities) were introduced for effective filtering of sequential patterns, which are unlikely to be semantic events. The main contribution of their approach was to transform the raw video data into multi-stream metadata, which not only characterises the semantic events, but can also be used as input to traditional sequential pattern mining algorithms designed for transactional data. In their proposed multi-stream approach, different aspects of multimedia contents were captured in order to be used for sequential pattern mining process. The representation of these meta-string is given below (as described in Shirahama et al. (2007)).

CH, CS, CV: reflect the background information of key frame in hue, saturation, and intensity axis.

LN: represents the number of object visible in each key frame.

LL: reflects the feature set of shape of objects (other than humans) in key frame.

LB: represents the dominant direction of straight lines in key frame.

SA: represents the size of the main character in the key frame.

LA: reflects if a weapon is present in the key frame of the shot.

SL: stores information about the duration of the shot.

MS: represents the movement of objects or background in a shot.

MV: is the direction of moment (object or background).

SM: stores the sound type information (such as speech, music or no sound).

AM: reflects the loudest sound in a shot (such as gunshots, screams, etc.).

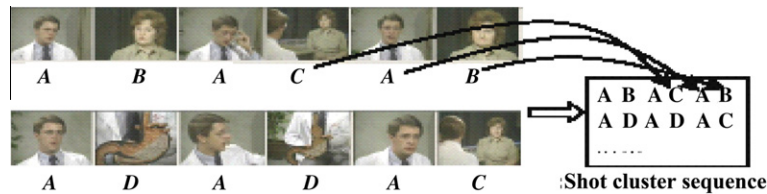


Fig. 5. Transforming a video into a relational dataset (Xingquan and Xindong, 2003).

An example of a multi-dimensional categorical stream S where only the occurrence of a 4-pattern $p_4 = (A_2, \text{nil}), (C_3, \text{parallel}), (C_4, \text{serial}), (E_1, \text{serial})$ from the time point $t = 1$ to $t = 4$ satisfies both SEB and T DT time constraints (see Fig. 6) (Shirahama et al., 2007).

In Tseng et al. (2006) applied association rules and sequential pattern mining approaches on both low-level feature and high-level semantic rules for effective video annotation. Later on in Tseng, Ja-Hwung, Jhih-Hong, and Chih-Jen (2008) they extend their work by incorporating the speech features into the model. The model consists of three stages: preprocessing, training and prediction. During the training stage four kinds of models for video annotations were constructed namely: Model_{CRM}, Model_{Vasso}, Model_{VSeq} and Model_{Sasso}. In the prediction stage these already constructed models are utilised for video annotation. The approach presented in Teredesai et al. (2006) combined low level image features (colour, orientation, intensity) with corresponding text annotations to generate association rules across multiple tables using multi-relational association rule mining. The multi-relational algorithm is basically an extension of the FP-Tree algorithm to discover association rules more effectively. Motivated by the presence of inter-concept association relationships and inter-shot temporal dependency Ken-Hao, Ming-Fang, Chi-Yao, Yung-Yu, and Ming-Syan (2008) presented a post-filtering framework for semantic concept detection in videos. The proposed framework applies association mining algorithms for the discovery of inter-concept association rules from annotation. Furthermore, a temporal filter was proposed to explore the inter-shot temporal dependency for improved detection accuracy (Ken-Hao et al., 2008).

2.1. Related research contributions from the data mining domain

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 (Agrawal & Srikant, 1995). The problem was to find all frequent sequential patterns from a given customer transaction database. They proposed three different algorithms: AprioriAll, AprioriSome and AprioriDynamic. Since then, a lot of work and improvements to the original algorithm have taken place (Heikki, Hannu, & Verkamo, 1997; Jiawei et al., 2000; Pei et al., 2001; Srikant and Agrawal, 1996; Zaki, 2001). 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. they predict what will be the next event/event-set in a sequence. For example, “Customers who purchase a HD-TV are likely to buy a SkyHD box within the next 30 days”, (HD-TV → SkyHD Box, “30 days”) is a sequential pattern with positive behaviour. However, an aspect of sequential patterns, which can be very helpful in multimedia mining system has not been fully explored. This aspect is to find the events, which have anomalous properties 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 (Cornells et al., 2006; Maria-Luiza and Osmar, 2004; Xindong, Chengqi, & Shichao, 2004). However they ignore the temporal aspects of events (temporal order of events in a pattern). The work presented in Kazienko (2008), concentrated on discovering the negative conclusion for a given sequential pattern in post mining environment. The concept is that if there is a frequently observed sequence “ f_s ” then elements of set X usually do not occur after that sequence.

2.2. Evaluation and discussion

Most of research efforts presented in Cucchiara et al. (2000), Dailey et al. (2000), Huang et al. (1994), Kamijo et al. (2000), Dieter et al. (1994), Shu-Ching et al. (2000, 2001a, 2001b), and Chen et al. (2001) focused on mining traffic related data and were mainly concerned with modelling and retrieving different known events, such as accidents, U-Turns, line crossings, etc. In Jung-Hwan et al. (2003) and Oh and Bandi (2002) research efforts were concentrated on indexing unstructured video contents. However, only low level information (motion intensity and location) was used for indexing and summarisation processes. The event mining approaches presented in Turaga et al. (2007), Hamid et al. (2005, 2007), Valery (2005) and Hanning and Kimber (2006) used unsupervised event discovery methods to discover usual/unusual events from single and multicamera environments. Different computational models are used, such as clustering algorithms, HMM and coupled HMM models, dynamic graphical models. The event mining approaches presented in Turaga et al. (2007), Hamid et al. (2005, 2007), Valery (2005) and Hanning and Kimber (2006) mainly concentrated on

	SEB1										SEB2					
time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
stream A:	A2	A7	A3	A2	A2	A2	A6	A6	A2	A4	A2	A5	A1	A3	A6
stream B:	B4	B4	B3	B3	B4	B4	B4	B3	B4	B4	B4	B4	B2	B4	B4
stream C:	C3	C4	C3	C0	C3	C0	C4	C0	C3	C2	C4	C0	C1	C1	C0
stream D:	D3	D3	D2	D1	D1	D2	D2	D2	D0	D1	D2	D1	D2	D3	D2
stream E:	E0	E3	E0	E1	E0	E2	E2	E2	E1	E1	E1	E3	E2	E1	E2

Fig. 6. An example of a multi-dimensional categorical stream (Shirahama et al., 2007).

discovering the usual/unusual events from raw multimedia surveillance footage; whereas, the work presented in this paper focuses on mining the already detected events to extract hidden information from them. Similar to our work, the approach presented in Toshev et al. (2006) falls into the category of a post-event detection model; however it does not explain how temporal aspects can be modelled during problem definition of the mining process. Furthermore, that approach discovers frequent complex events, whereas in surveillance videos the unusual events are more valuable.

The mining efforts presented in Zhang et al. (1995), Mohan (1996), Chen and Faudemay (1997), Chen et al. (2003), Zhu et al. (2005), Chen et al. (2007), Lawrence and Yi-Jen (2008), Hauptmann and Witbrock (1998), Hauptmann and Smith (1995) and Xingquan and Xindong (2003) are mainly based on already known relationships and are tightly domain specific as well; hence it is not feasible to use these frameworks for any other domain videos. Work presented in Shirahama et al. (2007), Tseng et al. (2006, 2008), Teredesai et al. (2006) and Ken-Hao et al. (2008) provides excellent frameworks for indexing and mining video information by utilising semantic information; however, these approaches are based purely on structured data. The transformation of raw multimedia data into metadata streams presented in Shirahama et al. (2007) is a solid concept as this data can be used with any association rules/sequential pattern mining algorithm developed for a transaction database. However the metadata presented in that approach is very much focused on structured data (more specifically on movies). Moreover, the experiment shows that discovery of semantic events was mainly due to the structured nature of movie contents. Although the work presented in Ken-Hao et al. (2008) applied mining algorithms on processed data, it is mainly focused on video retrieval and discovering frequent events.

Most of the work presented above is based upon finding the frequent association patterns to provide a hierarchical structure of the events in structured videos to be used for video retrieval and indexing. The research efforts dedicated to unstructured videos mainly deal with discovering the already known abnormal events. In contrast, the work presented in this paper is focused on the problem of discovering unknown anomalous events associated with a frequent sequence of events (A_{EASP}); that is to discover events which are unlikely to follow a frequent sequence of events. Furthermore, we also propose the event mining framework, which explores the relationship between entity feature-sets and associated text strings to generate appearance models of event entities automatically. The research approach suggested in our work differs from the “Pre-event detection” approach used in most previous research work as it is based upon a “Post-event detection environment”. In a “Pre-event detection environment”, the mining methods are not reliant on previous knowledge about the data, whereas in a “Post-event detection environment”, mining algorithms utilise already detected events results and perceptions of domain experts or already discovered patterns. An advantage of such an approach is that the proposed method can be integrated with any surveillance system or multimedia event mining method.

The work presented in Anwar et al. (2010), Kazienko (2008) and Anwar and Petrounias (2005) is similar to our concept of discovery of anomalous event against the given frequent sequence of events (presented in this paper); however, these approaches are specific to transactional databases as compared to the multimedia data addressed in our work. Due to the nature of multimedia data, the discovery process of A_{EASP} will be more intensive. Unlike in transaction data, where support is counted for each customer data sequence, with multimedia data all frequent sequences of events need to be searched and then events with anomalous properties are to be discovered and ranked. Multimedia data also introduce challenges of generating data sequences and handling frequent

sequences of events, which exist on the boundary of two data sequences. Furthermore, the algorithms presented in Kazienko (2008) and Anwar and Petrounias (2005) discover the items/item-sets having negative conclusion for only one given sequential pattern at a time, whereas, the work presented in this paper can discover anomalous events against multiple sequential patterns simultaneously. Confronting multiple sequential patterns simultaneously increases the complexity of the A_{EASP} 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 (SD) (SD is a time limit during which the sequential pattern must exist). Therefore, the process of searching the existence of each sequential pattern and then discovering the anomalous events can span into multiple search spaces, with each given sequential pattern having its own different search spaces to be confronted, these search spaces can also be known as search windows (SW). By SW we mean the limited search space in which we have to find the existence 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 min, respectively, and the data-sequence length is 10 min, we will have the following multiple SW for each given sequential pattern (Table 1).

With reference to the above mentioned consideration, it is important to devise a mechanism which can minimise the expensive process of searching the existence of multiple sequential patterns in each data sequence. To confront the above mentioned complexities, we introduce the dynamic sequential pattern search mechanism (D_{SPS_SM}). D_{SPS_SM} facilitates the discovery of all anomalous events against all the given sequential patterns in one database scan and optimises the A_{EASP} discovery process considerably. In addition to this our proposed mining framework also takes Anwar et al. (2010), the temporal aspect of A_{EASP} into consideration, that is to discover anomalous events which are true for a specific time interval only and might not be an A_{EASP} over a whole time line and vice versa.

