Applications of Pattern Discovery Using Sequential Data Mining

Manish Gupta
University of Illinois at Urbana-Champaign, USA

Jiawei Han
University of Illinois at Urbana-Champaign, USA

ABSTRACT

Sequential pattern mining methods have been found to be applicable in a large number of domains. Sequential data is omnipresent. Sequential pattern mining methods have been used to analyze this data and identify patterns. Such patterns have been used to implement efficient systems that can recommend based on previously observed patterns, help in making predictions, improve usability of systems, detect events, and in general help in making strategic product decisions. In this chapter, we discuss the applications of sequential data mining in a variety of domains like healthcare, education, Web usage mining, text mining, bioinformatics, telecommunications, intrusion detection, et cetera. We conclude with a summary of the work.

HEALTHCARE

Patterns in healthcare domain include the common patterns in paths followed by patients in hospitals, patterns observed in symptoms of a particular disease, patterns in daily activity and health data. Works related to these applications are discussed in this sub-section.

Patterns in patient paths: The purpose of the French Diagnosis Related Group's information system is to describe hospital activity by focusing on hospital stays. (Nicolas, Herengt & Albuisson, 2004) propose usage of sequential pattern mining for patient path analysis across multiple healthcare institutions. The objective is to discover, to classify and to visualize frequent patterns among patient path. They view a patient path as a sequence of sets. Each set in the sequence is a hospitalization instance. Each element in a hospitalization can be any symbolic data gathered by the PMSI (medical data source). They used the SLPMiner system (Seno & Karypis, 2002) for mining the patient path database in order to find frequent sequential patterns among the patient path. They tested the model on the 2002 year of PMSI data at the Nancy University Hospital and also propose an interactive tool to perform inter-institutional patient path analysis.

Patterns in dyspepsia symptoms: Consider a domain expert, who is an epidemiologist and is interested in finding relationships between symptoms of dyspepsia within and across time points. This can be done by first mining patterns from symptom data and then using patterns to define association rules. Rules could look like ANOREX2=0 VOMIT2=0 NAUSEA3=0 ANOREX3=0 VOMIT3=0 ⇒ DYSPH2=0 where each symptom is represented as <symptom>N=V (time=N and value=V). ANOREX (anorexia), VOMIT

(vomiting), DYSPH (dysphagia) and NAUSEA (nausea) are the different symptoms. However, a better way of handling this is to define subgroups as a set of symptoms at a single time point. (Lau, Ong, Mahidadia, Hoffmann, Westbrook, & Zrimec, 2003) solve the problem of identifying symptom patterns by implementing a framework for constraint based association rule mining across subgroups. Their framework, Apriori with Subgroup and Constraint (ASC), is built on top of the existing Apriori framework. They have identified four different types of phase-wise constraints for subgroups: constraint across subgroups, constraint on subgroup, constraint on pattern content and constraint on rule. A constraint across subgroups specifies the order of subgroups in which they are to be mined. A constraint on subgroup describes the intra-subgroup criteria of the association rules. It describes a minimum support for subgroups and a set of constraints for each subgroup. A constraint on pattern content outlines the inter-subgroup criteria on association rules. It describes the criteria on the relationships between subgroups. A constraint on rule outlines the composition of an association rule; it describes the attributes that form the antecedents and the consequents, and calculates the confidence of an association rule. It also specifies the minimum support for a rule and prunes away item-sets that do not meet this support at the end of each subgroup-merging step. A typical user constraint can look like [1,2,3][1, a=A1&n<=2][2, a=B1&n<=2][3, v=1][rule, (s1 s2) ⇒s3]. This can be interpreted as: looking at subgroups 1, 2 and 3, from subgroup 1, extract patterns that contain the attribute A1 (a=A1) and contain no more than 2 attributes (n<=2); from subgroup 2, extract patterns that contain the attribute B1 (a=B1) and contain no more than 2 attributes (n<=2); then from subgroup 3, extract patterns with at least one attribute that has a value of 1 (v=1). Attributes from subgroups 1 and 2 form the antecedents in a rule, and attributes from subgroup 3 form the consequents ([rule, (s1 s2) ⇒ s3]). Such constraints are easily incorporated into the Apriori process by pruning away more candidates based on these constraints.

They experimented on a dataset with records of 303 patients treated for dyspepsia. Each record represented a patient, the absence or presence of 10 dyspepsia symptoms at three time points (initial presentation to a general practitioner, 18 months after endoscopy screening, and 8–9 years after endoscopy) and the endoscopic diagnosis for the patient. Each of these symptoms can have one of the following three values: symptom present, symptom absent, missing (unknown). At each of the three time points, a symptom can take any of these three possible values. They show that their approach leads to interesting symptom pattern discovery.

