Journal of Biomedical Informatics 78 (2018) 43–53

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Patient ranking with temporally annotated data

Luca Bonomi*, Xiaoqian Jiang

Department of Biomedical Informatics, University of California, San Diego, United States

ARTICLE INFO

Keywords:
Sequential patterns
Temporal data
Data mining
EHR data

ABSTRACT

Modern medical information systems enable the collection of massive temporal health data. Albeit these data have great potentials for advancing medical research, the data exploration and extraction of useful knowledge present significant challenges. In this work, we develop a new pattern matching technique which aims to facilitate the discovery of clinically useful knowledge from large temporal datasets. Our approach receives in input a set of temporal patterns modeling specific events of interest (e.g., doctor's knowledge, symptoms of diseases) and it returns data instances matching these patterns (e.g., patients exhibiting the specified symptoms). The resulting instances are ranked according to a significance score based on the *p*-value. Our experimental evaluations on a real-world dataset demonstrate the efficiency and effectiveness of our approach.

1. Introduction

Nowadays, the large amount of temporally annotated data being collected within healthcare institutions holds great promises for many medical applications. For example, the analysis of temporal data associated with patients electronic health records (EHRs) can be used to facilitate early diagnosis and clinical decision-making. While these data may be beneficial for many research studies, the extraction of useful knowledge is often very challenging. Over the last decades, a variety of techniques have been proposed by the research community to model and extract useful information. For example, temporal *pattern mining* techniques [1,2] aim to extract useful information from the data in the form of patterns which model multiple events that occur sequentially within a short period of time. While these approaches have been shown to be effective in many applications, their extraction criterion is solely based on the input data (e.g., frequency, significance) and therefore are unable to take into consideration external domain knowledge (e.g., doctor's knowledge). As a consequence, data that are clinically relevant may still remain hidden. To facilitate the discovery of such information, *pattern matching* approaches (e.g., temporal query languages) have been proposed [3,4]. These techniques receive in input a set of parameters defining the extraction criteria and return the data instances satisfying them. The works in [5–7] are examples of pattern matching techniques developed to identify time-series data that match user-defined patterns.

In this work, we propose a novel pattern matching approach that aims to identify temporally annotated sequences that match user-defined patterns of interest and are statistically significant. With our solution, clinicians can use temporal medical data, such as EHRs, to conduct patients similarity search and identify data instances to support their decisions. In general, data driven decisions are central in modern healthcare and have the potential for advancing medical research and saving patient lives. For example, a study by Frankovich et al. [8] discussed how a 13-year-old girl admitted to the hospital with severe inflammation of the kidneys and pancreas was helped by evidence generated by 98 pediatric lupus patients with similar conditions. Furthermore, the retrieval of relevant patient data can also be utilized to improve the understanding of rare phenotypes. For example, in the case of Kawasaki disease the identification of children exhibiting symptoms such as fever for more than 5 days, or fever in conjunction with cervical lymphadenopathy would help local pediatricians to separate patients at risk of Kawasaki from those who are affected by common diseases such as measles [9].

The search for relevant temporally annotated sequences in the healthcare domain presents novel challenges. First, the temporal sequences modeling the medical data have high dimensionality (e.g., thousand of diagnosis codes) and due to the different hospitalization time these sequences are asynchronous (e.g., different sample rate). However, traditional pattern matching techniques often rely on spatial indexes such as R^*-tree which suffer from the curse of dimensionality [10]. Moreover, to facilitate the identification of useful data (e.g., patients exhibiting certain symptoms) it is crucial to rank the sequences matching the patterns so that the most significant data are reported first. Unfortunately, current pattern matching techniques report all the sequences matching the given patterns in input regardless their significance.

In this work, we address the aforementioned challenges and we

* Corresponding author.
E-mail addresses: lbonomi@ucsd.edu (L. Bonomi), x1jiang@ucsd.edu (X. Jiang).

https://doi.org/10.1016/j.jbi.2017.12.007
Received 18 January 2017; Received in revised form 23 September 2017; Accepted 13 December 2017
Available online 19 December 2017
1532-0464/

make the following contributions. First, we develop an efficient graph-matching model that enables the search of temporally annotated sequences matching the input patterns. In this method, we use a compact multi-partite graph to represent the original sequences where the input patterns are associated with paths in the graph. In this formulation, the sequences matching the input patterns can be efficiently determined. Second, we propose a formal method to compute the statistical significance of each sequence and rank the matching results. In our approach, we use the notion of p-value computed under a null-hypothesis model to evaluate and rank the significance of each sequence matching the input patterns. Lastly, we evaluate our approach on temporally annotated sequences obtained from a real-world medical dataset. Our evaluations demonstrate the efficiency of our technique and its efficacy in identifying useful temporal sequences.

1.1. Related work

Here, we present the works most relevant to the problem considered in this paper.

Temporal Pattern Extraction. These techniques aim to extract useful patterns from the data in an unsupervised manner thus without relying on any prior hypothesis. A popular criterion consists in extracting patterns that have large *support* (i.e., commonly shared by many sequences) [11,1,12–14]. Despite the importance of these works, they tend to extract too many patterns that often are not informative. To reduce the number of candidate patterns, recent techniques use an extraction criterion based on *significance* [15–17,2,18–20]. The significance of the extracted patterns can be determined in different ways. For example, via a statistical model based on the independence assumption [20] or using a Markov model as in [19].

Temporal Query Languages and Pattern Matching. These solutions receive in input a set of parameters (e.g., similarity threshold) and aim to identify useful data instances satisfying these parameters (e.g., patterns satisfying the specified similarity threshold). Among these approaches, *temporal query language* techniques aim to extend traditional database languages to incorporate the time domain so that temporal queries can be issued to extract useful information from the data. Popular languages, such as TSQL2 [3] and TQuel [4], provide primitive functions to perform point-based operations on temporal databases (e.g., events x and y happened at the same time). While these approaches overcome the syntactic limitation of static query languages (e.g., SQL), they offer limited flexibility for solving complex tasks, for example searching for multiple events over time intervals, or events occurring with some degree of approximation. Recently, a plethora of techniques has been developed to enable *temporal pattern matching* in time-series data [5–7]. Specifically, these approaches aim to identify those sequences that match a set of user-defined patterns in input. This search is often performed using a variety of similarity measures (e.g., Dynamic Time Warping Distance [21], Longest Common Subsequence [22]) which allow patterns to occur with some degree of approximation. Despite the efficiency of these techniques and their robustness against noise in the data, these solutions are not suitable in our setting for several reasons. First, they are often designed for numeric time-series data (e.g., biomedical signals such as blood pressure, temperature) where the data search is performed using efficient spatial–temporal indexes (e.g., R^*-tree) that cannot be easily adapted to index symbolic high dimensional temporal data (e.g., thousand of diagnosis codes). Second, these solutions report all the sequences matching the patterns without considering their significance with respect to the input patterns. As a consequence, human review is often required to further separate the useful data from the overall reported sequences.

Our proposed solution belongs to the temporal pattern matching category because it retrieves patient temporal sequences exhibiting the user specified patterns. With this work, we aim to address the limitations of current temporal pattern matching solutions. Specifically, we propose an efficient approach for matching the patterns in high dimensional data. Furthermore, we design a statistical test that determines the significance of each matching sequence with respect to the input patterns. By ranking the results according to their significance, we enable the user (e.g., doctors, clinicians) to focus on the most statistically significant data instances (e.g., patients exhibiting conditions of interests).

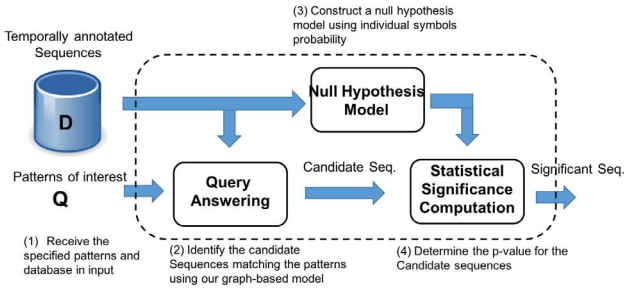


Fig. 1. Overview of our framework. (1) A query containing a set of patterns and a dataset D of temporally annotated sequences are received in input. (2) Our graph-matching model is used to identify all the candidate sequences matching the specified patterns. (3) A null-hypothesis model is constructed from the individual symbol probabilities in the data D. (4) For each candidate sequence a statistical score under a null-hypothesis model is computed and all the significant sequences are reported to the user.

1.2. Overview of our approach

In our solution, we identify two major steps: *query answering* and *statistical significance computation* which aim to first find the sequences matching the input query and then identify those that are statistically significant. The overall framework is illustrated in Fig. 1. Below, we provide a brief description of these steps.

- **Query Answering**. Given a dataset D and a set of patterns specified in the input query Q, we aim to identify those sequences in D that contains some occurrences of the patterns, named *candidate sequences*. Due to the high dimensionality and temporality of the data, there are a large number of combinations in which patterns can appear in the sequences. Hence, this step is computationally intensive in general. To this end, we propose a compact graph-model, where the occurrences of each pattern are associated with paths in the graph. We will demonstrate that the search of the sequences matching the input query with this representation incurs small memory and computational cost.
- **Statistical Significance Computation**. In this phase, we aim to determine a statistical significance score for the candidate sequences using the p-value computed according to a *null-hypothesis model*. In this model, we use an assumption of global independence for the symbols in the original dataset. Let $|\lambda|$ be the number of patterns matching a candidate sequence $\mathscr{S}$, our idea is to determine the statistical significance of $\mathscr{S}$ (i.e., p-value), by computing the probability to observe more than $|\lambda|$ patterns in a sequence generated under our null-hypothesis model. We use the p-value because it provides a better separation across the sequences compared to just using the number of patterns matching the sequence $|\lambda|$. In fact, as pointed out in Fig. 2, we observe that even though some sequences support all the specified patterns they may not be statistical significant as others. Therefore, using the p-value, we are able to identify the significant sequences from the overall candidate sequences. Given a user-defined significance threshold (i.e., th_p), all the sequences satisfying such a threshold (i.e., p-value $\leqslant th_p$) are ranked and returned as *significant sequences*.