3. Problem definition framework

One of the most important ingredients of any multimedia mining application is the provision of a flexible and comprehensive problem definition framework in which users can express the problem statement comprehensively and easily. Due to the temporal nature of A_{EASP} and sequential patterns it is important that before discussing the problem definition framework in depth, we introduce the two basic temporal entities that of chronon and interval. These temporal entities will be used for defining the temporal aspects of A_{EASP} and sequential pattern.

A chronon is an application dependent, non-decomposable time interval of some fixed minimal duration, at which an event takes place. The granularity of a chronon varies with each application, for example, it can be a millisecond, second, a minute. Whereas, an interval is a non-empty set of contiguous chronons; the granularity of intervals in multimedia mining applications varies due to the user preference. For example, interval granularity can be of minute, hour, shift (set of hours), or a day (Chen, 1999).

In a broader view, the problem of discovering all the A_{EASP} for all of the given sequential patterns can be defined as:

Table 1
Search windows for each sequential pattern.

SP	Search windows	No. of 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

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 A_{EASP} ($^{CL}A_{EASP}$). The goal is to discover all A_{EASP} with reference to each given sequential pattern. $^{CL}A_{EASP}$ can be all the events in the database or a user-defined list of events.

Definition 2.

$$\langle SP_{SET}, ^{CL}A_{EASP} \rangle$$

$$SP_{SET} = \{SP_1, SP_2, SP_3 \dots SP_n\}$$

$$^{CL}A_{EASP} = \{e_1, e_2 \dots e_n\}$$

As we are dealing with two concepts, sequential pattern and A_{EASP} , 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 events $SP = (e_1, e_2, e_3 \dots e_n)$ with temporal/sequential ordering satisfying the sequence duration (SD). SD is the time limit during which the sequential pattern must exist.

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

Since given sequential patterns can have different SD and their granularity can differ as well, the problem definition is further expanded as

$$SP_{SET} = \{(SP_1, SD_1, GR_1), (SP_2, SD_2, GR_2) \dots (SP_n, SD_n, GR_n)\}$$

$$\langle \text{"Carinentrancearea} \rightarrow \text{Carinparkingroad} \rightarrow \text{Carturnleft"}, 2, \text{minutes} \rangle$$

For a better understanding of problem definition of A_{EASP} we introduce four user-defined parameters, namely: time period (TP), Data Sequence Temporal Granularity (DS_{GR}), Maximum Time Interval (MT_{ITVL}) and Maximum Tolerance (M_{TOL}).

3.1. Time period (TP)

It is quite possible that the user is very much interested in discovering A_{EASP} against a given sequential pattern 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 A_{EASP} needs to be discovered. For example, we can say an event E is A_{EASP} against a sequential pattern X during the time period of 1st Jan. 2009 to 30th March 2009. This information not only reduces 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.

Definition 3.

$$\langle TP, SP_{SET}, ^{CL}A_{EASP} \rangle.$$

3.2. Data sequence temporal granularity (DS_{GR})

Unlike a transactional database where each data sequence is normally related to a specific customer or uniqueID, in a video events database there is no specific segmentation of video events into data sequences. Detected video events are normally stored sequentially with temporal information, such as event E at 7:00 pm on 10th July, 2009. Hence, the logical segmentation of detected events can be to divide them in some user-defined time granularity, which we call the Data Sequence Temporal Granularity (DS_{GR}). One important property of the user-defined DS_{GR} is that it should be higher than the granularity of given sequential patterns. For

example, if the given sequential pattern granularity is “Minutes” then DS_{GR} cannot be set to seconds or minutes it has to be hours or any other higher granularity.

Definition 4.

$$\langle TP, SP_{SET}, ^{CL}A_{EASP}, DS_{GR} \rangle.$$

3.3. 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 A_{EASP} . 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. For example, the information that an event “Vehicle on the parking road” does not normally follow the sequential pattern of events (“a vehicle on the road” \rightarrow “a vehicle on the parking road” \rightarrow “a vehicle in the parking place”) is only interesting within 02 minutes after the given sequential pattern. As each sequential pattern's nature can be different, MT_{ITVL} needs to be defined for each given sequential pattern.

Definition 5.

$$\langle TP, SP_{SET}, ^{CL}A_{EASP}, DS_{GR} \rangle$$

$$SP_{SET} = (SP_1, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle) \dots (SP_n, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle).$$

3.4. Maximum tolerance (M_{TOL})

Anomalous events 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 anomalous event against the given sequential pattern.

Definition 6.

$$\langle TP, SP_{SET}, ^{CL}A_{EASP}, DS_{GR}, M_{TOL} \rangle$$

$$SP_{SET} = (SP_1, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle, M_{TOL}) \dots (SP_n, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle).$$

Since the video event database will be segmented according to DS_{GR} , M_{TOL} needs to deal with the percentage of an event presence after the given sequential pattern during the user-defined MT_{ITVL} . Hence, if the M_{TOL} parameter is set to 10%, this means for an event to be considered as A_{EASP} , its support for a given sequential pattern must not exceed 10%. By support we mean presence of the event after the given SP within the user defined MT_{ITVL} . For example, if a given SP is found 10 times in a data sequence and event E support against the given SP is found 3 times within the user-defined MT_{ITVL} , then it cannot be considered as A_{EASP} for this data sequence since it exceeds the user defined parameter of M_{TOL} (10%). However, it is possible that an event with more support than M_{TOL} is still discovered as overall anomalous event against that specific sequential pattern (that is the average support of specific event in all data sequence is less than or equal to M_{TOL}).

3.5. Final problem definition

We have a database of multimedia events D spanning over a time domain T , each record is a tuple of $\langle \text{EventID}, \text{Event start time}, \text{Event end time} \rangle$; a known set of sequential patterns $SP_{SET} = \{SP_1, SP_2, SP_3 \dots SP_n\}$ where $SP(e_1, e_2, e_3 \dots e_i \dots e_n)$ is a set of events in sequential/temporal order ($e_1, e_2 \dots < e_j \dots < e_n$) along with

sequence duration (SD); a list of candidate events for A_{EASP} ($^{CL}A_{EASP}$) and the user-defined parameters of Time Period (TP), Data Sequence Granularity (DS_{GR}), Maximum Tolerance (M_{TOL}), and Maximum Time Interval (MT_{ITVL}). The problem to investigate here is to find all the events, which are unlikely to follow each given sequential pattern satisfying the user-defined parameters of M_{TOL} and MT_{ITVL} .

Definition 7.

$$SP_{SET} = \langle TP, SP_{SET}, ^{CL}A_{EASP}, DS_{GR}, M_{TOL} \rangle$$

$$SP_{SET} = (SP_1, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle) \dots (SP_n, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle)$$

$$\dots (SP_n, \langle SD, GR \rangle, \langle MT_{ITVL}, GR \rangle, MT_{ITVL}, ^{CL}A_{EASP}, DS_{GR}, M_{TOL})$$

The problem definition illustrates that every discovered A_{EASP} must have the following properties:

- A_{EASP} start time should always be greater than the given sequential pattern end-time (SP_{ET}). SP_{ET} is the time of the last event in the given sequential pattern.

$$A_{EASP_ST} > SP_{ET}$$

- A_{EASP} end time should always be less than or equal to the user-defined parameter of MT_{ITVL} .

$$A_{EASP_ST} \leq MT_{ITVL}$$

4. A_{EASP} discovery process

The discovery process of A_{EASP} consists of three main phases; Data Pre-Processing, Searching the instances of multiple sequential patterns and Discovery of the presence of $^{CL}A_{EASP}$ (Fig. 7). The last two phases work alternatively. 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 $^{CL}A_{EASP}$ and user-defined parameters of MT_{ITVL} and M_{TOL} to the next phase in which it attempts to discover the support of all $^{CL}A_{EASP}$ during the user-defined time limit (MT_{ITVL}). Any sequential pattern can occur multiple times in any data sequence, therefore we may have to find all the instances of the given sequential patterns in each data sequence and then find the support of all $^{CL}A_{EASP}$ during the user-defined time limit (MT_{ITVL}).

4.1. Data pre-processing

The data preprocessing phase consists of two steps: data filtering and data segmentation, that is to divide the event database into

data sequences. Firstly, we filter the data according to the user-defined parameter of Period Time (TP). For example, if TP is defined as 1st January, 2009 to 30th March, 2009 then all the events detected during this period will be filtered out for onwards data processing. Secondly, we divide the events database into user-defined Data Sequence Temporal Granularity (DS_{GR}). For example, if the granularity is defined as hours, then we will divide the filtered events database into different data sequences (DS), each will contain all the events detected during that specific hour (Fig. 8).

This segmentation of event data into data sequences is important as it enables us to discover A_{EASP} which are true for specific time intervals only. For example, it is possible that event X is an anomalous event for SP_n between 10 am and 11 am every day. However, during the rest of the day it is not an anomalous event against SP_n ; moreover dividing the data into DS also reduces the amount of memory/processing power required for the proposed algorithm.

A sequential pattern is a set of events with temporal and sequential ordering. Therefore, if we divide data into different time intervals according to the user-defined temporal granularity (DS_{GR}), it is possible that we may find a sequential pattern instance, which exists in two data sequences. To capture this property of sequential patterns, we generate a flying data sequence by taking a portion of the current and previous data sequences (Fig. 9). The length, starting position and end position of the flying data sequence can be calculated using the following equations:

$$F_{DS_SP}(P_{SD_P}) = (P_{SD_ET} - SD) + 1,$$

$$F_{DS_EP}(C_{SD_P}) = (((C_{SD_ST} + SD) - 1) + MT_{ITVL}) - C_{SD_ET},$$

$$F_{DS_L} = (P_{DS_ET} - F_{DS_SP}) + F_{DS_EP}.$$

P_{DS} = Previous data sequence

C_{DS} = Current data sequence

$F_{DS_SP}(P_{SD_P})$ = Flying data sequence start position from P_{DS}

$F_{DS_EP}(C_{SD_P})$ = Flying data sequence end position from C_{DS}

F_{DS_L} = Flying data sequence length

P_{SD_ET} = Previous data sequence end time

C_{SD_ST} = Current data sequence start time

C_{SD_ET} = Current data sequence end time

For example, if P_{SD_ET} and C_{SD_ST} is 60 and the given sequential pattern sequence duration is 10 along with an MT_{ITVL} of 5 then:

$$F_{DS_SP} = (60 - 10) + 1 = 51,$$

$$F_{DS_EP} = ((60 + 10) - 1) + 5 - 60 = 14,$$

$$F_{DS_L} = ((60 - 51) + 14) = 23.$$

4.2. Efficient 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 anomalous events against multiple

A_{EASP} Discovery Process

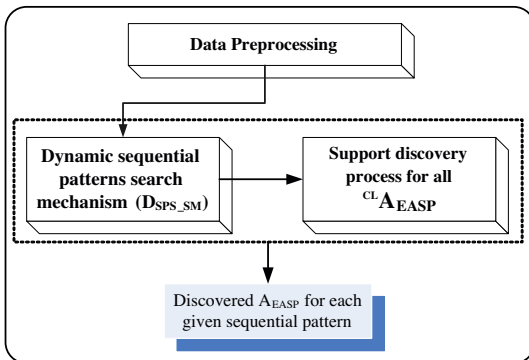


Fig. 7. A_{EASP} discovery process.