Patterns in daily activity data: There are also works, which investigate techniques for using agent-based smart home technologies to provide at-home automated assistance and health monitoring. These systems first learn patterns from at-home health and activity data. Further, for any new test cases, they identify behaviors that do not conform to normal behavior and report them as predicted anomalous health problems.

EDUCATION

In the education domain, work has been done to extract patterns from source code and student teamwork data.

Patterns in source code: A coding pattern is a frequent sequence of method calls and control statements to implement a particular behavior. Coding patterns include copy-and-pasted code, crosscutting concerns (parts of a program which rely on or must affect many other parts of the system) and implementation idioms. Duplicated code fragments and crosscutting concerns that spread across modules are problematic in software maintenance. (Ishio, Date, Miyake, & Inoue, 2008) propose a sequential pattern mining approach to capture coding patterns in Java programs. They define a set of rules to translate Java source code into a sequence database for pattern mining, and apply PrefixSpan algorithm to the sequence database. They define constraints for mining source code patterns. A constraint for control statements could be: If a pattern includes a LOOP/IF element, the pattern must include its corresponding element generated from the same control statement. They classify sub-patterns into pattern groups. As a case study, they applied their tool to six open-source programs and manually investigated the resultant patterns.

They identify about 17 pattern groups which they classify into 5 categories:

1. A boolean method to insert an additional action: <Boolean method>, <IF>, <action-method>, <END-IF>
2. A boolean method to change the behavior of multiple methods: <Boolean method>, <IF>, <action-method>, <END-IF>
3. A pair of set-up and clean-up: <set-up method>, <misc action>, … , <clean-up method>
4. Exception Handling: Every instance is included in a try-catch statement.
5. Other patterns.

They have made this technique available as a tool: Fung(http://sel.ist.osaka-u.ac.jp/~ishio/fung/)

Patterns in student team-work data: (Kay, Maisonneuve, Yacef, & Zaïane, 2006) describe data mining of student group interaction data to identify significant sequences of activity. The goal is to build tools that can flag interaction sequences indicative of problems, so that they can be used to assist student teams in early recognition of problems. They also want tools that can identify patterns that are markers of success so that these might indicate improvements during the learning process. They obtain their data using TRAC which is an open source tool designed for use in software development projects. Students collaborate by sharing tasks via the TRAC system. These tasks are managed by a “Ticket” system; source code writing tasks are managed by a version control system called “SVN”; students communicate by means of collaborative web page writing called “Wiki”. Data consist of events where each event is represented as Event = {EventType, ResourceId, Author, Time} where: EventType is one of T (for Ticket), S (for SVN), W (for Wiki). One such sequence is generated for each of the group of students.

The original sequence obtained for each group was 285 to 1287 long. These event sequences were then broken down into several “sequences” of events using a per session approach or a per resource approach. In breakdown per session approach, date and the resourceId are omitted and a sequence is of form: (iXj) which captures the number of i consecutive times a medium X was used by j different authors, e.g., <(2T1), (5W3), (2S1),(1W1)>. In breakdown per resource approach, sequence is of form <iXj, t> which captures the number of i different events of type X, the number j of authors, and the number of days over which t the resource was modified, e.g., <10W5, 2>. In a follow-up paper (Perera, Kay, Yacef, & Koprinska, 2007), they have a third approach, breakdown by task where every sequence is of the form (i,X,A) which captures the number of consecutive events (i) occurring on a particular TRAC medium (X), and the role of the author (A).

Patterns observed in group sessions: Better groups had many alternations of SVN and Wiki events, and SVN and Ticket events whereas weaker groups had almost none. The best group also had the highest proportion of author sessions containing many consecutive ticket events (matching their high use of ticketing) and SVN events (suggesting they committed their work to the group repository more often).

A more detailed analysis of these patterns revealed that the best group used the Ticket more than the Wiki, whereas the weakest group displayed the opposite pattern. The data suggested group leaders in good groups were much less involved in technical work, suggesting work was being delegated properly and the leader was leading rather than simply doing all the work. In contrast, the leaders of the poorer groups either seemed to use the Wiki (a less focused medium) more than the tickets, or be involved in too much technical work.