2. Materials and methods

In this section, we first report the notions used for representing

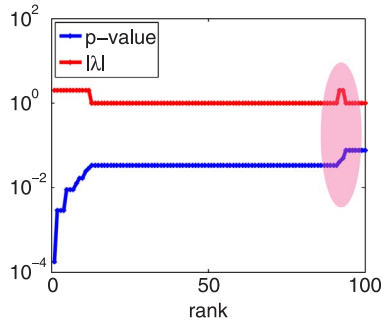


Fig. 2. Ranking results for top-100 sequences: *p*-value (blue), and number of matching patterns denoted with |λ| (red). Ranking according the *p*-value highlights the difference in significance among the sequences even though they match the same number of patterns. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

temporally annotated data and formally define the problem studied in this paper. Then, we provide a detailed description of the core steps of our solution.

2.1. Definitions and problem formulation

Here, we provide the basic definitions and notions used throughout the paper.

2.1.1. Temporally annotated data representation

In the healthcare domain, data are often very complex modeling events in an high dimensional space that have a temporal component. For example, in a hospital, each time a patient is hospitalized, a set of medical features, such as diagnosis codes, lab tests, and medications are recorded to represent the patient's medical condition over time. In our approach, each medical feature recorded at the time of visit is abstracted with an item x_i that belongs to a finite alphabet Σ of size M. For example, each diagnosis code can be encoded into a symbol where the size of the overall alphabet comprises all the codes in the ICD9 encoding schema. Since, multiple features may be recorded at each visit, all the items are collected in an *itemset*, denoted as $\mathscr{I}_j = \{x_1,...,x_n\}$, which is associated with a time stamp t_j storing the time of the visit. In this representation, the sequence of visits from the same patient describes the evolution of his/her medical condition over time. To model the temporal data, we use the notion of *temporally annotated sequences* as introduced in [12,23].

Definition 1 (*Temporally Annotated Sequence*). A temporally annotated sequence $\mathscr{S}$ of length $n > 0$, is a pair $\mathscr{S} = (S,T)$, where $S = \langle \mathscr{I}_1,...,\mathscr{I}_n \rangle$ defines a sequence of events represented as itemsets (e.g., diagnosis codes) and $T = \langle t_1,...,t_n \rangle$ denotes the temporal annotation of the events (i.e., time of visit), where $t_i < t_{i+1}$ for all i. The temporally annotated sequence $\mathscr{S}$ will be also represented as: $\mathscr{S} = \langle \{\mathscr{I}_1;t_1\},...,\{\mathscr{I}_n;t_n\} \rangle$ throughout the rest of the paper.

An example of a dataset with three temporally annotated sequences is reported in Table 1.

A variety of representations for temporal data have been proposed in different domain applications [24,25]. For example, in the work by Günther and van der Aalst [25], the notion of *trace* is proposed to model sequences of events (e.g., activities) in logs. In our setting, we use the notion of temporally annotated sequences to model temporal medical data because it has several advantages. For example, our notion allows us to capture concurrent events (e.g., diagnosis and medications occurring at the same time) and to represent recurring events over time (e.g., chronic disease, repetitive medications/prescriptions) that otherwise would not be possible with standard sequential data representations (e.g., data logs).

Table 1
Example of dataset containing three temporally annotated data sequences. The medical events are mapped into symbols for simplicity.

ID	Temporally annotated sequence
S_1	$\langle \{(a);2\},\{(a,b);5\},\{(a,c);6\},\{(b,c);10\},\{(a,b,c);15\} \rangle$
S_2	$\langle \{(b);4\},\{(a,c);8\},\{(c);20\} \rangle$
S_3	$\langle \{(a,b);2\},\{(a,c);3\},\{(c);5\},\{(b,c);6\} \rangle$

To represent evolving medical events, we use the notion of *temporal patterns*. Intuitively, a pattern represents a sequence of multiple events that occur within a short period of time. Using these patterns, we enable clinicians to represent medical conditions of interest (e.g., evolving symptoms of a specific disease) that can be used as a criterion for identifying patients. To find the temporally annotated sequences that contain the specified patterns, we use the notion of *temporal match* as in [12,23].

Definition 2 (*Temporal Matching*). Given a pattern $P = \langle \mathscr{X}_1,...,\mathscr{X}_k \rangle$ and a temporal constraint $\delta t \geqslant 0$, we say that P occurs in the temporally annotated sequence $\mathscr{S} = \langle \{\mathscr{I}_1;t_1\},...,\{\mathscr{I}_n;t_n\} \rangle$ if there exists a sequence of indexes $i_1 < i_2 < \cdots < i_k$ such that:

1. $\mathscr{X}_{ij} \subseteq \mathscr{I}_{ij}$ for all $j = 1,...,k$
2. $t_{i_k} - t_{i_1} \leqslant \delta t$

Modeling Temporal Relationships between Events.Temporal associations across medical events carry useful information that can be beneficial in many applications such as clinical decision-making [26] and in defining guidelines [27]. However, capturing the temporal relationships among medical events is a very challenging task which has lead to a variety of temporal models over the years [28]. Among them, Allen's rules [29] have been widely applied to enable the use of data mining techniques on temporal data [30–32]. In that model, a set of temporal rules is defined for each pairwise time intervals in the data. While that model captures all the relationships between events, it may become unpractical in real scenarios as too many rules are often generated. Furthermore, those rules may be ambiguous and in some cases lead to inconsistency in the results. Compared to that model, the representation of medical events with patterns tends to be more compact and robust. In fact, the patterns are uniquely identified by the sequential order of their itemsets without the need of considering the relationships between each pair of events.

In our model, it is important to define a suitable value for the temporal constraint δt to identify the matching sequences. In principle, different values of δt may lead to different matches. This time parametrization is a common challenge in modeling temporal data. For example, Hripcsak et al. [33] proposed a technique for temporal modeling based on a constraint satisfaction problem that has shown promising results for discharge summaries. In our case, we leave the choice of δt to the user as the best setting for δt often depends on the domain specific application.

2.1.2. Problem formulation

With the notions and definitions presented above, we are now ready to formulate our problem. In our approach, we consider an input query $\mathscr{Q} = (Q,\delta t)$, where Q is a collection of m patterns $Q = \{P_1,P_2,...,P_m\}$ expressing medical events of interest, $\delta t > 0$ is a time constraint for the medical events in these patterns, and D is a dataset of temporally annotated sequences. Then, our *goals* are: (1) find all the sequences in D that match the patterns in Q within the temporal constraint δt, and (2) rank the matching sequences in terms of their statistical significance. In this way, we aim to identify those patients that are most statistically significant with respect to the input patterns and may be clinically useful for the clinicians.

It is important to consider multiple patterns in the input query because they allow to model complex medical conditions that may be a result of multiple medical events experienced by the patients.

Table 2
Table of common symbols.

Symbols	Description
$P_1,...,P_m$	Patterns
$x_1,x_2,...,x_M$	Individual items
$x_1^i,x_2^i,...,x_M^i$	Bernulli random variables
p_j	Prob. of an item x_j
$\mathscr{I}_1,...,\mathscr{I}_n$	Itemsets for sequences
$\mathscr{X}_1,...,\mathscr{X}_k$	Itemsets for patterns
$\bar{I}_1,...,\bar{I}_n$	Random vector associated with itemsets
$\bar{\lambda}(S)$	Vector of answer for the sequence S
$p_{i,j}$	Prob. of $\mathscr{X}_i$ to match the jth itemset ($Pr[\mathscr{X}_i \subseteq \mathscr{I}_j]$)
$pf_i(j)$	Prob. of $\mathscr{X}_i$ to first occur on the jth itemset
ps_i	Prob. of P_i to occur

Furthermore, the interaction between patterns (e.g., how many occur on the same sequence) may provided useful indications about their importance in modeling the events of interest.

2.2. Methods

In this section, we describe the core components of our approach as illustrated in Fig. 1. We first present our graph-matching model for identifying the temporally annotated sequences matching the input query. Then, we present our null-hypothesis model and describe a dynamic programming algorithm for computing the significance of the temporally annotated sequences matching the patterns in Q. A summary of the common symbols used in the description of our approach is reported in Table 2.

2.2.1. Query answering: Matching temporally annotated sequences

The first step in our solution, as illustrated Fig. 1, consists in the identification of the sequences containing some occurrences of the patterns specified in the input query $Q = \{P_1,P_2,...,P_m\}$. In principle, one can scan each sequence in D and for each temporal window of length δt consider whenever a pattern P_i has a match. However, this naïve approach requires to consider all the possible configurations in which a pattern can occur in each sequence resulting in poor efficiency. To this end, we develop a graph-matching approach which allows to identify the matching patterns and their positions in the sequences using only one pass on the data.