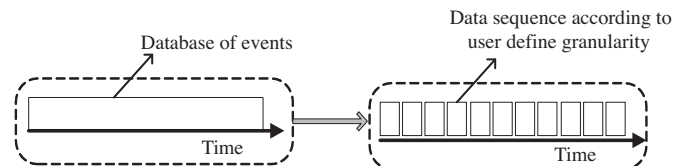


Fig. 8. Data transformation.

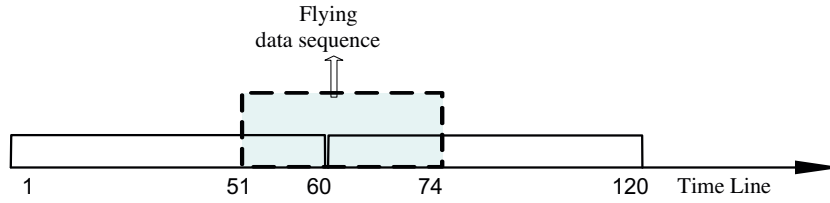


Fig. 9. Flying data sequence (suppose SD is 10 min).

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 min, respectively and the data-sequence length is 10 min, we will have the following multiple SWs for each given sequential pattern (Table 2).

The length of any SW is equal to the given SD and the number of SWs for a specific sequential pattern in a data sequence can be calculated from following equation:

$$(DS_{LEN} - SD_{LEN}) + 1.$$

$$DS_{LEN} = \text{Data sequence length}$$

$$SD_{LEN} = \text{Sequence duration (SD) length}$$

For example if SD for a given sequential pattern is 28 min and the length of the data sequence is equal to 1 h (60 min) then the total number of SWs for this specific SP is equal to 33 in each data sequence.

$$(DS_{LEN} - SD_{LEN}) + 1,$$

$$(60 - 28) + 1 = 33.$$

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 anomalous events against all the given sequential patterns in a single database scan and optimises the A_{EASP} 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.

4.3. Sequential pattern events status during D_{SPS_SM}

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

Table 2
Search windows for each sequential pattern.

SP	Search windows	No. of 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

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} .

If the C_{Event} is the first event of a given sequential pattern then there will be no E_{LHS} and if the C_{Event} is the last event of a given sequential pattern then there will be no E_{RHS} (Fig. 10).

Valid discovered event

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}).

$$E_T > E_{LHS_LDT}$$

E_T = Current event time

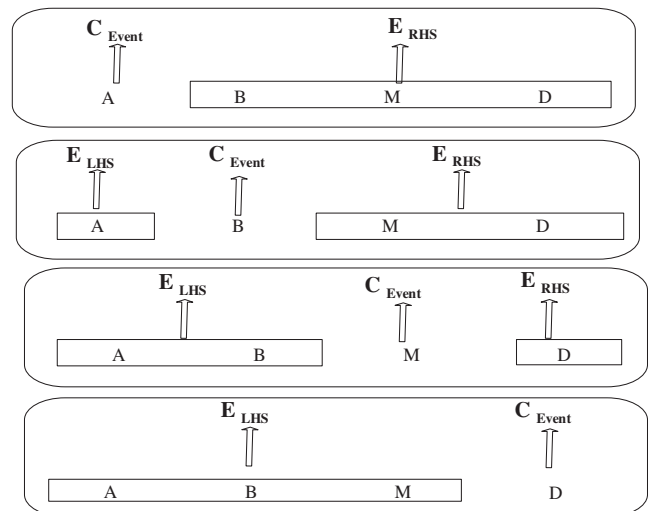
$C_{SW(ST)}$ = Start time of current search window

$C_{SW(ET)}$ = End time of current search window

E_{LHS_LDT} = Last discovered left hand side event time

4.4. Principles of dynamic sequential patterns search mechanism

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 a

Fig. 10. Sequential pattern events status in D_{SPS_SM} .

single database scan. Moreover, D_{SPS_SM} also utilises the temporal/sequential nature of sequential patterns 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.

4.5. 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 (Fig. 11(a)).

SD Property 2: The sequential pattern, which has the largest SD may overlap with multiple SWs of other SPs with shorter SD as shown in Fig. 11(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 expand with a single minute (Fig. 11(d)).

4.6. Sequential pattern temporal/sequential properties

During the sequential pattern search process a significant number of different SWs overlap with each other (Fig. 11(d)). 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:

4.7. 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}) (Fig. 12).

Suppose we have a sequential pattern $SP = (A \rightarrow C \rightarrow M \rightarrow Y \rightarrow C \rightarrow B)$ and support of the events A and C is discovered in C_{SW} . However, if 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} (see Fig. 12).

4.8. 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.

4.9. 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 pattern 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 patterns). The D_{SPS_SM} processes the data with the following data structure elements.

Sequential pattern information (SP_{Info}): This data structure is used to hold information about all the sequential patterns for which support needs to be searched. The SP_{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 (Table 3).

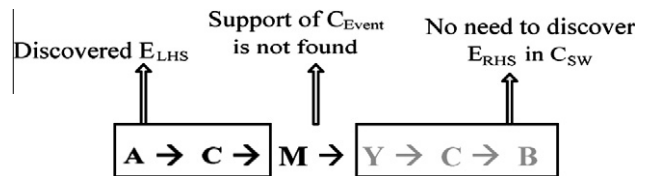


Fig. 12. First temporal/sequential property.

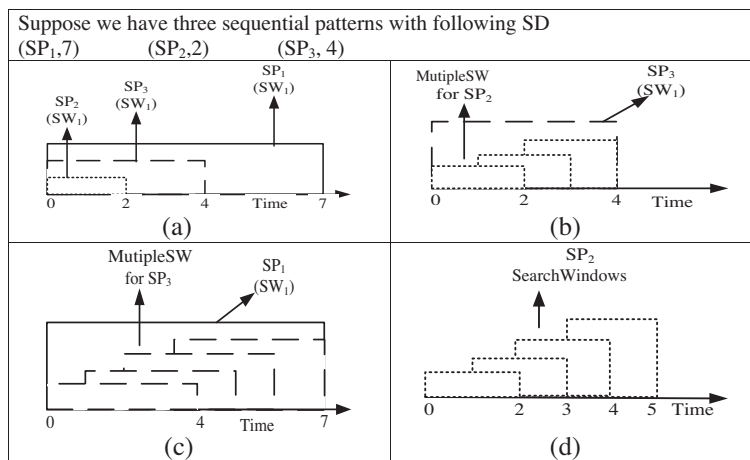


Fig. 11. Multiple search windows with sequence duration properties.

Search window information (SW_{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 patterns (Table 4). By F^{EP} and L^{EP} we mean the first and last detected event positions in a specific search window.

Sequential pattern events information (SPE_{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 (Table 5).

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} (Table 6).

SP discovered events information (SP_{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 (Table 7).

5. D_{SPS_SM} algorithm validation

For a better understanding of the D_{SPS_SM} concept and algorithm validation we run through the algorithm with the following example:

Suppose we have a sequence database “D” over a time domain “T” and three given sequential patterns $SP_{SET} = (SP_1, SP_2, SP_3)$ along with respective SDs (8,3,4). The problem to investigate is to find all instances of multiple given sequential patterns in a data-sequence

Table 3

SP_{info} .

SP_{ID}	SP	SD	SP^{SP}	SP^{EP}
Unique ID for each SP	SP event list	Sequence duration	SP start position	SP end position

Table 4

SW_{info} .

$SW^{\#}$	SP_{ID}	F^{EP}	L^{EP}
Search window number for specific SP	SP_{ID} from SP_{info} data structure	First event position	Last event position

Table 5

SPE_{info} .

Event ¹	Event ²	Event ⁱ	Event ⁿ
Position ¹	Position ¹	Position ¹	Position ¹
Position ⁱ	Position ⁱ	Position ⁱ	Position ⁱ
Position ⁿ	Position ⁿ	Position ⁿ	Position ⁿ

Table 6

C_{SE} .

SP_{ID}	Event
SP_{ID} from SP_{info} data structure	Event to be search

Table 7

SP_{DE} for each given SP.

(a) SP_1			(b) SP_2			(c) SP_n		
E^1	E^i	E^n	E^1	E^i	E^n	E^1	E^i	E^n
Pos	Pos	Pos	Pos	Pos	Pos	Pos	Pos	Pos

$$SP_1 = M \rightarrow A \rightarrow B \rightarrow M \rightarrow C, 8$$

$$SP_2 = \{B \rightarrow Q \rightarrow M, 3\},$$

$$SP_3 = \{P \rightarrow Q \rightarrow A \rightarrow B, 4\}$$

In the first step, D_{SPS_SM} populates SP_{info} and C_{SE} as per information given in the above problem definition. Since we are at the start of the discovery process C_{SE} will be populated with the first event of each SP (Table 8, Table 9).

Next, the algorithm populates the SW_{info} in accordance with the user-defined parameter of SD for each sequential pattern. For simplicity we assign ordered numeric values to events according to date/time and call this the position of the event during the example. Here, it is important to note that the number of events within a SW can vary, however the length of all the SWs of a specific SP will remain equal. Since SP_2 has the shortest first search window (SW_1) with the last event position (L^{EP}) being 07 (see Fig. 13); therefore, the L^{EP} for first SWs of all the other SPs will be set to 07 in SW_{info} (Table 10). As the SWs expand during the search process, information about the first search window will keep updating until it reaches the sequence duration (SD) length of that specific SP. At that point SW_{info} will be populated with new search window information for that specific SP.

D_{SPS_SM} then populates SPE_{info} by accessing the event position of all the unique events in given sequential patterns during C_{SW} . In

Table 8

SP_{info-1} .

SP_{ID}	SP	SD	SP^{SP}	SP^{EP}
1	$M \rightarrow A \rightarrow B \rightarrow M \rightarrow C$	8	?	?
2	$B \rightarrow Q \rightarrow M$	3	?	?
3	$P \rightarrow Q \rightarrow A \rightarrow B$	4	?	?