Patterns observed in task sequences: The two best groups had the greatest percentage support for the pattern (1,t,L)(1,t,b), which were most likely tickets initiated by the leader and accepted by another team member. The fact this occurred more often than (1,t,L)(2,t,b), suggests that the better groups were distinguished by tasks being performed on the Wiki or SVN files before the ticket was closed by the second member. Notably, the weakest group had higher support for this latter pattern than the former. The best group was one of the only two to display the patterns (1,t,b)(1,s,b) and (1,s,b)(1,t,b) – the first likely being a ticket being accepted by a team member and then SVN work relating to that task being completed and the second likely being work being done followed by the ticket being closed. The close coupling of task-related SVN and Wiki activity and Ticket events for this group was also shown by relatively high support for the patterns (1,t,b)(1,t,b)(1,t,b), (1,t,b)(1,s,b)(1,t,b) and (1,t,b)(1,w,b)(1,t,b). The poorest group displayed the highest support for the last pattern, but no support for the former, again indicating their lack of SVN use in tasks.

Patterns observed in resource sequences: The best group had very high support for patterns where the leader interacted with group members on tickets, such as (L,1,t)(b,1,t)(L,1,t). The poorest group in contrast lacked these interaction patterns, and had more tickets which were created by the Tracker rather than the Leader, suggestive

of weaker leadership. The best group displayed the highest support for patterns such as (b,3,t) and (b,4,t), suggestive of group members making at least one update on tickets before closing them. In contrast, the weaker groups showed support mainly for the pattern (b,2,t), most likely indicative of group members accepting and closing tickets with no update events in between.

WEB USAGE MINING

The complexity of tasks such as Web site design, Web server design, and of simply navigating through a Web site has been increasing continuously. An important input to these design tasks is the analysis of how a Web site is being used. Usage analysis includes straightforward statistics, such as page access frequency, as well as more sophisticated forms of analysis, such as finding the common traversal paths through a Web site. Web Usage Mining is the application of pattern mining techniques to usage logs of large Web data repositories in order to produce results that can be used in the design tasks mentioned above. However, there are several preprocessing tasks that must be performed prior to applying data mining algorithms to the data collected from server logs.

Transaction identification from web usage data: (Cooley, Mobasher, & Srivastava, 1999) present several data preparation techniques in order to identify unique users and user sessions. Also, a method to divide user sessions into semantically meaningful transactions is defined. Each user session in a user session file can be thought of in two ways; either as a single transaction of many page references, or a set of many transactions each consisting of a single page reference. The goal of transaction identification is to create meaningful clusters of references for each user. Therefore, the task of identifying transactions is one of either dividing a large transaction into multiple smaller ones or merging small transactions into fewer larger ones. This process can be extended into multiple steps of merge or divide in order to create transactions appropriate for a given data mining task. Both types of approaches take a transaction list and possibly some parameters as input, and output a transaction list that has been operated on by the function in the approach in the same format as the input. They consider three different ways of identifying transactions based on: Reference Length (time spent when visiting a page), Maximal Forward Reference (set of pages in the path from the first page in a user session up to the page before a backward reference is made) and Time Window.

By analyzing this information, a Web Usage Mining system can determine temporal relationships among data items such as the following Olympics Web site examples:
– 9.81% of the site visitors accessed the Atlanta home page followed by the Sneakpeek main page.
– 0.42% of the site visitors accessed the Sports main page followed by the Schedules main page.

Patterns for customer acquisition: (Buchner & Mulvenna, 1998) propose an environment that allows the discovery of patterns from trading related web sites, which can be

harnessed for electronic commerce activities, such as personalization, adaptation, customization, profiling, and recommendation.

The two essential parts of customer attraction are the selection of new prospective customers and the acquisition of the selected potential candidates. One marketing strategy to perform this exercise, among others, is to find common characteristics in already existing visitors' information and behavior for the classes of profitable and non-profitable customers. The authors discover these sequences by extending GSP so it can handle duplicates in sequences, which is relevant to discover navigational behavior.

A found sequence looks as the following.
{ecom.infm.ulst.ac.uk/, ecom.infm.ulst.ac.uk/News_Resources.html,
ecom.infm.ulst.ac.uk/Journals.html, ecom.infm.ulst.ac.uk/,
ecom.infm.ulst.ac.uk/search.htm} Support = 3.8%; Confidence = 31.0%
The discovered sequence can then be used to display special offers dynamically to keep a customer interested in the site, after a certain page sequence with a threshold support and/or confidence value has been visited.

Patterns to improve web site design: For the analysis of visitor navigation behavior in web sites integrating multiple information systems (multiple underlying database servers or archives), (Berendt, 2000) proposed the web usage miner (WUM), which discovers navigation patterns subject to advanced statistical and structural constraints. Experiments with a real web site that integrates data from multiple databases, the German SchulWeb (a database of German-language school magazines), demonstrate the appropriateness of WUM in discovering navigation patterns and show how those discoveries can help in assessing and improving the quality of the site design i.e. conformance of the web site's structure to the intuition of each group of visitors accessing the site. The intuition of the visitors is indirectly reflected in their navigation behavior, as represented in their browsing patterns. By comparing the typical patterns with the site usage expected by the site designer, one can examine the quality of the site and give concrete suggestions for its improvement. For instance, repeated refinements of a query may indicate a search environment that is not intuitive for some users. Also, long lists of results may signal that sufficiently selective search options are lacking, or that they are not understood by everyone.