2.2.1.1. Graph-matching model. Our approach is inspired by the work in [34], where the occurrences of a pattern in a sequence S are determined by finding paths on a multi-partite graph constructed from S. We extend that idea to support temporal sequences where events may consist of more than one symbol.

Definition 3 (*Matching Graph*). Given an input query $\mathscr{Q} = (Q,\delta t)$ and a temporally annotated sequence $\mathscr{S} = (S,T)$, a matching graph $G_{\mathscr{Q},\mathscr{S}} = (V,E)$ is a multi-partite graph where there is a layer for each distinct itemset of the patterns in Q. Each node v_i in V maps to a pair $(\mathscr{I}_i,t_i)$ in $\mathscr{S}$ and it belongs to the layers corresponding to itemsets of the patterns that are contained in $\mathscr{I}_i$. Each edge $e = (v_i,v_j) \in E$ is labeled with a label $l(e)$ corresponding to a pattern in Q. An edge $e = (v_i,v_j)$ exists if: (1) $t_i < t_j$ and (2) the nodes v_i and v_j are placed into layers that correspond to adjacent itemsets for the pattern $P_{l(e)}$.

In this representation, we determine the existence of temporal matches for a pattern P_i on a sequence $\mathscr{S}$ by finding paths using only edges associated with P_i across layers of the graph $G_{\mathscr{Q},\mathscr{S}}$. The matching graph $G_{\mathscr{Q},\mathscr{S}}$ is constructed iteratively by processing an itemset at the time from the temporally annotated sequence $\mathscr{S}$. At each time t_i, two operations are performed: (i) some edges to nodes associated with the layers corresponding to $(\mathscr{I}_i,t_i)$ may be added, and (ii) all the nodes associated with $(\mathscr{I}_j,t_j)$ where $t_j < t_i - \delta t$ are removed together with their edges. Therefore, only the most recent δt itemsets are responsible for the nodes and edges of $G_{\mathscr{Q},\mathscr{S}}$. The layers s_i and d_i correspond to the first and last itemset in P_i respectively. Then, each occurrence of the pattern P_i in $\mathscr{S}$ is associated with a path from s_i to d_i using only edges of P_i.

Example 1 (*Running Example*). Consider the input query $\mathscr{Q} = (\{P_1,P_2\},\delta t = 10)$ and the temporally annotated sequence $\mathscr{S} = (S,T)$ illustrated in Fig. 3. The overall matching graph comprises five layers, where L_2 and L_3 are associated with P_1, while L_1, L_2, L_4 and L_5 with P_2. The layers L_2 and L_3 are the source s_1 and destination d_1 for P_1 while L_1 and L_5 are the source s_2 and destination d_2 for P_2. The algorithm examines the sequence S an itemset at the time and adds nodes in the matching layer. For example, the first itemset $\mathscr{I}_1 = \langle(a,b,e)\rangle$ produces three nodes that are placed in L_1, L_2, and L_5 since these layers correspond to itemsets that are subsets of $\mathscr{I}_1$. At time 7, the node associated with itemset $\langle(b,c)\rangle$ is placed in L_3 and an edge for P_1 is added between the current layer and its previous one (i.e., L_2). Since this edge forms a path from the source to destination of P_1, this indicates that P_1 occurs in S in the time interval [1,7]. The situation after the first four itemsets are processed is reported in Fig. 3a. When the itemset associated with time stamp 12 is processed, some of the previous nodes are removed from the layers due to the time constraint fixed to $\delta t = 10$. The resulting configuration is illustrated in Fig. 3b. The mining algorithm terminates when the last itemset $\mathscr{I}_7$ is processed. This itemset generates nodes that are placed in layers L_2, L_3, and L_5. These contribute to form a complete path from s_2 to d_2, and from s_1 to d_1, as illustrated in Fig. 3c. Therefore, forming an occurrence for both patterns.

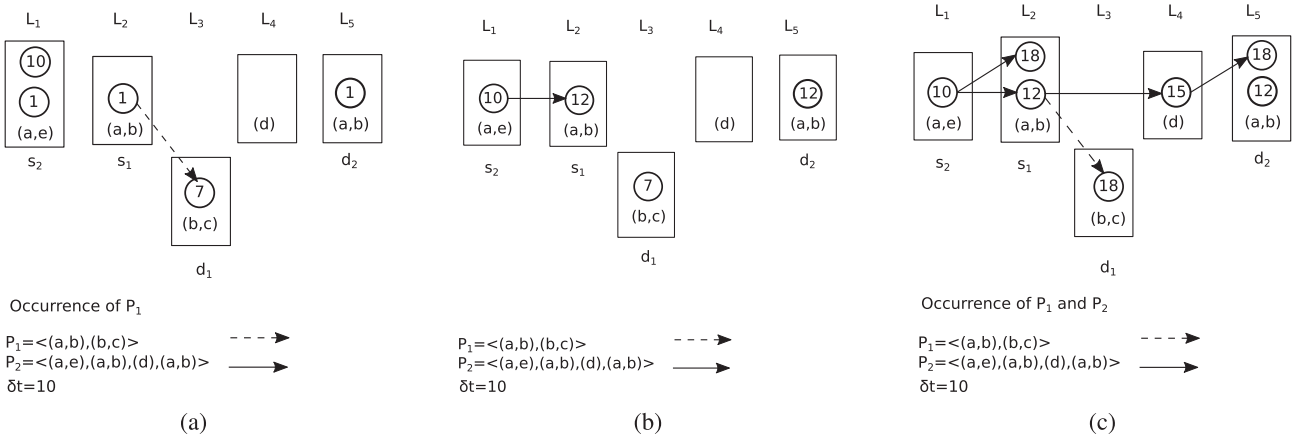


Fig. 3. Running example of the mining approach for the query $\mathscr{Q} = (\{P_1,P_2\},\delta t = 10)$ on the temporally annotated sequence $\mathscr{S} = (S,T)$, where $S = \langle(a,b,e),(c),(b,c),(a,e),(a,b),(d),(a,b,c)\rangle$ and $T = \langle1,5,7,10,12,15,18\rangle$.

Using this approach, we can efficiently determine the occurrences of each pattern specified in the input query. We denote by $\overline{\lambda}(S) = [\lambda_1,...,\lambda_m]$ the binary vector of answers where $\lambda_i = 1$ iff P_i has an occurrence in $\mathscr{S}$. For example, using the situation in Example 1, the vector of answer for S is $\overline{\lambda}(S) = [1,1]$, since both patterns specified in the query have a temporal match in S.

Complexity analysis. Let N be the total number of distinct itemsets for the patterns in the input query $\mathscr{Q} = (Q,\delta t)$. Then, we observe that the total number of layers in the matching graph $G_{\mathscr{Q},\mathscr{S}} = (V,E)$ is $O(N)$. When a new itemset $\mathscr{I}_i$ of the temporally annotated sequence $\mathscr{S}$ is processed, our approach creates a new node for each matching layers. In principle, to determine the matching layers we could use any itemset matching algorithm as a black box. For example, comparing an item at the time, this step requires $O(N \times |\Sigma|)$. To detect a path representing an occurrence of a pattern in the graph, we need a liner scan of the layers associated with the pattern. Since, this test is required each time an element is process, the total running time for determining the occurrences of the patterns in $\mathscr{S}$ is $O(|S| \times N \times |\Sigma|)$. Due to the temporal constraint, we have that the total number of nodes in V at any time is $O(\delta t \times N)$. Furthermore, since the graph is multi-partite, and each edge can be associated with at most a pattern of Q, the total number of edges in E is $O(m \times \delta t \times N)$. Hence, our approach is efficient both in terms of memory consumption and running time.

Considerations. Our approach introduces significant improvements over the pattern mining technique originally presented in [34]. First, our solution allows us to find the matching positions for patterns defined as sets of itemsets while the original technique works only with sequences of items. Second, our query representation yields to a more compact graph compared to the approach in [34]. In our case, the nodes in the graph can be shared by multiple patterns which considerably reduces the size of the overall data structure. Lastly, by enforcing the temporal constraint on the patterns, we can remove those itemsets that cannot constitute an occurrence for the patterns, which leads to a smaller number of nodes and edges in the graph.

2.2.2. Statistical significance computation

In this section, we determine the *significance* of the sequences matching the specified patterns in input. Specifically, we are interested in those sequences that exhibit a statistically significant number of patterns. Following the framework illustrated in Fig. 1, we first present our null-hypothesis model and then describe an efficient algorithm for computing the statistical significance.

From the query answering step, we have that for each temporally annotated sequence S, a binary vector $\overline{\lambda}(S) = [\lambda_1,...,\lambda_m]$ is computed. Furthermore, let $\gamma_0 = |\overline{\lambda}(S)| = \sum_{i=1}^{m} \lambda_i$ denote the norm of such a vector (i.e., how many patterns of the input query occur in S). In this representation, the *statistical significance* of a sequence $\mathscr{S}$ is defined as the probability that a random sequence $\mathscr{S}^*$ generated from our null-hypothesis model supports $\gamma \geqslant \gamma_0$ patterns, namely the p-value of $\mathscr{S}$. Using this notion, we rank the sequences that support the patterns in Q where the smaller the p-value the more statistically significant is the sequence.

To compute such a p-value, we first need to determine the probability for each pattern $P_1,...,P_m$ to appear in a random sequence generated under our null-hypothesis presented in paragraph 2.2.2.1. Due to the high dimensionality of the data, this computation is very challenging. To this end, in paragraph 2.2.2.2, we propose an efficient solution which computes such a probability using a dynamic programming algorithm. Later in paragraph 2.2.2.3, we illustrate our statistical significance test to compute the p-value for each sequence matching some of the patterns in input.