Table 9

C_{SE-1} .

SP_{ID}	Event
1	M
2	B
3	P

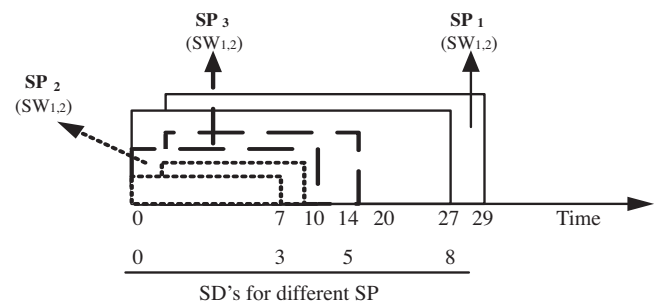


Fig. 13. SD of different given SP.

Table 10SW_{Info}-1.

SW [#]	SP ^{ID}	F ^{EP}	F ^{LP}
1	1	0	7
1	2	0	7
1	3	0	7

the first SW, SPE_{Info} is populated up to event position 07. For example, event M appears at 03 and event B in 04, 07 (Table 11 (first row)).

Next, D_{SPS_SM} populates the SP_{DE} of three given sequential patterns with valid events by taking into account the definition of valid discovered event conditions (as discussed in “Valid discovered event section”). D_{SPS_SM} discovers the first event “M” of SP₁ at position 03, which fulfills the valid discovered event conditions. Although event “A” is discovered (position 01) in C_{SW}, it does not satisfy the second condition of a valid discovered event (the event time should be greater than the last discovered E_{LHS}). Since the second event of SP₁ is not discovered, D_{SPS_SM} stops searching for any other events due to the first temporal/sequential property of SP (see “SP temporal/sequential property section”) and SP_{DE} for SP₁ is populated as follows (Table 12):

The first and second events “B”, “Q” of SP₂ are discovered at positions 04 and 05. Since both these detected events fulfil the required conditions of valid discovered events the algorithm updates the SP_{DE} for SP₂. Again although event ‘M’ (position 03) is discovered within C_{SW}, it does not satisfy the second condition of a valid discovered event. Hence, SP_{DE} for SP₂ is populated as follows (Table 13):

The first event of SP₃ is not found in C_{SW}, therefore SP_{DE} for SP₃ remains at its initial state (Table 14).

Table 11SPE_{Info}-1.

M	A	B	C	Q	P
3	1	4	2	5	
	2	7	3		
10	9	8	8	10	9
			9		
13	11	12	14	11	
				12	
				13	

Table 12SP_{DE}-SP₁-1.

SP ₁					
M	A	B	M	C	
3	?	?	?	?	

Table 13SP_{DE}-SP₂-1.

SP ₂					
B	Q			M	
4	5			?	

Table 14SP_{DE}-SP₃-1.

SP ₃			
P	Q	A	B
?	?	?	?

Table 15C_{SE}-2.

A(8)	P(9)	A(9)	M(10)	B(10)	Q(10)
SP ^{ID}					Event
1					A
2					M
3					P

5.1. First search window expansion

Before moving to the next search window the C_{SE} is updated with the next event of each SP to be searched (Table 15). Next, D_{SPS_SM} populates SW_{Info} and SPE_{Info} by only using the expanded portion of the new SW (Fig. 14), which has the following six events (Table 16 and Table 11 (second row)).

The already discovered first event of SP₁ is at position 03, which is within the C_{SW} boundary for SP₁ (0–10) positions, therefore it is a valid discovered event. As per the notion of (SP temporal/sequential property 2) we do not have to check the discovered E_{RHS} validity, because they are valid automatically. Now the search process of all the remaining events is carried out in only the expanded portion of SPE_{Info}. Event “A” is discovered at position (09), which fulfills the valid discovered event conditions. Although event “B” is discovered in C_{SW}, it does not satisfy the second condition of a valid discovered event. Hence D_{SPS_SM} terminates the search of any remaining events due to (SP temporal/sequential Property 1) and SP_{DE} for SP₁ is updated as follows (Table 17).

For SP₂ the already discovered event “B” is on position 04 and C_{SW} boundary for SP₂ is 03–10 positions, therefore, it is a valid event. According to the second temporal/sequential property of SP, if a first discovered event of a SP is valid, we do not have to check the E_{RHS} discovered event’s validity, because it will be valid automatically. Now, support of the undiscovered third event of SP₂ “M” must be searched in the expanded portion of the SPE_{Info}. D_{SPS_SM} finds support for “M” at position 10 (Table 11 second row). Since it satisfies all the conditions of a valid discovered event and it is the last event for the SP₂, this means an instance of SP₂ has been found successfully (Table 18). Next, D_{SPS_SM} passes the end position of the discovered SP₂ support, along with other user-defined parameters to “Support discovery process for all C_{EA}SP” and D_{SPS_SM} restarts the search process for the next instance of SP₂.

D_{SPS_SM} discovers events “P and Q” of SP₃ at positions 09 and 10, which fulfil the valid discovered event conditions. Although event “A” is discovered in C_{SW}, it does not satisfy the second condition of a valid discovered event; hence, SP_{DE} for SP₃ is populated as follows (Table 19):

5.2. Second search window expansion

After updating the C_{SE}, the D_{SPS_SM} populates SW_{Info} and SPE_{Info} by only using the expanded portion of the new SW (Table 20, Table 21 and Table 11 (third row)), which has the following seven events:

The already first discovered event of SP₁ is at location 03, which is within the C_{SW} boundary for SP₁ (0–14), therefore it is a valid discovered event. Moreover, we do not need to check the discovered

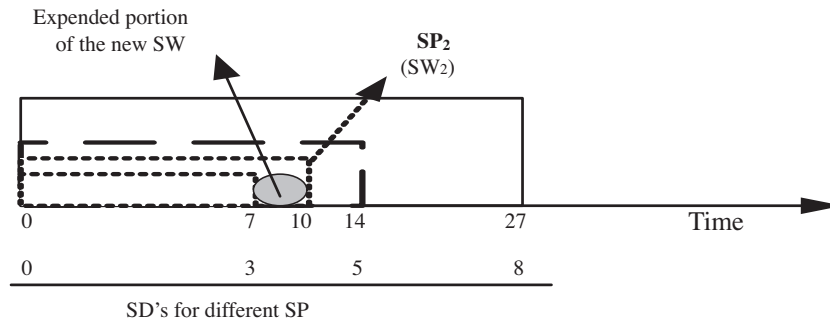


Fig. 14. Expanded portions of search windows.

Table 16

SW_{Info_2}.

SW ^{ID}	SP ^{ID}	F ^{SP}	F ^{EP}
1	1	0	10
2	2	3	10
1	3	0	10

Table 17

SP_{DE-SP1_2}.

SP ₁				
M	A	B	M	C
3	9	?	?	?

Table 18

SP_{DE-SP2_2}.

SP ₂		
B	Q	M
4	5	10

Table 19

SP_{DE-SP3_2}.

SP ₃			
P	Q	A	B
9	10	?	?

Table 20

SW_{Info_3}.

A(11)	Q(11)	Q(12)	B(12)	M(13)	Q(13)	C(14)
SP ^{ID}						Event
1						B
2						B
3						A

ERHS validity, because they are valid automatically (SP temporal/sequential property 2). Now the search process of all the remaining events is carried out in only expanded portion of SPE_{Info}. Events “B, M, C” are discovered at positions (12, 13, 14) and all of them fulfil the valid discovered event conditions. Since all the events of SP₁

Table 21

SW_{Info_3}.

SW [#]	SP ^{ID}	SP	EP
1	1	0	14
3	2	7	14
2	3	3	14

Table 22

SP_{DE-SP1_3}.

SP ₁				
M	A	B	M	C
3	8	12	13	14

Table 23

SP_{DE-SP2_3}.

SP ₂		
B	Q	M
7	11	13

are discovered as shown in Table 22, D_{SPS_SM} passes the end position of discovered SP₁ instance along with other user-defined parameters to “Support discovery process for all $^{CL}A_{EASP}$ ” and D_{SPS_SM} restarts the search process for the next instance of SP₁.

The events ‘B, Q, M’ of SP₂ are discovered at positions 07, 11 and 13. Since all three detections of events fulfil the required conditions, this means the second instance of SP₂ is discovered in the third SW of SP₂ (Table 23). D_{SPS_SM} passes the end position of the discovered SP₂ instance along with other user-defined parameters to “Support discovery process for all $^{CL}A_{EASP}$ ” and D_{SPS_SM} restarts the search process for the next instance of SP₂.

D_{SPS_SM} discovers event “A, B” of SP₃ at positions 11, 12, which fulfil the valid discovered event conditions. Since event “B” is the last event of SP₃, it means the first instance of SP₃ is discovered in the second SW of SP₃ (Table 24). Next, D_{SPS_SM} passes this

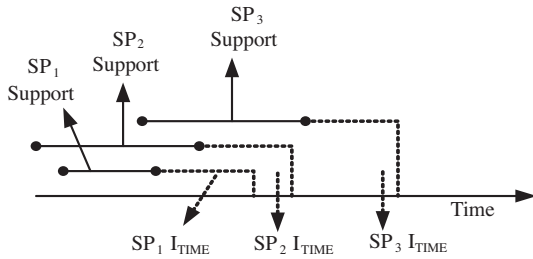
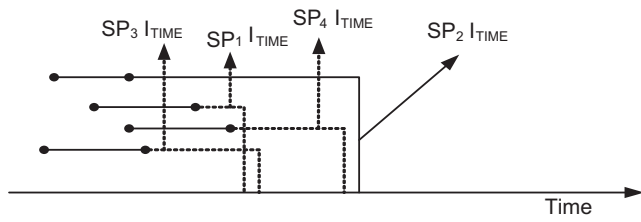
Table 24

SP_{DE-SP3_3}.

SP ₃			
P	Q	A	B
9	10	11	12

Table 25Updated SP_{Info-1} .

SP^{ID}	SP	SD	SP^{SP}	SP^{EP}
1	$M \rightarrow A \rightarrow B \rightarrow M \rightarrow C$	8	3	14
2	$B \rightarrow Q \rightarrow M$	3	4, 7	10, 13
3	$P \rightarrow Q \rightarrow A \rightarrow B$	4	9	12

**Fig. 15.** Different I_{TIME} for each sequential pattern.**Fig. 16.** Overlapping I_{TIME} 's for different SP's.

information to “Support discovery process for all $^{CL}A_{EASP}$ ” and re-starts the search process for the next instance of SP_3 .

At this stage of D_{SPS_SM} the values of SP^{EP} can be seen in Table 25.