A session is a directed list of page accesses performed by a user during her/his visit in a site. Pages of a session are mapped onto elements of a sequence, whereby each element is a pair comprised of the page and a positive integer. This integer is the occurrence of the page in the session, taking the fact into account that a user may visit the same page more than once during a single session. Further, they also define generalized sequences which are sequences with length constraints on gaps. These constraints are expressed in a mining language MINT.

The patterns that they observe are as follows. Searches reaching a 'school' entry are a dominant sub-pattern. 'State' lists of schools are the most popular lists. Schools are rarely reached in short searches.

Pattern discovery for web personalization: Pattern discovery from usage data can also be used for Web personalization. (Mobasher, Dai, Luo, & Nakagawa, 2002) find that more restrictive patterns, such as contiguous sequential patterns (e.g., frequent navigational paths) are more suitable for predictive tasks, such as Web pre-fetching, which involve predicting which item is accessed next by a user), while less constrained patterns, such as frequent item-sets or general sequential patterns are more effective alternatives in the context of Web personalization and recommender systems.

Web usage preprocessing ultimately results in a set of n page-views, $P = \{p_1, p_2 \ldots p_n\}$, and a set of m user transactions, $T = \{t_1, t_2 \ldots t_m\}$. Each transaction t is defined as an l-length sequence of ordered pairs: $t = <(p_t^1, w(p_t^1)), (p_t^2, w(p_t^2)),\ldots,(p_t^l, w(p_t^l))>$ where $w(p_t^i)$ is the weight associated with page-view p_t^i. Contiguous sequential patterns (CSPs -- patterns in which the items appearing in the sequence must be adjacent with respect to the underlying ordering) are used to capture frequent navigational paths among user trails. General sequential patterns are used to represent more general navigational patterns within the site.

To build a recommendation algorithm using sequential patterns, the authors focus on frequent sequences of size |w| + 1 whose prefix contains an active user session w. The candidate page-views to be recommended are the last items in all such sequences. The recommendation values are based on the confidence of the patterns. A simple trie structure is used to store both the sequential and contiguous sequential patterns discovered during the pattern discovery phase. The recommendation algorithm is extended to generate all k^{th} order recommendations as follows. First, the recommendation engine uses the largest possible active session window as an input for recommendation engine. If the engine cannot generate any recommendations, the size of active session window is iteratively decreased until a recommendation is generated or the window size becomes 0.

The CSP model can do better in terms of precision, but the coverage levels, in general, may be too low when the goal is to generate as many good recommendations as possible. On the other hand, when dealing with applications such as Web pre-fetching in which the primary goal is to predict the user's immediate next actions (rather than providing a broader set of recommendations), the CSP model provides the best choice. This is particularly true in sites with many dynamically generated pages, where often a contiguous navigational path represents a semantically meaningful sequence of user actions each depending on the previous actions.

TEXT MINING

Pattern mining has been used for text databases to discover trends, for text categorization, for document classification and authorship identification. We discuss these works below.

Trends in text databases: (Lent, Agrawal, & Srikant, 1997) describe a system for identifying trends in text documents collected over a period of time. Trends can be used, for example, to discover that a company is shifting interests from one domain to another. Their system mines these trends and also provides a method to visualize them. The unit of text is a word and a phrase is a list of words. Associated with each phrase is a history of the frequency of occurrence of the phrase, obtained by partitioning the documents based upon their timestamps. The frequency of occurrence in a particular time period is the number of documents that contain the phrase. A trend is a specific subsequence of the history of a phrase that satisfies the users' query over the histories. For example, the user may specify a shape query like a spike query to find those phrases whose frequency of occurrence increased and then decreased. In this trend analysis, sequential pattern mining is used for phrase identification.

A transaction ID is assigned to each word of every document treating the words as items in the data mining algorithms. This transformed data is then mined for dominant words and phrases, and the results saved. The user's query is translated into a shape query and this query is then executed over the mined data yielding the desired trends. The results of the mining are a set of phrases that occur frequently in the underlying documents and that match a query supplied by the user. Thus, the system has three major steps: Identifying frequent phrases using sequential patterns mining, generating histories of phrases and finding phrases that satisfy a specified trend.