Considerations. The use of statistical significance notions in data mining and query answering has appeared in literature before. For example, in the SigSpan approach [2] the authors use a statistical test to identify significant patterns. Compared to that work, our statistical significance test is based on the p-value. As a consequence, our results can be easily interpreted since the p-value has a well known statistical meaning. Furthermore, our significance score aims to identify significant sequences with respect to a set of specific patterns in input rather than considering the overall data. In this way, we facilitate the retrieval of significant sequences that otherwise would remain hidden.

2.2.2.1. Null-hypothesis model. In our null-hypothesis model, we consider a global independence assumption as in [2]. For an itemset $\mathscr{I}_i$, we consider a random binary vector $\overline{I}_i = (x_1^i,x_2^i,...,x_M^i)$, with an entry x_j^i for each possible item in the alphabet Σ, where x_j^i is a random independent variable of Bernulli distribution of parameter p_j. Where p_j is a probability associated with the single item x_j and it can be estimated from the data. Assuming independency between the items, the probability of observing an itemset $\mathscr{I}_i$, can be computed as the product of the single item probabilities, as follows:

$$Pr[\overline{I}_i] = Pr[x_1^i] \times Pr[x_2^i] \times \cdots \times Pr[x_M^i] \tag{1}$$

Example 2. Consider the data illustrated in Table 1, where we assume an alphabet $\Sigma = \{a,b,c\}$. Then, we have that the single item probability for a is $p_a = Pr[a] = 7/12 = 0.58$, since the item a occurs in 7 itemsets out of 12. Similarly, $p_b = Pr[b] = 6/12 = 0.5$ and $p_c = Pr[c] = 8/12 = 0.67$. Given the itemset $\mathscr{I} = (a,c)$, we have that its binary vector representation is $\overline{I} = (1,0,1)$ since both items a and c are in $\mathscr{I}$ while b is missing. Then, under the global independence assumption, the probability of observing $\mathscr{I}$ in our model is $Pr[\overline{I}] = Pr[(a,c)] = Pr[a] \times Pr[c] = p_a \times p_c = 0.39$.

In general, a sequence S of length n can be represented as a stationary stochastic process $\overline{S} = \langle \overline{I}_1,\overline{I}_2,...,\overline{I}_n \rangle$ of n independent and identically distributed random vectors $\overline{I}_j$. Hence, the probability of observing a sequence S in the null model is computed as:

$$Pr[\overline{S}] = Pr[\overline{I}_1] \times Pr[\overline{I}_2] \times \cdots \times Pr[\overline{I}_n] \tag{2}$$

Although this model makes a global independence assumption (similar to Naïve Bayes), which may not hold in practice, it has several advantages. First, under the independence assumption the statistics incur limited computational cost, hence the overall statistical test is very efficient. Second, the probability for the null-hypothesis can be easily carried out in a closed form which provides a useful interpretation that can help in identifying frequent co-occurrences of items that cannot be explained by their marginal frequencies [2]. Lastly, the same independence model has been successfully applied in many itemset mining problems [2,35,36]. For these reasons, we choose to use the independence assumption for our null-hypothesis model.

2.2.2.2. Calculating the probabilities. To compute the significance of the sequences, we need to determine the probability for each pattern $P \in Q$ to occur in a sequence generated under the null-hypothesis model. Without loss of generality, let $P = \langle \mathscr{X}_1,\mathscr{X}_2,...,\mathscr{X}_k \rangle$ be a pattern in Q, with k itemsets and let $S = \langle \mathscr{I}_1,\mathscr{I}_2,...,\mathscr{I}_n \rangle$ be a random sequence generated under the null-hypothesis model. Furthermore, let l_j denote the size of $\mathscr{I}_j$ in S, then the vector $\overline{l} = [l_1,l_2,...,l_n]$ represents the size of each itemset in the sequence S. Using this notation, the probability of an itemset $\mathscr{X}_i$ to match any itemset $\mathscr{I}_j$ in S, denoted as $p_{i,j} = Pr[\mathscr{X}_i \subseteq \mathscr{I}_j]$, is computed as follows:

$$Pr[\mathscr{X}_i \subseteq \mathscr{I}_j] = \begin{cases} Pr[x_1^i] \times \cdots \times Pr[x_M^i], & \text{if } |\mathscr{X}_i| \leqslant l_j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Example 3 (*Probability of Matching an Itemset*). Consider a pattern $P = \langle (a,b),(c),(a,c) \rangle$ and a random sequence $S = \langle \mathscr{I}_1,\mathscr{I}_2,\mathscr{I}_3,\mathscr{I}_4 \rangle$ generated from the null model constructed from the data in Table 1. For simplicity, let assume that $l_j \geqslant 2$ for $j = 1,2,3,4$ (i.e., all the itemsets of P can occur in S). Then, we have that $p_{1,1} = p_{1,2} = p_{1,3} = p_{1,4} = Pr[(a,b)] = Pr[a] \times Pr[b] = 0.29$. Similarly, for the other itemsets of P, we have that $p_{2,1} = p_{2,2} = p_{2,3} = p_{2,4} = Pr[c] = 0.67$, and $p_{3,1} = p_{3,2} = p_{3,3} =$

Table 3
Possible configurations representing the matching positions for $P = \langle \mathscr{I}_1,\mathscr{I}_2,\mathscr{I}_3 \rangle$ in $S = \langle \mathscr{I}_1,\mathscr{I}_2,\mathscr{I}_3,\mathscr{I}_4 \rangle$.

	I_1	I_2	I_3	I_4
c_1	$\mathscr{X}_1$	$\mathscr{X}_2$	$\mathscr{X}_3$	
c_2	$\mathscr{X}_1$	$\mathscr{X}_2$		$\mathscr{X}_3$
c_3	$\mathscr{X}_1$		$\mathscr{X}_2$	$\mathscr{X}_3$
c_4		$\mathscr{X}_1$	$\mathscr{X}_2$	$\mathscr{X}_3$

$p_{3,4} = Pr[(a,c)] = Pr[a] \times Pr[c] = 0.39.$

Due to the global independence assumption, the probability of P to appear in S can be computed by assembling the probability of its itemsets. However, since there may be many matching configurations for the itemsets of P (i.e., exponential in k), a naïve implementation is computational prohibitive in practice.

Example 4. To illustrate the complexity of this problem, consider a random sequence $S = \langle \mathscr{I}_1,\mathscr{I}_2,\mathscr{I}_3,\mathscr{I}_4 \rangle$ and a pattern $P = \langle \mathscr{X}_1,\mathscr{X}_2,\mathscr{X}_3 \rangle$. There are four possible configurations: c_1,c_2,c_3, and c_4 representing the positions of S in which P can occur as reported in Table 3. Where, for example in configuration c_2: $\mathscr{X}_3$ matches I_4,$\mathscr{X}_2$ matches I_2 , and $\mathscr{X}_1$ matches I_1. Clearly, as the length of the pattern and sequence increase, the number of possible configurations rapidly grows.

Inspired by the work in [2], we propose a technique that computes the probability of P to occur in S without explicitly considering all these configurations. Instead, the overall probability of P to occur in S can be derived by carefully assembling the probability of each suffix of P.

Example 5. Continuing from Example 4, we observe that the probability of $P = \langle \mathscr{X}_1,\mathscr{X}_2,\mathscr{X}_3 \rangle$ to occur after a certain position j in S can be derived by combining the probabilities associated with $\mathscr{X}_1$ and the suffix $\langle \mathscr{X}_2,\mathscr{X}_3 \rangle$. For example, the probability of P to have a first occurrence at or after the second position in S can be determine by multiplying the probability of $\mathscr{X}_1$ to have the itemset $\mathscr{I}_2$ as a first match, with the probability of $\langle \mathscr{X}_2,\mathscr{X}_3 \rangle$ to have a first occurrence at or after the third position in S.

To develop this idea, we need to derive the probability of any itemset $\mathscr{X}_i$ to have a first match on a position j in S, denoted as $pf_i(j)$. Due to the independence assumption in our null model, this probability can be computed by considering the individual $p_{i,j}$, as follows:

$$pf_i(j) = \begin{cases} 0, & \text{for } j < i \\ \prod_{j'=1}^{j-1} (1-p_{i,j'}) \times p_{i,j} & \text{otherwise} \end{cases} \quad (4)$$

Example 6 (*Probability of first Occurrence*). Consider the setting from Example 3. Using Eq. (4), we compute the probabilities for the itemset (a,c) to have a first occurrence at positions 4 and 3 in S, as $pf_{3,4} = (1-p_{3,1}) \times (1-p_{3,2}) \times (1-p_{3,3}) \times p_{3,4} = (1-0.39)^3 \times 0.39 \approx 0.09$ and $pf_{3,3} = (1-0.39)^2 \times 0.39 \approx 0.15$ respectively.

Using this probability for the itemset, we can derive for each suffix of P the probability to occur after a certain position in S. Our idea is to start from the shortest suffix of $P = \langle \mathscr{X}_1,...,\mathscr{X}_k \rangle$, denoted as $suff(P)_k = \langle \mathscr{X}_k \rangle$ (i.e., single itemset), and progressively process each suffix of P by increasing their length with an itemset at a time. For each $suff(P)_i$ and position j in S, we determine the probability of $suff(P)_i$ to have a first occurrence at or after the jth itemset in S by using the already computed probabilities associated with the suffix $suff(P)_{i-1}$ and position $j+1$. We formulate this backward extension process with a dynamic programming algorithm.