5.3. Support discovery process for all $^{CL}A_{EASP}$ (A_{EASP_DS})

Once D_{SPS_SM} finds the instance of a specific given SP , it passes the end time/position of the discovered SP support to A_{EASP_DS} along with user-defined parameters of M_{TOL} and MT_{ITVL} . The main purpose of A_{EASP_DS} is to count the support of all the candidate events given in $^{CL}A_{EASP}$ during the limited time space (based on the 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 $MT_{ITVL-ET}$ as shown in Fig. 15. Since each given sequential pattern has its own MT_{ITVL} and their SP_{ET} might be different, therefore A_{EASP_DS} may have to deal with different I_{TIME} simultaneously.

Just like search windows of different sequential patterns, the I_{TIME} s of different SPs can also overlap each other. Due to this property of I_{TIME} , we can utilise the support discovery of $^{CL}A_{EASP}$ between I_{TIME} s of different SPs. In Fig. 16, the 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 .

Furthermore, the I_{TIME} s of the same SP can also overlap with each other, therefore support discovery of $^{CL}A_{EASP}$ between I_{TIME} s of the same SPs can be reused as well (see Fig. 17). For example, if the first I_{TIME} is from 10 to 16 and the second I_{TIME} is from 12 to 17 the support of $^{CL}A_{EASP}$ between 12 and 16 can be reused.

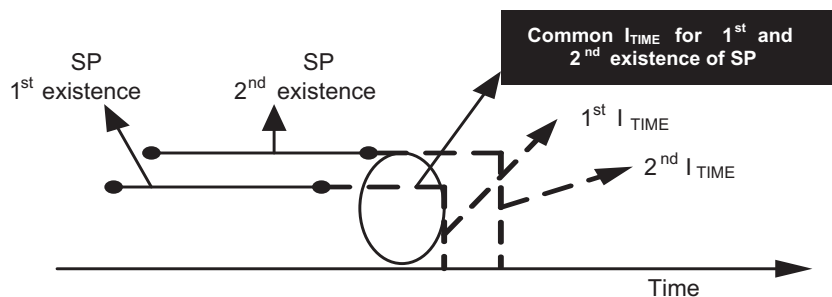
Based on the A_{EASP} discovered in all data sequences the proposed algorithm then calculates the average support (as a percentage) of $^{CL}A_{EASP}$ events against each sequential pattern and outputs the discovered anomalous events against each sequential pattern. That is, to identify the events with average support less than the user defined parameter of M_{TOL} . The algorithm also outputs A_{EASPs} which are true for a specific time interval only and might not be an A_{EASP} over a whole time spectrum and vice versa. For example, event X is anomalous event for SP_n between 10 am and 11 am every day, however during rest of the day it is not an anomalous event against SP_n . To discover the periodicity of A_{EASP} on the granularity level in which data sequences were divided, the proposed algorithm calculates the average support of $^{CL}A_{EASP}$ events against each sequential pattern during each set of respective intervals, for example, 1st hour of all the 30 days, 2nd hour of all the 30 days, etc.

6. Algorithm

Fig. 18.

6.1. A_{EASP} algorithm evaluation

The proposed algorithm to discover A_{EASP} (see Fig. 18) for all the given sequential patterns was implemented in the Matlab environment and experiments have been conducted to evaluate the algorithm's accuracy and performance. The main inputs to the algorithm are already detected events and known frequent sequences of events (sequential patterns). These sequential patterns can be a perception of a domain expert or they can be already dis-

**Fig. 17.** Overlapping I_{TIME} 's for same SP.

Inputs	
1	D_{EVENTS} //Already detected events dataset
2	DS_{GR} //Data sequence granularity/
3	TP //Time Period
4	SP_{SET} //Set of know sequential pattern
5	SD //Sequence duration of each SP
6	MT_{ITVL} //Maximum time interval for each SP
7	M_{TOL} //Maximum Tolerance
8	$CL_{A_{EASP}}$ //Candidate list for A_{EASP}
Output	
9	Anomalous event against each given sequential pattern (A_{EASP})
Data Structure	
10	$SP_{Info}, SW_{Info}, SPE_{Info}, C_{SE}, SP_{DE}(s)$
11	A_{EASP}^{Info} // store information about discovered A_{EASP} in each DS
Data filtering	
12	$F_{DATA} = \text{dataFilter}(D_{EVENTS}, TP, DS_{GR});$
13	Loop // Loop till last data sequence
14	if ($F_{DATA} =$ // all data sequence has been processed.
15	break;
16	end;
17	populateSWInfo (SW_{Info}, F_{DATA}^n) // Populate search windows
18	Loop // Loop for all the given SP(s)
	// fetch values according to current search window (C_{SW})
19	$[SP_{Info}, SPE_{Info}, C_{SE}] = \text{fetchValues}(SW_{Info}, F_{DATA}^n);$
20	Loop //check if already discovered event is valid in C_{SW}
21	result= validEvent ($SP_{DE}(n)$);
22	if (result=Yes)
23	// next event of SP to search
24	nextEvent2Search (SP_{Info}, C_{SE});
25	else
	//populate C_{SE} with 1 st event of SP
26	firstEvent (SP_{Info}, C_{SE});
27	end;
28	endLoop;
29	result= searchValidEvent (SP_{Info}, C_{SE});
30	if (result=Yes)
31	update($C_{SE}, SP_{DE}(n)$);
32	if ($SP_{DE}(n) = C_{SE}$) //if all the events of SP found
33	update(SP_{Info});
34	initialise($C_{SE}, SP_{DE}(n)$);
	//function to discovery A_{EASP} for each SP
35	A_{EASP_Fun} ($D_{EVENTS}, A_{EASP}^{Info}, SP_{Info}, MT_{ITVL}$);
36	break; // move to next SP
37	end;
38	break; // move to next SP
39	endLoop; // move to next data sequence
40	// conditional process if periodic A_{EASP} is not required to be discover
41	if ($CL_{A_{EASP}(n)} > M_{TOL}$)
42	//function to remove the events having support $> M_{TOL}$ parameter//
43	$CL_{A_{EASP}} = \text{removeInvalidEvents}(A_{EASP}^{Info});$
44	end;
45	endLoop; // all data sequence has been processed
46	discover $A_{EASP}(A_{EASP}^{Info})$ // discover overall A_{EASP} against each SP
47	discover $A_{EASP}(A_{EASP}^{Info}, T_{ITVL})$ // discover A_{EASP} for specific Time Interval

Fig. 18. Algorithm for the discovery of A_{EASP} for all the given SPs.

Table 26

List of source sequential patterns.

SP #	Sequential pattern
1	$B \rightarrow M \rightarrow A \rightarrow G$
2	$F \rightarrow B \rightarrow M \rightarrow A \rightarrow G$
3	$A \rightarrow X \rightarrow S \rightarrow X$
4	$B \rightarrow H \rightarrow Q \rightarrow Y \rightarrow Z \rightarrow P \rightarrow U$
5	$P \rightarrow Q \rightarrow Y \rightarrow Z \rightarrow P \rightarrow U$
6	$P \rightarrow Q \rightarrow Y \rightarrow Z \rightarrow P$
7	$Q \rightarrow N \rightarrow P \rightarrow U$
8	$E \rightarrow G \rightarrow M \rightarrow Q$
8	$P \rightarrow U \rightarrow Z \rightarrow P$
10	$P \rightarrow V \rightarrow P$

Table 28

Evaluation of algorithm results on data sequence (2–5).

SP	# of SP instances detected by the algorithm	# of SP instances checked manually	# of SP instances not detected by algorithm	A_{EASP} discovered by algorithm and manually confirmed	$CL_{A_{EASP}}$ events whose existence $> M_{TOL}$ and manually confirmed
Data sequence 02					
1	05(0)	05(0)	0	05	03
2	01(0)	01(0)	0	04	04
3	01(0)	01(0)	0	07	01
4	02(0)	02(0)	0	08	0
5	01(0)	01(0)	0	08	0
6	01(1)	01(1)	0	07	01
7	06(0)	06(0)	0	07	01
8	05(0)	05(0)	0	02	06
9	05(0)	05(0)	0	07	01
10	07(0)	07(0)	0	03	05
Data sequence 03					
1	03(0)	03(0)	0	04	04
2	01(0)	01(0)	0	07	01
3	03(0)	03(0)	0	07	01
4	01(0)	01(0)	0	07	01
5	03(0)	03(0)	0	05	03
6	04(0)	04(0)	0	01	07
7	04(1)	04(1)	0	07	01
8	02(0)	02(0)	0	06	02
9	06(0)	06(0)	0	04	04
10	03(0)	03(0)	0	04	04
Data sequence 04					
1	03(0)	03(0)	0	04	04
2	01(0)	01(0)	0	05	03
3	03(0)	03(0)	0	07	01
4	00(0)	00(0)	0	0	0
5	00(0)	00(0)	0	0	0
6	01(0)	01(0)	0	07	01
7	02(0)	02(0)	0	08	0
8	01(0)	01(0)	0	05	03
9	04(0)	04(0)	0	05	03
10	05(0)	05(0)	0	04	04
Data sequence 05					
1	03(0)	03(0)	0	03	05
2	01(0)	01(0)	0	06	02
3	04(0)	04(0)	0	06	02
4	02(0)	02(0)	0	05	03
5	03(0)	03(0)	0	05	03
6	03(0)	03(0)	0	04	04
7	04(0)	04(0)	0	05	03
8	05(0)	05(0)	0	05	03
9	06(0)	06(0)	0	06	03
10	01(0)	01(0)	0	06	02

Table 27

Evaluation of algorithm results on first data sequence.

SP	# of SP instances detected by the algorithm	# of SP instances checked manually	# of SP instances not detected by algorithm	A_{EASP} discovered by algorithm and manually confirmed	$CL_{A_{EASP}}$ events whose existence $> M_{TOL}$ and manually confirmed
1	09	09	0	03	05
2	05	05	0	02	06
3	07	07	0	06	02
4	02	02	0	06	02
5	02	02	0	05	03
6	03	03	0	05	03
7	07	07	0	04	04
8	05	05	0	04	04
9	02	02	0	07	01
10	03	03	0	03	05

covered patterns by using any of the sequential pattern discovery methods presented in the literature or a combination of both.

6.2. Experimental data

We have used synthetic data to evaluate the accuracy and efficiency of the algorithm. The choice of using synthetic data is motivated by the fact that we can control the nature of the input data, which means we can ensure that input data actually contains the sequential patterns and A_{EASP} s that need to be discovered by the algorithm. Moreover, since the proposed algorithm mainly focuses on the discovery of anomalous events against sequential patterns and assumes already detected events and known sequential patterns as its input, the use of synthetic data instead of real world data does not impact on the evaluation of algorithm's accuracy or efficiency.