1-phrase is a list of elements where each element is a phrase. k-phrase is an iterated list of phrases with k levels of nesting. <<(IBM)><(data mining)>> is a 1-phrase, which can mean that IBM and "data mining" should occur in the same paragraph, with "data mining" being contiguous words in the paragraph.

A word in a text field is mapped to an item in a data-sequence or sequential pattern and a phrase to a sequential pattern that has just one item in each element. Each element of a data sequence in the sequential pattern problem has some associated timestamp relative to the other elements in the sequence thereby defining an ordering of the elements of a sequence. Sequential pattern algorithms can now be applied to the transaction ID labeled words to identify simple phrases from the document collection.

User may be interested in phrases that are contained in individual sentences only. Alternatively, the words comprising a phrase may come from sequential sentences so that a phrase spans a paragraph. This generalization can be accommodated by the use of distance constraints that specify a minimum and/or maximum gap between adjacent words of a phrase. For example, the first variation described above would be constrained by specifying a minimum gap of one word and a maximum gap of one sentence. The second variation would have a minimum gap of one sentence and a maximum gap of one paragraph. For this latter example, one could further generalize the notion from a single word from each sentence to a set of words from each sentence by using a sliding transaction time window within sentences. The generalizations made in the GSP algorithm for mining sequential patterns allow a one-to-one mapping of the

minimum gap, maximum gap, and transaction window to the parameters of the algorithm.

Basic mapping of phrases to sequential patterns is extended by providing a hierarchical mapping over sentences, paragraphs, or even sections of a text document. This extended mapping helps in taking advantage of the structure of a document to obtain a richer set of phrases. Where a document has completely separate sections, phrases that span multiple sections can also be mined, thereby discovering a new set of relationships. This enhancement of the GSP algorithm can be implemented by changing the Apriori-like candidate generation algorithm, to consider both phrases and words as individual elements when generating candidate k-phrases. The manner in which these candidates are counted would similarly change.

Patterns for text categorization: (Jaillet, Laurent, & Teisseire, 2006) propose usage of sequential patterns in the SPaC method (Sequential Patterns for Classification) for text categorization. Text categorization is the task of assigning a boolean value to each pair (document, category) where the value is true if the document belongs to the particular category. SPaC method consists of two steps. In the first step, sequential patterns are built from texts. In the second step, sequential patterns are used to classify texts.

The text consists of a set of sentences. Each sentence is associated with a timestamp (its position in the text). Finally the set of words contained in a sentence corresponds to the set of items purchased by the client in the market basket analysis framework. This representation is coupled with a stemming step and a stop-list. Sequential patterns are extracted using a different support applied for each category C_i. The support of a frequent pattern is the number of texts containing the sequence of words. E.g., the sequential pattern < (data) (information) (machine)> means that some texts contain words ‘data’ then ‘information’ then ‘machine’ in three different sentences. Once sequential patterns have been extracted for each category, the goal is to derive a categorizer from the obtained patterns. This is done by computing, for each category, the confidence of each associated sequential pattern. To solve this problem, a rule R is generated in the following way:
R :<s_1 . . . s_p> ⇒ C_i; confidence(R)=(#texts from C_i matching <s_1 . . . s_p>)/(#texts matching <s_1 . . . s_p>).

Rules are sorted depending on their confidence level and the size of the associated sequence. When considering a new text to be classified, a simple categorization policy is applied: the K rules having the best confidence level and being supported are applied. The text is then assigned to the class mainly obtained within the K rules.

Patterns for XML document classification: (Garboni, Masseglia, & Trousse, 2005) present a supervised classification technique for XML documents which is based on structure only. Each XML document is viewed as an ordered labeled tree, represented by its tags only. After a cleaning step, each predefined cluster is characterized in terms of frequent structural subsequences. Then the XML documents are classified based on the mined patterns of each cluster.

Documents are characterized using frequent sub-trees which are common to at least x% (the minimum support) documents of the collection. The system is provided a set of training documents each of which is associated with a category. Frequently occurring tags common to all clusters are removed. In order to transform an XML document to a sequence, the nodes of the XML tree are mapped into identifiers. Then each identifier is associated with its depth in the tree. Finally a depth-first exploration of the tree gives the corresponding sequence. An example sequential pattern looks like <(0 movie), (1 title), (1 url), (1 CountryOfProduction), (2 item), (2 item), (1 filmography), (3 name)>. Once the whole set of sequences (corresponding to the XML documents of a collection) is obtained, a traditional sequential pattern extraction algorithm is used to extract the frequent sequences. Those sequences, once mapped back into trees, will give the frequent sub-trees embedded in the collection.

They tested several measures in order to decide which class each test document belongs to. The two best measures are based on the longest common subsequence. The first one computes the average matching between the test document and the set of sequential patterns and the second measure is a modified measure, which incorporates the actual length of the pattern compared to the maximum length of a sequential pattern in the cluster.