Formally, let $\mathbf{P}(i,j)$ denote the probability of the suffix $\langle \mathscr{X}_i,\mathscr{X}_{i+1},...,\mathscr{X}_k \rangle$ to have a first occurrence at or after the jth itemset in S. Then, the following relationship holds:

Table 4
Example of dynamic programming for computing the probability for the pattern $P = \langle (a,b),(c),(a,c) \rangle$ to occur in $S = \langle \mathscr{I}_1,\mathscr{I}_2,\mathscr{I}_3,\mathscr{I}_4 \rangle$. The process starts from the last suffix ($suff_3(P)$) with the probability $\mathbf{P}(3,4) = 0.09$. The algorithm progressively extends each suffix an itemset at the time. The final probability is stored in the entry $\mathbf{P}(1,1) = 0.018$.

Suffixes	I_1	I_2	I_3	I_4
$suff_1(P) = \langle (a,b),(c),(a,c) \rangle$	0.018	0.001	0	0
$suff_2(P) = \langle (c),(a,c) \rangle$	0	0.056	0.007	0
$suff_3(P) = \langle (a,c) \rangle$	0	0	0.23	0.09

$$\mathbf{P}(i,j) = pf_i(j) \times \mathbf{P}(i+1,j+1) + \mathbf{P}(i,j+1) \quad (5)$$

where $\mathbf{P}(i,j) = 0$ for $j \geqslant n-k+i$ and $j < i$ (i.e. no valid positions for a match).

In this process, we start with the suffix comprising the last itemset of P (i.e., $suff_k(P)$), where we initialize $\mathbf{P}(k,n) = pf_k(n)$ (i.e., last possible matching position for $suff_k(P)$) and compute $\mathbf{P}(k,j) = pf_k(j) + \mathbf{P}(k,j+1)$ for $k \leqslant j \leqslant n$. Then using Eq. (5), we progressively extend it with an itemset at a time. Finally, we compute the entry $\mathbf{P}(1,1)$ which denotes the probability of $P = \langle \mathscr{X}_1,\mathscr{X}_2,...,\mathscr{X}_k \rangle$ to appear in S. We observe that overall this computation requires only in $O(n \times k)$ time.

Example 7 (*Running Example*). Consider the pattern $P = \langle (a,b),(c),(a,c) \rangle$, the random sequence $S = \langle \mathscr{I}_1,\mathscr{I}_2,\mathscr{I}_3,\mathscr{I}_4 \rangle$ and the probabilities of the itemsets computed in Example 3. Here, we describe the steps in our dynamic programming algorithm to compute the probability of P to occur in S as illustrated in Table 4. (i) Our algorithm starts from $suff_3(P) = \langle (a,c) \rangle$, where we initialize the probability of $suff_3(P)$ to have a first occurrence in the last itemset of S as follows: $\mathbf{P}(3,4) = pf_3(4) \approx 0.09$. Then, we compute the probability of $suff_3(P)$ to occur at or after position 3 in S, denoted as $\mathbf{P}(3,3)$. Specifically, we have that $\mathbf{P}(3,3) = pf_3(3) + \mathbf{P}(3,4) = (1-0.39)^2 \times 0.39 + 0.09 \approx 0.23$. Since P comprises 3 itemsets and the sequence has 4 itemsets, these are the only two positions where such a suffix can occur to form an occurrence of P in S. (ii) Then, the algorithm moves on processing the suffix $suff_2(P) = \langle (c),(a,c) \rangle$. The first position, moving from the right, in which such a suffix can appear is position 3. The probability $\mathbf{P}(2,3)$ of $suff_2(P)$ is obtained combining the probability of the itemset $\langle (c) \rangle$ to have a first occurrence at position 3 (i.e., $pf_2(3)$) with the probability of $suff_3(P)$ to occur at or after position 4 in S (i.e., $\mathbf{P}(3,4)$). Hence, we have that $\mathbf{P}(2,3) = pf_2(3) \times \mathbf{P}(3,4) = 0.073 \times 0.09 \approx 0.007$. The second position in which $suff_2(P)$ can form an occurrence of P in S is position 2. The probability for such a position is $\mathbf{P}(2,2) = pf_2(2) \times \mathbf{P}(3,3) + \mathbf{P}(2,3) \approx 0.056$. (iii) Finally, we use these probabilities in processing the last suffix of P (i.e., $suff_1(P)$). For the left most position of this suffix, we obtain that $\mathbf{P}(1,1) = 0.018$, which also denotes the probability of P to occur in S.

Using this procedure, we can efficiently compute the probability of each pattern P_i to occur in a random sequence. For simplicity, we will refer to this probability as ps_i in the analysis of the following section.

2.2.2.3. Computing the statistically significance. Given a binary vector of answers $\overline{\lambda}(S) = [\lambda_1,...,\lambda_m]$ for a sequence S with respect to an input query $\mathscr{Q} = (Q,\delta t)$, with m patterns $P_1,...,P_m$ and the number of patterns supported by S denoted with $\gamma_0 = |\overline{\lambda}(S)| = \sum_{i=1}^{m} \lambda_i$. We determine the statistical significance (p-value) of S by computing the probability of a random sequence S^* generated under the null-hypothesis model to support more than γ_0 patterns. Formally,

$$Pr[|\overline{\lambda}(S^*)| \geqslant \gamma_0;\overline{I}^*,Q] = \sum_{\gamma=\gamma_0}^{m} Pr[|\overline{\lambda}(S^*)| = \gamma;\overline{I}^*,Q] \quad (6)$$

We model the probability of a pattern P_i to appear in S^* as a Bernulli distribution with parameter ps_i. This probability of success depends on the pattern and the length of the sequence. Here, we make an assumption of independence among the patterns in input which is motivated by the following observation. As the patterns are individually

defined, it is likely to assume that they do not share common events, hence there is no correlation. Therefore, we have that λ follows a Poisson Binomial Distribution, where:

$$Pr[|\overline{\lambda}(S^*)| = \gamma; \overline{l}^*, Q] = \sum_{\mathbb{S} \in F_\gamma} \prod_{j \in \mathbb{S}} ps_j \prod_{i \in \mathbb{S}^c} (1 - ps_i) \quad (7)$$

with $0 \leqslant \gamma \leqslant m$. The set F_γ contains all subsets of γ integers that can be selected from {1,2,...,m} determining the patterns in Q having a positive answer in the sequence S, and $\mathbb{S}^c$ is the complements of $\mathbb{S}$.

Complexity analysis. Here, we analyze the complexity for computing the statistical significance. We start by observing that the computation of $Pr[|\overline{\lambda}(S^*)| = \gamma; \overline{l}^*, Q]$ can be challenging when a large number of patterns is considered in input (e.g., $m > 20$). However, the recursive formula in Eq. (7) is still efficient in real scenarios when m is small. In addition, we notice that there exists an efficient recursive formulation for Eq. (7) that requires only $O(\gamma \times m)$ operations [37]. Therefore, using such an implementation, the p-value computation for each candidate sequence requires $O(m \times (n \times k + m^2))$ operations which is polynomial in the size of the patterns, length of the sequence and number of patterns in Q.

In our solution, we use the p-value to represent the statistical significance of the sequences supporting the query. The user can set a p-value threshold and all the sequences below the threshold are reported in increasing order of p-value (i.e., most significant first). In practice, determining the appropriate p-value threshold could be very challenging because of the choice of the patterns in the query and the type of clinical findings associated with the retrieved sequences. Generally, there are guidelines that users can follow in selecting an appropriate threshold of p-value (e.g., 0.05, 0.01). In addition, given the specific patterns in Q the Hoeffding's concentration inequality [38] could be used to determine an upper bound on the p-value similarly to [2].

3. Results

Dataset. We conduct our experiments on the real-world medical dataset MIMIC-III[1] which is a de-identified dataset comprising over 58000 hospital admissions for 38645 adults and 7875 neonates [39]. For each patient, we construct a temporally annotated sequence where for each hospital admission we create an itemset $\mathscr{I}_j$ recording the set of ICD9 codes assigned by the medical personnel to the patient at the time of admission. Furthermore, in our temporal sequence for each itemset $\mathscr{I}_j$, we maintain the time of visit (in days) in the temporal index t_j.

From the overall sequences, we discard those with less than 2 visits. Then the resulting dataset contains 6577 sequences defined over an alphabet of 1939 symbols. Some general data statistics (i.e., minimum, average, and maximum values) are reported in Table 5, where T denotes the temporal span (i.e., for how long a patient visited the institution), s denotes the number of items in the itemsets (i.e., number of codes recorded during a visit), and n denotes the number of itemsets in the sequences (i.e., number of visits). From these statistics, we observe that the average length for the sequence (s_{avg}) is only 2.7, indicating that patients tend to have only a few visits. Furthermore, each itemset in the sequences contains on average 13.3 items, hence a large number of diagnosis codes are recorded at each visit.