6.3. Experiments and results

We first generated the synthetic data to replicate the already detected events along with date/time information and designed 10 different sequential patterns along with their parameters to

Table 29 A_{EASP} for each sequential pattern.

$CL_{A_{EASP}}$	1	2	3	4	5	6	7	8	9	10
Sequential patterns										
Q	8.70	22.22	0.00	28.57	22.22	16.67	13.04	100.00	0.00	15.79
L	21.74	33.33	11.11	28.57	11.11	33.33	21.74	5.56	8.70	26.32
M	34.78	44.44	0.00	0.00	22.22	33.33	0.00	16.67	21.74	15.79
N	26.09	33.33	0.00	14.29	11.11	33.33	0.00	5.56	4.35	26.32
P	13.04	0.00	0.00	28.57	11.11	100.00	17.39	16.67	100.00	100.00
X	30.43	33.33	100.00	28.57	33.33	33.33	39.13	27.78	0.00	15.79
Z	21.74	33.33	0.00	28.57	44.44	16.67	4.35	22.22	21.74	26.32
A	39.13	33.33	0.00	0.00	0.00	16.67	0.00	44.44	0.00	5.26

Table 30Overall A_{EASP} discovered from 40 data sequences.

$CL_{A_{EASP}}$	1	2	3	4	5
<i>Sequential pattern</i>					
Q	38.46	36.12	21.41	8.87	3.16
L	28.23	29.12	15.86	15.67	17.75
M	25.72	25.24	18.87	21.04	25.46
N	17.76	19.62	20.98	24.74	17.05
P	26.06	26.70	20.31	21.03	12.50
X	25.20	24.31	100.00	30.85	19.19
Z	18.83	16.38	11.77	27.16	26.41
A	16.85	18.10	17.51	16.39	14.05

be used in different experiments. Next, we manually altered the data sequences to inject instances of different, already designed sequential patterns.

In our first experiment; the input to the algorithm are a synthetically generated events dataset; 10 designed sequential patterns (Table 26), user-defined parameter of $CL_{A_{EASP}}$, which is assigned 8 different events (Q, L, M, N, P, X, Z, A), and a user-defined parameter of maximum tolerance (M_{TOL}) with 6% as its value. The synthetic data was then divided into five different data sequences, each containing 300 random events along with the event time in seconds and intervals (minutes). We then applied the proposed algorithm on the first data sequence and conducted the following investigation on the results:

Manually check that instances of different sequential patterns detected by the algorithm actually exist in the data sequence.

Inspect through the data sequence to see if there are any instances of sequential patterns that are not detected by the algorithm.

Manually check that the discovered anomalous events against specific sequential pattern truly exist in the data sequence.

Table 27 presents the results of the algorithm and onward investigations. The first column of Table 27 contains the sequential pattern IDs. The second column gives the number of sequential pattern instances detected by the algorithm. In the third column the number of each sequential pattern instance successfully checked manually is given. The fourth column represents the number of sequential pattern instances, which are not found by the algorithm. Now, if the sum of columns 3 and 4 is equal to the value of column 2 the algorithm has detected all the instances of the given sequential patterns with complete accuracy. In column five, the number of anomalous events discovered by the algorithm and subsequently manually confirmed are given, whereas in the last column of Table 27 the number of events (from $CL_{A_{EASP}}$) having greater presence than the user-defined parameter of M_{TOL} and subsequently manually confirmed are presented. The experiment's results show that the proposed algorithm successfully detected all the instances of different sequential patterns and discovered all the true A_{EASP} from the $CL_{A_{EASP}}$.

We repeated the above process on four more data sequences. Table 28 presents the results of each data sequence. In addition to the information given in Table 27, Table 28 also contains results of flying data sequences (results of flying data sequences are presented in column 2 and 3 with brackets around it). The use of a flying data sequence enables the algorithm to detect sequential pattern instances, which exist on the boundary of two data sequences. The flying data sequence is generated by taking a portion of the current and previous data sequences (for detail see "Data pre-processing section").

Based on the results of all five data sequences the proposed algorithm then calculates the presence of all $CL_{A_{EASP}}$ events against each sequential pattern and outputs the discovered anomalous events against each sequential pattern. In Table 29, we have aver-

Table 31 A_{EASP} discovered with hourly periodicity.

Hour/interval	$CL_{A_{EASP}}$	Sequential patterns				
		1	2	3	4	5
1	Q	18.57	33.33	20.14	0.00	0.00
	L	28.57	20.00	29.58	0.00	0.00
	M	51.83	29.33	70.14	0.00	12.50
	N	13.57	33.33	12.92	0.00	12.50
	P	17.86	26.67	25.42	100.00	18.75
	X	23.89	32.00	100.00	0.00	0.00
	Z	29.60	24.00	0.00	0.00	25.00
	A	28.89	32.00	66.39	0.00	25.00
2	Q	42.19	40.74	23.68	0.00	0.00
	L	5.73	5.56	35.96	0.00	5.00
	M	20.83	24.07	5.26	0.00	0.00
	N	6.77	12.96	65.79	100.00	68.82
	P	68.02	68.52	23.68	0.00	23.66
	X	4.69	7.41	100.00	0.00	8.60
	Z	10.42	11.11	2.63	0.00	22.58
	A	17.19	18.52	14.91	100.00	0.00
3	Q	77.89	63.64	29.82	N/A	N/A
	L	11.58	12.12	5.26	N/A	N/A
	M	2.11	6.06	29.82	N/A	N/A
	N	21.05	39.39	5.26	N/A	N/A
	P	9.47	3.03	25.81	N/A	N/A
	X	65.26	60.61	100.00	N/A	N/A
	Z	11.58	15.15	5.26	N/A	N/A
	A	11.58	15.15	29.82	N/A	N/A
4	Q	24.56	16.07	30.42	12.90	8.60
	L	14.12	10.71	16.13	0.00	0.00
	M	45.86	48.21	23.87	19.35	19.35
	N	7.31	0.00	0.00	12.90	8.60
	P	38.37	42.86	19.46	12.90	12.90
	X	20.00	10.71	100.00	19.35	19.35
	Z	3.95	5.36	21.07	67.74	67.74
	A	9.18	17.86	5.83	12.90	8.60
5	Q	11.58	33.64	15.14	0.00	0.00
	L	12.63	10.91	27.08	41.38	41.38
	M	2.63	0.00	21.11	34.48	34.48
	N	41.58	33.64	18.89	0.00	0.00
	P	31.58	27.27	58.06	0.00	0.00
	X	63.16	72.73	100.00	75.86	75.86
	Z	56.84	61.82	8.89	34.48	34.48
	A	21.58	21.82	22.50	17.24	17.24
6	Q	46.67	50.88	0.76	3.23	12.82
	L	43.33	57.02	0.76	22.58	6.41
	M	17.78	14.04	0.00	0.00	23.08
	N	33.33	44.74	38.38	6.45	23.08
	P	15.56	14.91	1.52	19.35	3.85
	X	0.00	0.00	100.00	45.16	17.95
	Z	33.33	18.42	22.47	6.45	34.62
	A	22.22	26.32	12.88	0.00	0.00
7	Q	27.49	9.03	27.27	35.29	0.00
	L	33.12	54.86	3.41	5.88	0.00
	M	5.84	14.58	4.55	47.06	38.10
	N	2.81	4.86	0.00	58.82	57.14
	P	2.81	4.86	3.41	5.88	33.33
	X	6.06	2.78	100.00	41.18	9.52
	Z	0.00	0.00	4.55	47.06	38.10
	A	7.36	7.64	1.14	52.94	38.10
8	Q	48.18	52.08	26.94	0.00	0.00
	L	58.18	56.25	34.29	0.00	0.00
	M	75.00	58.33	45.82	46.15	28.68
	N	2.73	0.00	7.14	82.05	16.18
	P	39.55	31.25	30.61	0.00	0.00
	X	13.64	12.50	100.00	0.00	11.03
	Z	0.00	0.00	46.02	0.00	0.00
	A	14.55	8.33	8.16	0.00	3.68

age support (in percentage) of $CL_{A_{EASP}}$ events against each sequential pattern. For example, event "Q" has 8.70% average support against SP₁, 22.22% against SP₂ and 0.00% against SP₃ etc. In Table 29, cells with shading indicate that the event is an anomalous

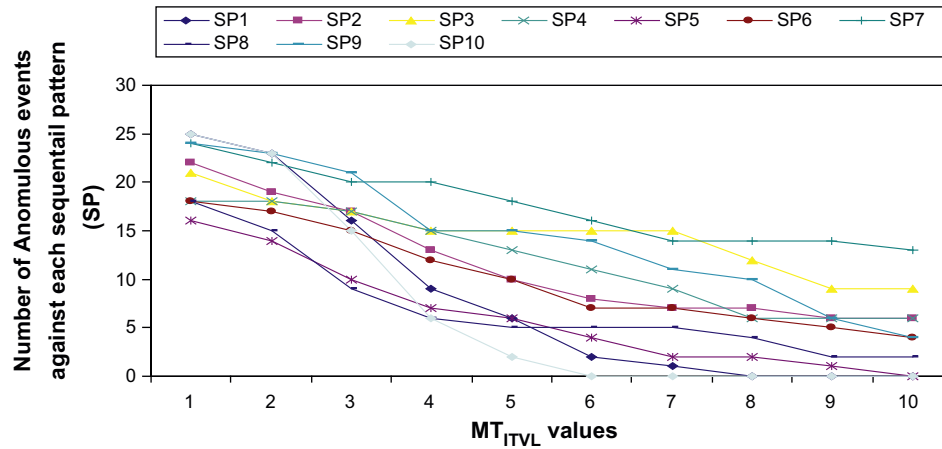


Fig. 19. Number of AEASP against each increasing MT_{ITVL} .

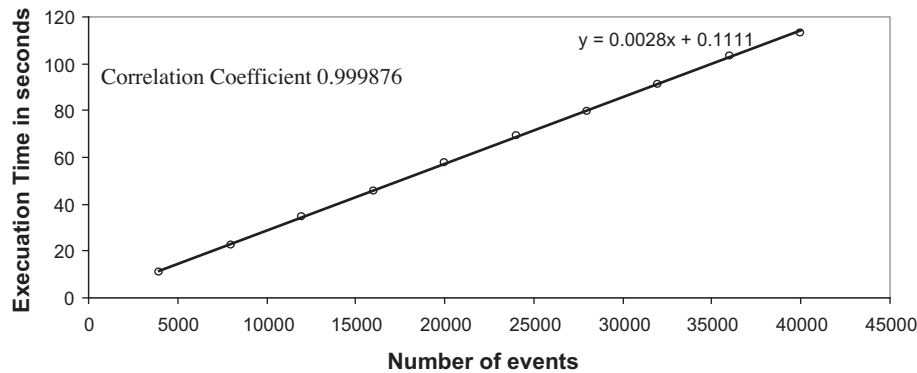


Fig. 20. Linear growth in execution time against number of input events.