Patterns to identify authors of documents: (Tsuboi, 2002) aims at identifying the authors of mailing list messages using a machine learning technique (Support Vector Machines). In addition, the classifier trained on the mailing list data is applied to identify the author of Web documents in order to investigate performance in authorship identification for more heterogeneous documents. Experimental results show better identification performance when features of not only conventional word N-gram information but also of frequent sequential patterns extracted by a data mining technique (PrefixSpan) are used.

They applied PrefixSpan to extract sequential word patterns from each sentence and used them as author's style markers in documents. The sequential word patterns are sequential patterns where item and sequence correspond to word and sentence, respectively.

Sequential pattern is $<w_1*w_2*...*w_l>$ where w_i is a word and l is the length of pattern. * is any sequence of words including empty sequence. These sequential word patterns were introduced for authorship identification based on the following assumption. Because people usually generate words from the beginning to the end of a sentence, how one orders words in a sentence can be an indicator of author's writing style. As word order in Japanese (they study a Japanese corpus) is relatively free, rigid word segments and non-contiguous word sequences may be a particularly important indicator of the writing style of authors.

While N-grams (consecutive word sequences) fail to account for non-contiguous patterns, sequential pattern mining methods can do so quite naturally.

BIOINFORMATICS

Pattern mining is useful in the bioinformatics domain for predicting rules for organization of certain elements in genes, for protein function prediction, for gene expression analysis, for protein fold recognition and for motif discovery in DNA sequences. We study these applications below.

Pattern mining for bio-sequences: Bio-sequences typically have a small alphabet, a long length, and patterns containing gaps (i.e., “don't care”) of arbitrary size. A long sequence (especially, with a small alphabet) often contains long patterns. Mining frequent patterns in such sequences faces a different type of explosion than in transaction sequences primarily motivated in market-basket analysis. (Wang, Xu, & Yu, 2004) study how this explosion affects the classic sequential pattern mining, and present a scalable two-phase algorithm to deal with this new explosion.

Biosequence patterns have the form of X_1 * ...* X_n spanning over a long region, where each X_i is a short region of consecutive items, called a segment, and * denotes a variable length gap corresponding to a region not conserved in the evolution. The presence of * implies that pattern matching is more permissible and involves the whole range in a sequence. The support of a pattern is the percentage of the sequences in the database that contain the pattern. Given a minimum segment length min_len and a minimum support min_sup, a pattern X_1 *...* X_n is frequent if $|X_i|$>=min_len for 1<=i<=n and the support of the pattern is at least min_sup. The problem of mining sequence patterns is to find all frequent patterns.

The Segment Phase first searches short patterns containing no gaps (X_i), called segments. This phase is efficient. This phase finds all frequent segments and builds an auxiliary structure for answering position queries. GST (generalized suffix tree) is used to find: (1) The frequent segments of length min_len, B_i, called base segments, and the position lists for each B_i, s:p_1, p_2 ... where p_j<p_j+1 and each <s, p_j> is a start position of B_i. (2) All frequent segments of length>min_len. Note that position lists for such frequent segments are not extracted. This information about the base segments and their positions is then stored in an index, Segment to Position Index.

The Pattern Phase searches for long patterns (X_1 *...* X_n) containing multiple segments separated by variable length gaps. This phase grows rapidly one segment at a time, as opposed to one item at a time. This phase is time consuming. The purpose of two phases is to exploit the information obtained from the first phase to speed up the pattern growth and matching and to prune the search space in the second phase.

Two types of pruning techniques are used. Consider a pattern P’, which is a super-pattern of P.
Pattern Generation Pruning: If P*X fails to be a frequent pattern, so does P'*X. So, we can prune P'*X.

Pattern Matching Pruning: If P*X fails to occur before position i in sequence s, so does P'*X. So, we only need to examine the positions after i when matching P'*X against s.

Further to deal with the huge size of the sequences, they introduce compression based querying. In this method, all positions in a non-coding region are compressed into a new item ε that matches no existing item except *. A non-coding region contains no part of a frequent segment. Each original sequence is scanned once, each consecutive region not overlapping with any frequent segment is identified and collapsed into the new item ε. For a long sequence and large min_len and min_sup, a compressed sequence is typically much shorter than the original sequence.

On real life datasets like DNA and protein sequences submitted from 2002/12, 2003/02, they show the superiority of their method compared to PrefixSpan with respect to execution time and the space required.