Settings. The default parameters for our algorithm are reported in Table 6. With these default values we are able to simulate realistic settings in which our technique can be applied. It is reasonable to assume that clinicians may be interested in identifying patients exhibiting at least two distinct medical events that are modeled in our setting by considering two patterns ($m = 2$). As a default setting, we consider patterns with two itemsets ($l = 2$), which allow clinicians to specify related events of interest (e.g., complications of a disease). Regarding to the time gap, we select 31 days which may be suitable to capture short/medium time clinical evolutions. We believe that this choice is appropriate given the nature of the MIMIC-III dataset which comprises patients hospitalized in critical care units. In general, our solution allows the users to specify arbitrary queries. Our results are obtained as an average of at least 100 runs using a randomly generated query constructed from the data, if not specified otherwise.

[1] Available upon registration at: https://mimic.physionet.org.

Table 5
Statistics of the dataset.

Statistic	Description	Value
ΔT_{min}	Min. Tem. Span	1 day
ΔT_{avg}	Avg. Tem. Span	644 days
ΔT_{max}	Max. Tem. Span	4034 days
s_{min}	Min. Num. Items	1
s_{avg}	Avg. Num. Items	13.3
s_{max}	Max. Num. Items	39
n_{min}	Min. Seq. Len	2
n_{avg}	Avg. Seq. Len	2.7
n_{max}	Max. Seq. Len	41

Table 6
Default values for the parameters in our algorithm.

Parameter	Description	Default value
m	Num. of patterns in Q	2
th_p	p-value threshold	10^{-3}
l	Length of the patterns in Q	2
δt	Temporal constraint	31 days

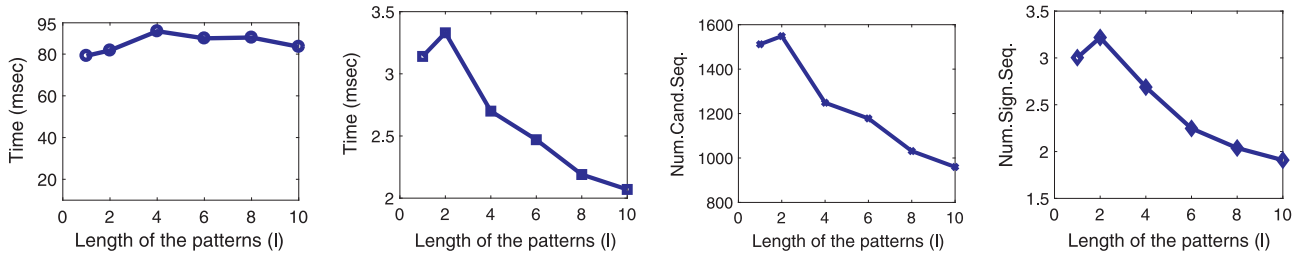
Our experimental evaluations are organized as follows. First, we study the efficiency of our solution and its efficacy in a general setting. Second, we consider a specific task in which we demonstrate how our solution can be used to search for target patients. Finally, we consider two medical tasks where our approach is used to identify outliers and retrieve similar patients.

3.1. General evaluations

Here, we evaluate our approach using the following metrics: (i) *running time* (measured in milliseconds), (ii) *number of candidate sequences* (Num.Cand.Seq) (i.e., sequences matching the input patterns), and (iii) *number of statistically significant sequences* (Num.Sign.Seq) (i.e., sequences reported as output). Regarding the running time, we report both the time required to generate the candidate sequences (i.e., query answering phase) and to determine their statistical significance (i.e., statistical test).

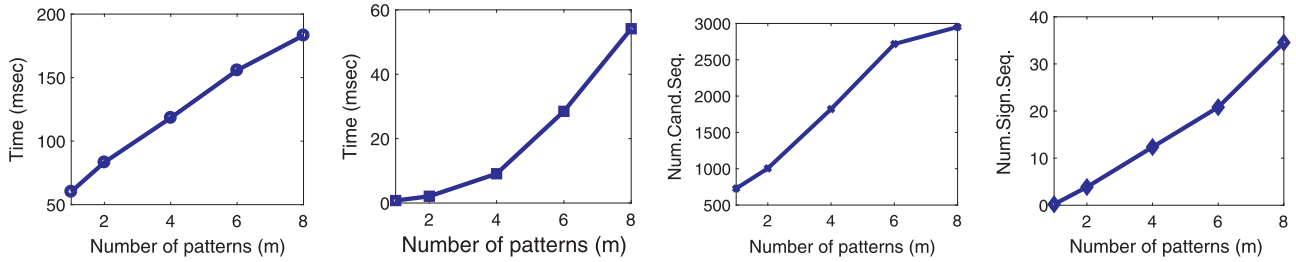
Impact of the length of patterns (l). We vary the length of the patterns (l) in the input query. From Fig. 4a, we observe that increasing the length of patterns has limited impact on the running time of our graph-matching model for answering the query. This result demonstrates the effectiveness of our compact graph representation in creating the layers as the length of the patterns increases. Hence, our solution scales well with respect to this parameter. The running time for the statistical test is reported in Fig. 4b. First, we observe that this running time is considerably smaller than the time required to identify the matching sequences (at least one order of magnitude). Second, it decreases as the length of the patterns increases due to the fact that fewer candidate sequences match the query, as reported in Fig. 4c. Intuitively, longer patterns enforce stronger constraints on the sequences, making the matching harder. As a result, fewer sequences match the query. Similarly, from Fig. 4d, we observe that the number of significant sequences reported by our approach decreases as l increases.

Impact of the number of patterns (m). When we vary the number of patterns (m) in the query we observe higher running times both in the query answering phase (Fig. 5a) and statistical test (Fig. 5b). Specifically, we notice that for the query answering phase the running time



(a) Time for matching the sequences (b) Time for computing the statistical score (c) Number of sequences matching the patterns (d) Number of significant sequences

Fig. 4. Impact of the length of the patterns (l) in the input query.



(a) Time for matching the sequences (b) Time for computing the statistical score (c) Number of sequences matching the patterns (d) Number of significant sequences

Fig. 5. Impact of the number of patterns (m) in the input query.

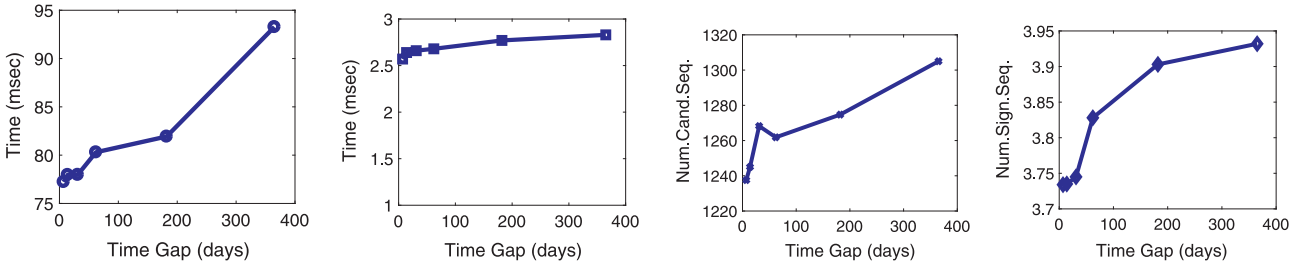
grows linearly as m increases, while for the statistical test this dependency is polynomial. These results are in line with the analysis conducted in the previous sections. From Fig. 5c, we observe that the number of candidate sequences increases as more patterns are included in the input query. In fact, when more patterns are included in the query it is more likely for a sequence to be considered as a candidate since it is sufficient to match any of the specified patterns. Consequently, the overall number of significant sequences increases as shown in Fig. 5d.

Impact of the temporal gap (δt). The time constraint (i.e., time gap) in the query determines the time window in which occurrences of the patterns have to appear in the temporally annotated sequences. Since this temporal constraint (δt) determines also the expiration time of a node in our graph representation, as this parameter increases more nodes have to be kept in our data structure. Hence, we observe an increment of the running time for our graph model as shown in Fig. 6a. Regarding the statistical test, we do not observe significant changes in the running time as δt increases (Fig. 6b). From Fig. 6c, we observe that the number of candidate sequences increases as the time gap increases. Nevertheless, this increment is not as evident as the one produced by varying m. We believe that this is due to the high temporal sparsity of the `MIMIC-III` data. In fact, as shown in Table 5, the temporal data sequences may span from 1 day to more than 10 years. Similarly, the number of significant sequences reported by our approach slightly increases as illustrated in Fig. 6d.

Scalability. Here, we evaluate how our solution scales with different sizes of the input dataset. Specifically, we focus our evaluation on the running times of the graph-matching model and statistical test. We artificially construct multiple input datasets from `MIMIC-III`, each containing a different number of sequences. From Fig. 7, we observe a linear dependency between the running time required both by the query answering and statistical test with respect to the size of the data in input. We notice that our overall approach requires roughly 83 ms to retrieve the most significant patient sequences from the largest dataset. This demonstrates the high scalability of our solution.

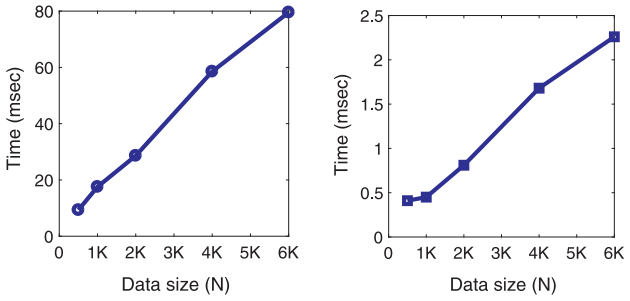
Impact of the p-value threshold (th_p). Finally, we measure the impact of the p-value on the number of reported sequences. As the parameter th_p increases the statistical significance required by the sequences decreases and therefore more sequences are reported by our solution. This behavior is illustrated in Fig. 8.