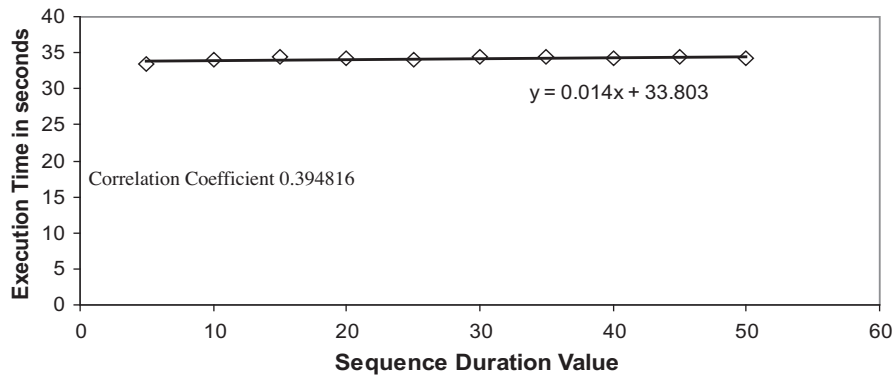


Fig. 21. Execution time with different value for SD parameter.

event against the specific sequential pattern, as their support against the specific sequential pattern is less than or equal to the user-defined parameter of T_{MOL} (which is 6 in this experiment). For example, SP_1 has no anomalous events and SP_2 has one anomalous event (P), etc.

In our second experiment, we applied the proposed algorithm to discover not only overall anomalous events against each SP, but also to discover A_{EASP} which are true for a specific time interval only and might not be an A_{EASP} over a whole time interval and vice versa. We then analyse the A_{EASP} discovered overall and at data se-

quence interval level. The inputs to the algorithm are a synthetically generated events dataset; 5 designed sequential patterns, user-defined $CL_{A_{EASP}}$ which is assigned with eight different events (Q, L, M, N, P, X, Z, A) and a user defined parameter of maximum tolerance (M_{TOL}) with 5% as its value. The synthetic data was then divided into 40 different data sequences representing 8 hours for each day (5 days); where each data sequence contains 300 random events.

In Table 30, we have average support (in percentage) of $CL_{A_{EASP}}$ events against each sequential pattern, whereas in Table

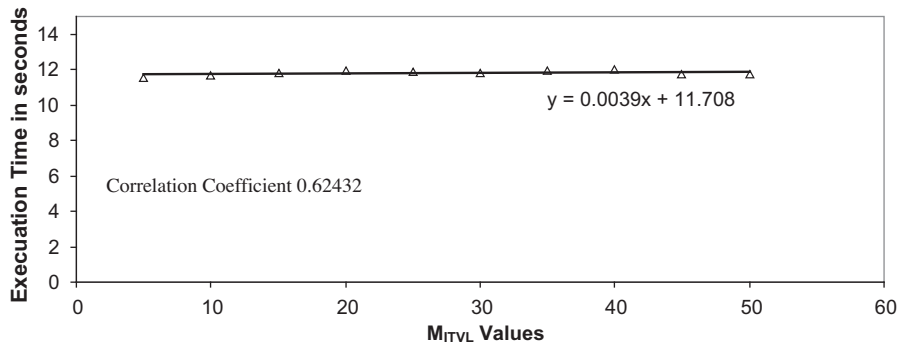


Fig. 22. Execution time with different value for MT_{ITVL} parameter.

31 we have average support of $cl_{A_{EASP}}$ events against each sequential pattern calculated at data sequence interval level (hourly basis). We can see from Table 30 that there is only one overall A_{EASP} discovered, “Q”, which is against SP_5 (the cell with shading). On the other hand, in Table 31 we can see that the algorithm has discovered a number of A_{EASP} which are true to specific data sequence intervals only, but they were not discovered over a whole time spectrum (Table 30). For example, in the first hour, event “Z” is discovered as an anomalous event against SP_3 and events (Q, L, M, N, X, Z) are discovered A_{EASP} against SP_4 . However, none of them were discovered as A_{EASP} over the whole time spectrum. Moreover, it can also be seen that although event “Q” is an anomalous event against SP_5 over the whole time spectrum, it is not an anomalous event against SP_5 during the 4th and 6th hour.

The results of the different experiments illustrate that the number of A_{EASPs} discovered is strongly proportional to the MT_{ITVL} , which means if we increase the value of MT_{ITVL} the algorithm will discover fewer A_{EASP} (see Fig. 19). This is expected because by increasing the value of MT_{ITVL} we extend the time interval during which an event can occur; hence, there is less chance of that specific event being discovered as an anomalous event against a specific sequential pattern.

We used the “Big O Notation” approach to evaluate the performance of our proposed algorithm. To evaluate the performance of the algorithm we raised the number of input events (4000 in each step) and ran the algorithm 5 times (each time with different randomly regenerated data-sets), and then calculated the average execution time in each step. The results, presented in Fig. 20, indicate that our algorithm has $O(N)$ nature, which means the algorithm execution time will grow linearly and is in direct proportion to the number of input events. We then evaluate the performance of the algorithm by increasing the length of sequence duration (SD) and MT_{ITVL} parameters in each step and calculated the execution time. As we can see from the results presented in Fig. 21 and Fig. 22, the proposed algorithm has $O(1)$ nature against SD and MT_{ITVL} parameters, which means the algorithm executes in the same time regardless of the size of the input parameters of SD and MT_{ITVL} .

All the above experiments were conducted on a 2 GHz Pentium machine with 1 GB RAM.

7. Conclusions and future work

Events occurring in observed scenes are one of the most important semantic entities that can be extracted from videos (Anwar & Naftel, 2008). Most of the work presented in the past is based upon finding frequent event patterns or deals with discovering already known abnormal events. In contrast, in this paper we presented a framework to discover unknown anomalous events associated with a frequent sequence of events (A_{EASP}); that is to discover

events which are unlikely to follow a frequent sequence of events. This information can be very useful for discovering unknown abnormal events and can provide early actionable intelligence to redeploy resources to specific areas of view (such as PTZ cameras or attention of a CCTV user). Discovery of anomalous events against a sequential pattern can also provide business intelligence for store management in the retail sector. A comprehensive and flexible problem definition framework is presented. This is followed by formulation of an efficient event mining framework to discover all the anomalous events against all the given sequential patterns (A_{EASP}) in one database scan. The proposed event mining framework also takes the temporal aspect of A_{EASP} into consideration, that is to discover anomalous events which are true for a specific time interval only and might not be an A_{EASP} over a whole time spectrum and vice versa. 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}) was also proposed. We then conducted different experiments to evaluate the proposed algorithm’s accuracy and performance. The experiment results show that the proposed algorithm detected all the instances of different sequential patterns and discovered all the true A_{EASP} successfully. We used the “Big O Notation” approach to evaluate the performance of our proposed algorithm. We raised the complexity of the algorithm by increasing the number of input events in each step. The results show that the proposed algorithm has $O(N)$ nature (algorithm execution time will grow linearly in direct proportion to the number of input events). We then raised the complexity of the algorithm by increasing the length of sequence duration (SD) and maximum interval MT_{ITVL} parameters. The results of the experiments indicate that the proposed algorithm has $O(1)$ nature against the SD and MT_{ITVL} parameters (algorithm executes in the same time regardless of the size of the input parameters of SD and MT_{ITVL}).

The following provide some of directions, which could be interesting to explore as an extension to the research work presented in this paper.

Although the proposed A_{EASP} discovery algorithm takes temporal aspects into consideration and can discover A_{EASP} with periodicity, it only discovers the periodicity of A_{EASP} on the granularity level at which the data sequences were divided. The proposed A_{EASP} discovery framework can be extended by providing a flexible mechanism in which the user can define the required periodicity, irrespective of the temporal granularity at which data sequences are divided. For example, if the detected events dataset is segmented on an hourly basis (data sequences), the user might be interested in different periodicity levels such as daily, shift-wise, weekly, monthly, etc.

In this paper we applied data mining techniques on already detected events to extract the hidden information from them. We mainly concentrated on event level, as events are collection

of entities and actions with temporal and sequential properties, therefore discovery of patterns on entities and actions level can be interesting to explore. Although discovering such patterns can be an expensive process, as the search space will increase depending upon the number of entities and actions in different events, patterns discovered at a lower level can be more precise and provide useful information.

Appendix A

A.1. Abbreviations and acronyms

A_{EASP}	anomalous events against a sequential pattern
SP	sequential pattern
min_supp	minimum support
SD	sequence duration
SP_{SET}	set of sequential patterns
$CL_{A_{EASP}}$	list of candidate anomalous events against a sequential pattern
GR	granularity
TP	time period
DS_{GR}	data sequence temporal granularity
MT_{ITVL}	maximum time interval
M_{TOL}	maximum tolerance
SP_{ET}	sequential pattern end-time
A_{EASP_ET}	anomalous events against a sequential pattern end-time
A_{EASP_ST}	anomalous events against a sequential pattern start-time
P_{DS}	previous data sequence
C_{DS}	current data sequence
$F_{DS_SP}(P_{SD_P})$	flying data sequence start position from P_{DS}
$F_{DS_EP}(C_{SD_P})$	flying data sequence end position from C_{DS}
F_{DS_L}	flying data sequence length
P_{SD_ET}	previous data sequence end time
C_{SD_ST}	current data sequence start time
C_{SD_ET}	current data sequence end time
D_{SPS_SM}	dynamic sequential patterns search mechanism
DS_{LEN}	data sequence length
SD_{LEN}	sequence duration (SD) length
C_{Event}	current event
E_{LHS}	left hand side events
E_{RHS}	right hand side events
C_{SW}	current search window
E_T	current event time
$C_{SW(ST)}$	start time of current search window
$C_{SW(ET)}$	end time of current search window
E_{LHS_LDT}	last discovered left hand side event time
F_{SW}	first search window
C_{SW}	current search window
N_{SW}	next search window
P_{SW}	previous search window
SP_{Info}	sequential pattern information data structure
SP^{SP}	sequential pattern start position
SP^{EP}	sequential pattern end position
SW_{Info}	search window information data structure
SPE_{Info}	sequential pattern events information data structure
C_{SE}	current search event data structure
SP_{DE}	sequential pattern discovered events information data structure
L^{EP}	last event position
A_{EASP_DS}	support discovery process for all $CL_{A_{EASP}}$
I_{TIME}	interested time

MT_{ITVL_ET}	end time of the user-defined parameter of MT_{ITVL}
D_{EVENTS}	already detected events dataset