Patterns in genes for predicting gene organization rules: In eukaryotes, rules regarding organization of cis-regulatory elements are complex. They sometimes govern multiple kinds of elements and positional restrictions on elements. (Terai & Takagi, 2004) propose a method for detecting rules, by which the order of elements is restricted. The order restriction is expressed as element patterns. They extract all the element patterns that occur in promoter regions of at least the specified number of genes. Then, significant patterns are found based on the expression similarity of genes with promoter regions containing each of the extracted patterns. By applying the method to Saccharomyces cerevisiae, they detected significant patterns overlooked by previous methods, thus demonstrating the utility of sequential pattern mining for analysis of eukaryotic gene regulation. Several types of element organization exist, those in which (i) only the order of elements is important, (ii) order and distance both are important and (iii) only the combination of elements is important. In this case, pattern support is the number of genes containing the pattern in their promoter region. Minimum length of the patterns may vary with the species. They use Apriori algorithm to perform mining.

Each element typically has a length of 10–20 base pairs. Therefore, two elements sometimes overlap one another. In this study, any two elements overlapping each other are not considered to be ordered elements, because they use elements defined by computational prediction. Most of these overlapping sites may have no biological meaning; they may simply be false-positive hits during computational prediction of elements. The decision of how to treat such overlapping elements is reflected in the count stage -- if a pattern consisting of element A followed by and overlapping with B should not be considered as <A,B>, we can exclude genes containing such elements when counting the support of <A,B>. This is an interesting tweak in counting support, specific to this problem.

Patterns for predicting protein sequence function: (Wang, Shang, & Li, 2008) present a novel method of protein sequence function prediction based on sequential pattern mining. First, known function sequence dataset is mined to get frequent patterns. Then, a classifier is built using the patterns generated to predict function of protein sequences.

They propose the usage of joined frequent patterns based and joined closed frequent patterns based sequential pattern mining algorithms for mining this data. First, the joined frequent pattern segments are generated. Then, longer frequent patters can be obtained by combining the above segments. They generate closed patterns only. The purpose of producing closed patterns is to use them to construct a classifier for protein function prediction. So using non-redundant patterns can improve the accuracy of classification.

Patterns for analysis of gene expression data: (Icev, 2003) introduces a sequential pattern mining based technique for the analysis of gene expression. Gene expression is the effective production of the protein that a gene encodes. They focus on the characterization of the expression patterns of genes based on their promoter regions. The promoter region of a gene contains short sequences called motifs to which gene regulatory proteins may bind, thereby controlling when and in which cell types the gene is expressed. Their approach addresses two important aspects of gene expression analysis: (1) Binding of proteins at more than one motif is usually required, and several different types of proteins may need to bind several different types of motifs in order to confer transcriptional specificity. (2) Since proteins controlling transcription may need to interact physically, the order and spacing in which motifs occur can affect expression. They use association rules to address the combinatorial aspect. The association rules have the ability to involve multiple motifs and to predict expression in multiple cell types. To address the second aspect, association rules are enhanced with information about the distances among the motifs, or items that are present in the rule. Rules of interest are those whose set of motifs deviates properly, i.e. set of motifs whose pair-wise distances are highly conserved in the promoter regions where these motifs occur.

They define the cvd of a pair of motifs with respect to a collection (or item-set) I of motifs as the ratio between the standard deviation and the mean of the distances between the motifs in those promoter regions that contain all the motifs in I.

Given a dataset of instances D, a minimum support min_sup, a minimum confidence min_conf, and a maximum coefficient of variation of distances (max-cvd), they find all distance-based association rules from D whose support and confidence are >= the min_sup and min_conf thresholds and such that the cvd's of all the pairs of items in the rule are <= the maximum cvd threshold. Their algorithm to mine distance-based association rules from a dataset of instances extends the Apriori algorithm.

In order to obtain distance-based association rules, one could use the Apriori algorithm to mine all association rules whose supports and confidences satisfy the thresholds, and then annotate those rules with the cvd's of all the pair of items present in the rule. Only those rules whose cvd's satisfy the max-cvd threshold are returned. They call this algorithm to mine distance-based association rules, Naïve distance-Apriori.

Distance-based Association Rule Mining (DARM) algorithm first generates all the frequent item-sets that satisfy the max-cvd constraint (cvd-frequent item-sets), and then generates all association rules with the required confidence from those item-sets. Note

that the max-cvd constraint is a non-monotonic property. An item-set that does not satisfy this constraint may have supersets that do. However, they define the following procedure that keeps under consideration only frequent item-sets that deviate properly in an interesting manner.

Let n be the number of promoter regions (instances) in the dataset. Let I be a frequent item-set, and let S be the set of promoter regions that contain I. I is then said to deviate properly if either:
1. I is cvd-frequent. That is, the cvd over S of each pair of motifs in I is <= max-cvd, or
2. For each pair of motifs $P \in I$, there is a subset S' of S with cardinality >= $\lceil min\_sup*n \rceil$ such that the cvd over S' of P is <= max-cvd.