We conclude this section with some important remarks. (1) Because of the high dimensionality of the data, the time for finding the sequences matching the input query dominates the overall running time. This is particular evident when the number of patterns (m) in the input query is small. (2) Even though the are many potential sequences matching the input query only a small fraction of them are statistically



(a) Time for matching the sequences (b) Time for computing the statistical score (c) Number of sequences matching the patterns (d) Number of significant sequences

Fig. 6. Impact of the temporal gap (δt) in the input query.



(a) Time for matching the sequences (b) Time for determining the statistical significance

Fig. 7. Scalability of our proposed approach. We measure the running time to match the sequence and to perform our statistical test.

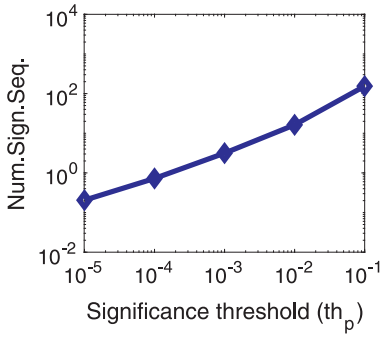


Fig. 8. Impact of the significance threshold on the reported sequences.

significant. Our technique identifies the sequences that are significant in an efficient manner, hence reducing considerably the human intervention cost needed to review the results.

3.2. Retrieving target sequences

In this setting, we evaluate the effectiveness of our solution focusing on an information retrieval setting. We create our input queries by randomly selecting m patterns from target sequences with the goal of retrieving them in the top-k most significant results. To measure the ability of our solution in retrieving the target sequences (i.e., utility), we use the well known notion of True Positive Rate (*TPR*), where higher *TPR* indicates higher utility. The results are reported in Fig. 9 for two different values of k. We observe that, as the number of patterns in the input query increases the *TPR* increases, and when $m = 4$ the *TPR* approaches 1 for $k = 10$. Intuitively, when more patterns are used in the input query a more stringent criterion for the search is used; hence, leading to higher utility. Overall, these results demonstrate that our approach can be effectively used by doctors in retrieving patient data by using just a few known patterns.

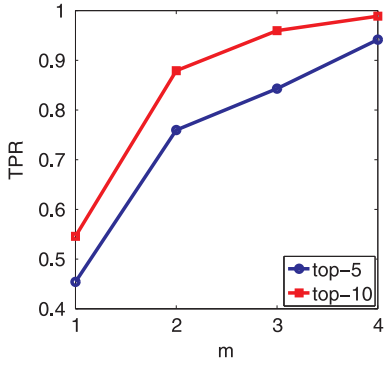


Fig. 9. True Positive Rate (*TPR*) with respect to different number of patterns in the input query.

3.3. Medical tasks

In this section, we conduct several evaluations of our approach within two specific medical applications. We first demonstrate how our solution can be used to discover potential outliers in a mortality study. Then, we show that our method can be applied to retrieve patients that are similar to a new hospitalized patient and how these results can provide useful insights on the medical condition of the new patient.

3.3.1. Mortality study

We consider a mortality study for the patients in `MIMIC-III` and we aim to find patients with surprising behaviors that have been diagnosed with high mortality rate diseases. We start by extracting common patterns that are associated with diagnosis codes with high mortality rate. Then, we use these patterns to formulate input queries with the goal of identifying statistically significant patients exhibiting such patterns. For example, we consider two queries $Q_1 = \{\langle$Ac.Kid.Fail., Ac.Resp.Fail$\rangle$,$\langle$Sep.Shock, Ac.Kid.Fail.-TubrNecr.$\rangle\}$, and $Q_2 = \{\langle$Ac.Kid.Fail., Ac.Kid.Fail.-TubrNecr.$\rangle$,$\langle$Ac.Resp.Fail, Ac.Kid.Fail.$\rangle\}$. We issue these queries independently on the dataset and we observe that their matching sequences have quite different statistical significance scores, as reported in Fig. 10. Specifically, from the ranking results in Q_2, all the patients appear to have nearly the same statistical significance. On the other hand when Q_1 is issued, the top ranked patient has a very high statistical significance with respect to the other patients. By looking at the individual patients corresponding to these results, we observe that: (1) the highest ranking patient in Q_1 is also in Q_2 but with a much lower rank, and (2) from the EHR data, this patient is among few that survived. The results of these two queries are quite interesting and in practice they can provide useful insights to doctors and clinicians that use our solution. First, by carefully selecting the input query the user can detect particular patients as outliers. Second, the distribution of the statistical scores for the resulting sequences provides indications on the discriminatory power of the query. For example, in this case Q_1 better separates interesting patients from the rest compared to Q_2.

3.3.2. Finding similar patients

Our solution can be also applied in the situation where a doctor wants to study potential relationships between previous diagnoses and the current medical condition of an hospitalized patient by finding similar patients. Below, we illustrate this idea with a specific use case focusing on respiration failure. Suppose that a doctor treats a patient with acute respiratory failure who has been previously diagnosed with kidney failure and scoliosis. At the moment the doctor is wondering if kidney failure and scoliosis may be common causes for acute respiratory failure. By using our approach, the doctor can proceed to

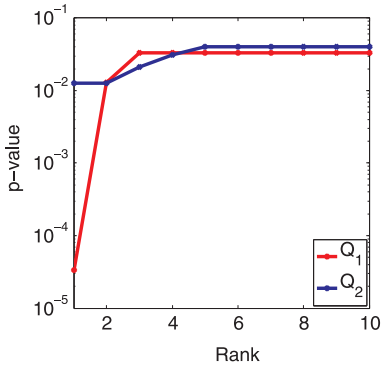


Fig. 10. Ranking results for Q_1 and Q_2.

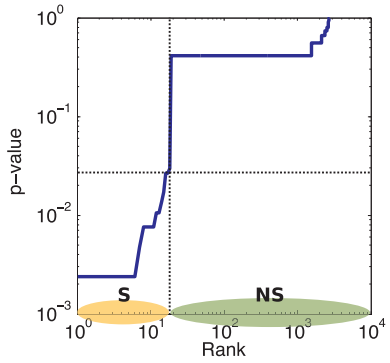


Fig. 11. *p*-value for temporally annotated sequences with respect to a defined query modeling know features (i.e. kidney failure and scoliosis). The returned sequences are divided in two sets: **S** (significant) and **NS** (non-significant) according to a given threshold.

Table 7
Percentage of patients with acute respiratory failure.

Significance	Acute respiratory failure
p-value $<3 \times 10^{-2}$	83%
p-value $\geq 3 \times 10^{-2}$	43%

identify patients with similar codes (i.e., similar patients) and study if acute respiratory failure is a common condition among the significant results. We start by modeling the known features (i.e., kidney failure and scoliosis) as an input query and retrieve significant patients from the dataset. We determine the significance of the matching patients and their *p*-values are reported in Fig. 11. We consider a minimum significance threshold $th_p \approx 0.03$ which defines two sets of sequences **S** and **NS** representing sequences with p-value $< th_p$ (i.e. significant sequences) and those with p-value $\geqslant th_p$ (i.e. non-significant sequences) respectively. Then in these two sets, we observe that 83% of the patients in **S** are diagnosed with acute respiratory failure, while for the patients in **NS** this percentage drops to 43%, as reported in Table 7. Therefore, the significant data instances matching the input query may suggest that kidney failure and scoliosis are common causes for acute respiratory failure. In practice, our solution provides doctors with concrete data instances that could facilitate the understanding of relationships between previous medical events and current conditions in individual patients.

4. Discussion

In this section, we discuss the limitations of our approach and investigate alternative solutions to improve our technique.

- **Selecting the patterns in input**. In our method, we use the a set of patterns, defined by domain experts, to identify the statistically relevant data sequences. These patterns, can either model medical events of interest (e.g., Kawasaki disease example) or hypothesis (e.g., example in Section 3.3.2). Even though there exist known diagnostic guidelines that can be used in modeling these patterns (e.g., evidence of patterns for patients diagnosed with Kawasaki [9]), their definition is still challenging in practice. Therefore, to facilitate the formulation of the patterns in the query, we can consider an iterative approach which uses feedback from the user to suggest the patterns. The user (e.g., clinician/researcher) starts with some known patterns in the input query and by observing the results (i.e., significant sequences) he can pinpoint the clinically useful sequences. Upon the received feedback, our approach will recommend new patterns in the query (e.g., common patterns across sequences pinpointed by the users). In this iterative process, the query patterns are determined to best suit the task required by the user.
- **Matching the patterns in noisy data**. In our approach, we use the notion of temporal match to determine the sequences supporting the patterns in the input query. While this notion allows us to identify the sequences containing the specified patterns with some flexibility on the temporal dimension, it requires all the itemsets in the patterns to occur exactly in the sequences to form a match. However, noise in the medical data may prevent these patterns to match relevant sequences. A possible alternative strategy to overcome this uncertainty in the data consists in the use of statistical models for matching the patterns. On this line of research, recent models have shown to be effective in the sequential data domain [40].
- **Assumptions for the itemset model**. In our null-hypothesis model, we introduce an assumption of global independence among the itemsets. However, medical data may present temporally correlated events which can be used to facilitate diagnosis [41] and link patients longitudinal data [42]. To take into consider such correlation, more sophisticated models can be used in our analysis. For example, Markov models have been shown to be successful in the frequency analysis and modeling of sequential data [19,17,43]. Inspired by these works, we could develop Markov models of fixed order k to compute the probability of k events sequentially occurring over time. To accelerate the computation under such a model, an assumption of non-overlapping between the pattern occurrences could be introduced similarly to the work in [44].