References

- Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. In *Eleventh international conference on data engineering* (pp. 3–14). Taipei, Taiwan, 1995.
- Anwar, F., & Naftel, A. (2008). Video event modelling and association rule mining in multimedia surveillance systems. In *Fifth international conference on visual information engineering* (pp. 426–431). Xi'an, China, 2008.
- Anwar, F., & Petrounias, I. (2005). 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.
- Anwar, F., Petrounias, I., Morris, T., & Kodogiannis, V. (2010). Discovery of events with negative behavior against given sequential patterns. In *Fifth International IEEE conference on intelligent systems (IS '10)*. London, UK, 2010.
- Chen, X. (1999). *Temporal data mining: algorithms, language and system for temporal association rules* (Ph.D. Thesis). Department of Computing and Mathematics Manchester, UK, Manchester Metropolitan University, 1999.
- Chen, L., Faudemay, P. (1997). Multi-criteria video segmentation for TV news. In *First multimedia signal processing workshop*, NJ, USA, 1997 (pp. 319–324).
- Chen, S., Shyu, M., Zhang, C., & Strickrott, J. (2001). Multimedia data mining for traffic video sequence. In *Proceeding international workshop on multimedia data mining (MDM/KDD)*, San Francisco, USA, 2001 (pp. 78–86).
- Chen, S. C., Shyu, M. L., Zhang, C., Luo, L., & Chen, M. (2003). Detection of soccer goal shots using joint multimedia features and classification rules. In *Fourth international workshop on multimedia data mining (MDM/KDD2003)* (pp. 36–44). Washington DC, USA, 2003.
- Chen, M., Chen, S. C., & Shyu, M. L. (2007). Hierarchical temporal association mining for video event detection in video databases. In *IEEE 23rd international conference on data engineering workshop*, 2007 (pp. 137–145).
- Cornells, C., Peng, Y., Xing, Z., & Guoqing, C. (2006). Mining positive and negative association rules from large databases. *IEEE Conference on Cybernetics and Intelligent Systems*, 1–6.
- Cucchiara, R., Piccardi, M., & Mello, P. (2000). Image analysis and rule-based reasoning for a traffic monitoring system. *IEEE Transactions on Intelligent Transportation Systems*, 1, 119–130.
- Dailey, D. J., Cathey, F. W., & Pumrin, S. (2000). An algorithm to estimate mean traffic speed using uncalibrated cameras. *IEEE Transactions on Intelligent Transportation Systems*, 1, 98–107.
- Dieter, K., Joseph, W., & Jitendra, M. (1994). Robust multiple car tracking with occlusion reasoning. In *Proceedings of the third European conference on Computer vision (ECCV'94)* (pp. 189–196). Stockholm, Sweden, 1994.
- Ghanem, N., Doermann, D., Davis, L., & DeMenthon, D. (2004). Mining tool for surveillance video. In *Sixth international symposium on electronic imaging, storage and retrieval methods and applications for multimedia (SPIE'04)*, 2004 (pp. 259–270).
- Hamid, R., Johnson, A., Batta, S., Bobick, A., Isbell, C., Coleman, G. (2005). Detection and explanation of anomalous activities: Representing activities as bags of event n-grams. In *IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, 2005 (pp. 1031–1038).
- Hamid, R., Maddi, S., Bobick, A., & Essa, M. Structure from statistics – unsupervised activity analysis using suffix trees. In *IEEE 11th international conference on computer vision*, 2007 (ICCV'07) (pp. 1–8).
- Hanning, Z., & Kimber, D. (2006). Unusual event detection via multi-camera video mining. In *18th International conference on pattern recognition*, 2006 (ICPR'06), 2006 (pp. 1161–1166).
- Hauptmann, A. G., & Witbrock, M. J. (1998). Story segmentation and detection of commercials in broadcast news video. In *Proceedings of the advances in digital libraries conference (ADL'98)*, 1998 (p. 168).
- Hauptmann, A. G., & Smith, M. A. (1995). Text, speech and vision for video segmentation: The informedia project. In *AAAI fall symposium, computational models for integrating language and vision*, 1995 (pp. 10–12).
- Heikki, M., Hannu, T., & Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1, 259–289.
- Huang, T., Koller, D., Malik, J., Ogasawara, G., Rao, B., Russell, S., et al. (1994). Automatic symbolic traffic scene analysis using belief networks. In *Proceedings of the 12th national conference on Artificial intelligence (AAAI'94)* (pp. 966–972). Seattle, Washington, United States, 1994.
- Jiawei, H., Jian, P., Behzad, M.-A., Qiming, C., Umeshwar, D., & Mei-Chun, H. (2000). FreeSpan: Frequent pattern-projected sequential pattern mining. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 355–359). Boston, Massachusetts, United States.
- Jung-Hwan, O., JeongKyu, L., & Sanjaykumar, K. (2003). Real time video data mining for surveillance video streams. In *Seventh Pacific-Asia conference on knowledge discovery and data mining (PAKDD'03)*, 2003 (pp. 222–233).
- Kamijo, S., Matsushita, Y., Ikeuchi, K., & Sakauchi, M. (2000). Traffic monitoring and accident detection at intersections. *IEEE Transactions on Intelligent Transportation Systems*, 1, 108–118.

- Kazienko, P. (2008). Mining sequential patterns with negative conclusions. In *Proceedings of the 10th international conference on data warehousing and knowledge discovery (DaWaK '08)* Turin (pp. 423–432). Italy: Springer-Verlag.
- Ken-Hao, L., Ming-Fang, W., Chi-Yao, T., Yung-Yu, C., & Ming-Syan, C. (2008). Association and temporal rule mining for post-filtering of semantic concept detection in video. *IEEE Transactions on Multimedia*, 10, 240–251.
- Lawrence, Y. D., & Yi-Jen, L. (2008). Semantic analysis and video event mining in sports video. In *22nd International conference on advanced information networking and applications – Workshops (AINAW '08)* (pp. 1517–1522). IEEE Computer Society.
- Maria-Luiza, A., & Osmar, R. Z. (2004). 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* (pp. 27–38). Pisa, Italy.
- Mohan, R. (1996). Text-based search of TV news stories. In *Proceedings, multimedia storage and archiving systems, NJ, USA, 1996* (pp. 2–13).
- Norris, C., McCahill, M., & Wood, D. (2004). Editorial. The Growth of CCTV: A global perspective on the international diffusion of video surveillance in publicly accessible space. *Surveillance & Society*, 2, 110–135.
- Oh, J., & Bandi, B. (2002). Multimedia data mining framework for raw video sequences. In *ACM third international workshop on multimedia data mining (MDM/KDD2002)* (pp. 1–10). Edmonton, Alberta, Canada, 2002.
- Özarak, H., & Yazıcı, A. (2006). Structural and event based multimodal video data modeling. In *Advances in Information Systems (ADVIS)* (pp. 264–273). Izmir, Turkey, 2006.
- Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., et al. (2001). PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *17th International conference on data engineering (ICDE'01)*, 2001 (pp. 215–224).
- Shirahama, K., Ideno, K., & Uehara, K. (2007). A time-constrained sequential pattern mining for extracting semantic events in videos. In *Book of multimedia data mining and knowledge discovery* (pp. 404–426). London: Springer.
- Shu-Ching, C., Mei-Ling, S., Chengcui, Z., & Kashyap, R. L. (2000). Object tracking and multimedia augmented transition network for video indexing and modeling. In *Proceedings of the 12th IEEE international conference on tools with artificial intelligence (ICTAI 2000)* (pp. 250–257).
- Shu-Ching, C., Mei-Ling, S., & Chengcui, Z. (2001). An unsupervised segmentation framework for texture image queries. In *Proceedings of the 25th international computer software and applications conference on invigorating software development (COMPSAC'01)* (pp. 569–573).
- Shu-Ching, C., Mei-Ling, S., & Chengcui, Z. (2001). An intelligent framework for spatio-temporal vehicle tracking. In *Proceedings 2001 IEEE intelligent transportation systems* (pp. 213–218).
- Srikant, R., & Agrawal, R. (1996). Mining sequential patterns: Generalizations and performance improvements. In *Proceeding 5th of the international conference on extending database technology (EDBT)*, 1996.
- Teredesai, A., Ahmad, M., Kanodia, J., & Gaborski, R. (2006). CoMMA: A framework for integrated multimedia mining using multi-relational associations. *Knowledge and Information Systems*, 135–162.
- Toshev, A., Bremond, F., & Thonnat, M. An APRIORI-based method for frequent composite event discovery in videos. In *Proceedings of the fourth IEEE international conference on computer vision systems, 2006* (p. 10).
- Tseng, V. S., Ja-Hwung, S., & Jhih-Hong, H. (2006). A novel video annotation method by integrating visual features and frequent patterns. In *Proceeding of the seventh international workshop on multimedia data mining (KDD/MDM)*. Philadelphia, PA, USA, 2006.
- Tseng, V. S., Ja-Hwung, S., Jhih-Hong, H., & Chih-Jen, C. (2008). Integrated mining of visual features speech features and frequent patterns for semantic video annotation. *IEEE Transactions on Multimedia*, 10, 260–267.
- Turaga, P. K., Veeraraghavan, A., & Chellappa, R. (2007). From videos to verbs: Mining videos for events using a cascade of dynamical systems. In *IEEE computer society conference on computer vision and pattern recognition (CVPR'07)*.
- Valery, A. P. (2005). Mining rare and frequent events in multi-camera surveillance video using self-organizing maps. In *Proceedings of the 11th ACM SIGKDD international conference on knowledge discovery in data mining, Chicago, Illinois, USA, 2005* (pp. 794–800).
- Xie, L., Sundaram, H., & Campbell, M. (2008). Event mining in multimedia streams. *Proceedings of the IEEE*, 96, 623–647.
- Xindong, W., Chengqi, Z., & Shichao, Z. (2004). Efficient mining of both positive and negative association rules. *ACM Transactions on Information Systems (TOIS)*, 22, 381–405.
- Xingquan, Z., & Xindong, W. (2003). Sequential association mining for video summarization. In *Proceedings of IEEE international conference on multimedia & expo (ICME'03)*, Baltimore, MD, 2003 (pp. 333–336).
- Zaki, M. J. (2001). SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning Journal special issue on Unsupervised Learning*, 4, 31–60.
- Zhang, H., Tan, S. Y., Smoliar, S. W., & Yihong, G. (1995). Automatic parsing and indexing of news video. *Multimedia Systems*, 2, 256–266.
- Zhu, X., Wu, X., Elmagarmid, A. K., Feng, Z., & Wu, L. (2005). Video data mining: Semantic indexing and event detection from the association perspective – appendices. *IEEE Transactions on Knowledge and Data Engineering*, 665–667.