5. Conclusion

In this paper, we proposed a technique that enables the identification of statistically significant sequences from a large collection of temporally annotated data. The information extraction process is efficiently performed using a graph-based representation of the temporal sequences where the significance of each sequence is computed according to a null-hypothesis model. Our experimental evaluations on a real-world dataset showed the effectiveness and efficiency of our approach. We believe that our solution is beneficial for many real medical applications and could facilitate the discovery of clinically useful information. In future research, we aim to develop more sophisticated statistical models for determining the significance of the retrieved sequences.

Acknowledgements

LB is supported in part by the National Institute of Health (NIH) under award number R21LM012060 and R01GM118574. XJ was supported in part by the Patient-Centered Outcomes Research Institute (PCORI) under contract ME-1310-07058, the National Institute of Health (NIH) under award number R01GM114612, R01GM118574, and U01TR00206. The authors also wish to thank the anonymous reviewers for their valuable feedback.

References

[1] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, M. Hsu, Prefixspan: mining sequential patterns by prefix-projected growth, Proceedings of the 17th International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, 2001, pp. 215–224 URL <http://dl.acm.org/citation.cfm?id=645484.656379>.

[2] C. Low-Kam, C. Raïssi, M. Kaytoue, J. Pei, Mining statistically significant sequential patterns, in: 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7–10, 2013, 2013, pp. 488–497. http://dx.doi.org/10.1109/ICDM.2013.124. URL < https://doi.org/10.1109/ICDM.2013.124 > .

[3] R.T. Snodgrass, The TSQL2 Temporal Query Language, Kluwer Academic Publishers, Norwell, MA, USA, 1995.

[4] R. Snodgrass, The temporal query language TQuel, ACM Trans. Database Syst. (TODS) 12 (2) (1987) 247–298.

[5] E. Keogh, S. Chu, D. Hart, M. Pazzani, Segmenting time series: a survey and novel approach, Data Min. Time Ser. Databases 57 (2004) 1–22.

[6] B. Chiu, E. Keogh, S. Lonardi, Probabilistic discovery of time series motifs, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 493–498.
[7] C. Faloutsos, M. Ranganathan, Y. Manolopoulos, Fast Subsequence Matching in Time-series Databases vol. 23, ACM, 1994.
[8] J. Frankovich, C.A. Longhurst, S.M. Sutherland, Evidence-based medicine in the EMR era, N. Engl. J. Med. 365 (19) (2011) 1758–1759.
[9] K. Seaton, A. Kharbanda, Evidence-based management of kawasaki disease in the emergency department, Pediatr. Emerg. Med. Pract. 12 (1) (2015) 1–20.
[10] S. Berchtold, C. Böhm, H.-P. Kriegal, The pyramid-technique: towards breaking the curse of dimensionality, ACM SIGMOD Record, vol. 27, ACM, 1998, pp. 142–153.
[11] C.C. Aggarwal, J. Han, Frequent Pattern Mining, Springer, 2014.
[12] F. Giannotti, M. Nanni, D. Pedreschi, Efficient Mining of Temporally Annotated Sequences, pp. 348–359 (Chapter 31). http://dx.doi.org/10.1137/1.9781611972764.31.
[13] M.J. Zaki, Spade: an efficient algorithm for mining frequent sequences, Mach. Learn. 42 (1–2) (2001) 31–60, http://dx.doi.org/10.1023/A:1007652502315.
[14] Q. Zhao, S.S. Bhowmick, Sequential Pattern Mining: A Survey, ITechnical Report CAIS Nayang Technological University Singapore, 2003, pp. 1–26.
[15] R. Gwadera, M.J. Atallah, W. Szpankowski, Reliable detection of episodes in event sequences, Knowl. Inform. Syst. 7 (4) (2005) 415–437, http://dx.doi.org/10.1007/s10115-004-0174-5.
[16] F. Mörchen, Unsupervised pattern mining from symbolic temporal data, SIGKDD Explor. Newsl. 9 (1) (2007) 41–55, http://dx.doi.org/10.1145/1294301.1294302.
[17] R. Gwadera, F. Crestani, Ranking sequential patterns with respect to significance, Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part I, PAKDD'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 286–299, http://dx.doi.org/10.1007/978-3-642-13657-3_32.
[18] N. Tatti, Ranking episodes using a partition model, Data Min. Knowl. Discov. 29 (5) (2015) 1312–1342.
[19] R. Gwadera, M. Atallah, W. Szpankowski, Markov models for identification of significant episodes, pp. 404–414 (Chapter 36). http://dx.doi.org/10.1137/1.9781611972757.36.
[20] W. Hämäläinen, M. Nykänen, Efficient discovery of statistically significant association rules, Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, ICDM '08, IEEE Computer Society, Washington, DC, USA, 2008, pp. 203–212, http://dx.doi.org/10.1109/ICDM.2008.144.
[21] E. Keogh, Exact indexing of dynamic time warping, in: Proceedings of the 28th International Conference on Very Large Data Bases, VLDB Endowment, 2002, pp. 406–417.
[22] D.S. Hirschberg, Algorithms for the longest common subsequence problem, J. ACM (JACM) 24 (4) (1977) 664–675.
[23] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, Mining sequences with temporal annotations, Proceedings of the 2006 ACM Symposium on Applied Computing, ACM, 2006, pp. 593–597.
[24] J. Ma, S. Perkins, Online novelty detection on temporal sequences, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 613–618.
[25] C.W. Günther, W.M. van der Aalst, Mining activity clusters from low-level event logs, Beta Res. School Oper. Manage. Logist. WP 165 (2006).
[26] Y. Shahar, A framework for knowledge-based temporal abstraction, Artif. Intell. 90 (1–2) (1997) 79–133.
[27] Y. Shahar, S. Miksch, P. Johnson, The asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines, Artif. Intell. Med. 14 (1) (1998) 29–51.
[28] J.F. Roddick, M. Spiliopoulou, A survey of temporal knowledge discovery paradigms and methods, IEEE Trans. Knowl. Data Eng. 14 (4) (2002) 750–767, http://dx.doi.org/10.1109/TKDE.2002.1019212.
[29] J.F. Allen, Maintaining knowledge about temporal intervals, Commun. ACM 26 (11) (1983) 832–843, http://dx.doi.org/10.1145/182.358434.
[30] F. Höppner, Learning temporal rules from state sequences, in: IJCAI Workshop on Learning from Temporal and Spatial Data, vol. 25, 2001.
[31] P.-s. Kam, A.W.-C. Fu, Discovering temporal patterns for interval-based events, International Conference on Data Warehousing and Knowledge Discovery, Springer, 2000, pp. 317–326.
[32] R. Moskovitch, Y. Shahar, Medical temporal-knowledge discovery via temporal abstraction, in: AMIA, 2009.
[33] G. Hripcsak, L. Zhou, S. Parsons, A.K. Das, S.B. Johnson, Modeling electronic discharge summaries as a simple temporal constraint satisfaction problem, J. Am. Med. Infor. Assoc. 12 (1) (2005) 55–63.
[34] A. Gkoulalas-Divanis, G. Loukides, Revisiting sequential pattern hiding to enhance utility, Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '11, ACM, New York, NY, USA, 2011, pp. 1316–1324, http://dx.doi.org/10.1145/2020408.2020605.
[35] A. Gallo, T. De Bie, N. Cristianini, MINI: Mining Informative Non-redundant Itemsets, Springer, Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 438–445, http://dx.doi.org/10.1007/978-3-540-74976-9_44.
[36] A. Gionis, H. Mannila, T. Mielikäinen, P. Tsaparas, Assessing data mining results via swap randomization, ACM Trans. Knowl. Discov. Data 1 (3). http://dx.doi.org/10.1145/1297332.1297338.
[37] S.X. Chen, J.S. Liu, Statistical applications of the poisson-binomial and conditional bernoulli distributions, Stat. Sin. (1997) 875–892.
[38] W. Hoeffding, Probability inequalities for sums of bounded random variables, J. Am. Stat. Assoc. 58 (301) (1963) 13–30, http://dx.doi.org/10.1080/01621459.1963.10500830.
[39] A.E. Johnson, T.J. Pollard, L. Shen, L.-W.H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L.A. Celi, R.G. Mark, Mimic-iii, a freely accessible critical care database, Sci. Data 3 (2016).
[40] E.J. Keogh, P. Smyth, A probabilistic approach to fast pattern matching in time series databases, in: Kdd, vol. 1997, 1997, pp. 24–30.
[41] E. Choi, M.T. Bahadori, A. Schuetz, W.F. Stewart, J. Sun, Doctor ai: Predicting clinical events via recurrent neural networks, in: Machine Learning for Healthcare Conference, 2016, pp. 301–318.
[42] L. Bonomi, X. Jiang, Linking temporal medical records using non-protected health information data, Stat. Methods Med. Res. (2017) 0962280217698005.
[43] S. Laxman, P.S. Sastry, A survey of temporal data mining, Sadhana 31 (2) (2006) 173–198.
[44] S. Laxman, P.S. Sastry, K.P. Unnikrishnan, Discovering frequent episodes and learning hidden markov models: a formal connection, IEEE Trans. Knowl. Data Eng. 17 (11) (2005) 1505–1517, http://dx.doi.org/10.1109/TKDE.2005.181.