The k-level of item-sets kept by the DARM algorithm is the collection of frequent item-sets of cardinality k that deviate properly. Those item-sets are used to generate the (k+1)-level. Once, all the frequent item-sets that deviate properly have been generated, distance-based association rules are constructed from those item-sets that satisfy the max-cvd constraint. As is the case with the Apriori algorithm, each possible split of such an item-set into two parts, one for the antecedent and one for the consequent of the rule, is considered. If the rule so formed satisfies the min_conf constraint, then the rule is added to the output. These rules are then used for building a classification/predictive model for gene expression.

Patterns for protein fold recognition: Protein data contain discriminative patterns that can be used in many beneficial applications if they are defined correctly. (Exarchos, Papaloukas, Lampros, & Fotiadis, 2008) use sequential pattern mining for sequence-based fold recognition. Protein classification in terms of fold recognition plays an important role in computational protein analysis, since it can contribute to the determination of the function of a protein whose structure is unknown. Fold means 3D structure of a protein. They use cSPADE (Zaki, Sequence mining in categorical domains: incorporating constraints, 2000), for the analysis of protein sequence. Sequential patterns were generated for each category (fold) separately. A $pattern_i$ extracted from $fold_i$, indicates an implication (rule) of the form $pattern_i \Rightarrow fold_i$. A maximum gap constraint is also used.

When classifying an unknown protein to one of the folds, all the extracted sequential patterns from all folds are examined to find which of them are contained in the protein. For a pattern contained in a protein, the score of this protein with respect to this fold is increased by: $score_a^i$=(length of the $pattern_a^i$-k) /(number of patterns in $fold^i$) where 'i' represents a fold, 'a' represents a pattern of a fold. Here, the length is the size of the pattern with gaps. $Pattern_a^i$ is the a^{th} pattern of the i^{th} fold and k is a value employed to assign the minimum score, to the minimal pattern. It should be mentioned that if a pattern is contained in a protein sequence more than once, it receives the same score as if it was contained only once. The scores for each fold are summed and the new protein is assigned to the fold exhibiting the highest sum.

The score of a protein with respect to a fold is calculated based on the number of sequential patterns of this fold contained in the protein. The higher the number of patterns of a fold contained in a protein, the higher the score of the protein for this fold.
A classifier uses the extracted sequential patterns to classify proteins in the appropriate fold category. For training and evaluating the proposed method they used the protein sequences from the Protein Data Bank and the annotation of the SCOP database. The method exhibited an overall accuracy of 25% (random would be 2.8%) in a classification problem with 36 candidate categories. The classification performance reaches up to 56% when the five most probable protein folds are considered.

Patterns for protein family detection: In another work on protein family detection (protein classification), (Ferreira & Azevedo, 2005) use the number and average length of the relevant subsequences shared with each of the protein families, as features to train a Bayes classifier. Priors for the classes are set using the number of patterns and average length of the patterns in the corresponding class.

They identify two types of patterns: Rigid Gap Patterns (only contain gaps with a fixed length) and Flexible Gap Patterns (allow a variable number of gaps between symbols of the sequence). Frequent patterns are mined with the constraint of minimum length. Apart from this, they also support item constraints (restricts set of other symbols that can occur in the pattern), gap constraints (minGap and maxGap), duration or window constraints which defines the maximum distance (window) between the first and the last event of the sequence patterns.

Protein sequences of the same family typically share common subsequences, also called motifs. These subsequences are possibly implied in a structural or biological function of the family and have been preserved through the protein evolution. Thus, if a sequence shares patterns with other sequences it is expected that the sequences are biologically related. Considering the two types of patterns, rigid gap patterns reveal better conserved regions of similarity. On the other hand, flexible gap patterns have a greater probability of occur by chance, having a smaller biological significance. Since the protein alphabet is small, many small patterns that express trivial local similarity may arise. Therefore, longer patterns are expected to express greater confidence in the sequences similarity.

Patterns in DNA sequences: Large collections of genomic information have been accumulated in recent years, and embedded in them is potentially significant knowledge for exploitation in medicine and in the pharmaceutical industry. (Guan, Liu, & Bell, 2004) detect strings in DNA sequences which appear frequently, either within a given sequence (e.g., for a particular patient) or across sequences (e.g., from different patients sharing a particular medical diagnosis). Motifs are strings that occur very frequently. Having discovered such motifs, they show how to mine association rules by an existing rough-sets based technique.

TELECOMMUNICATIONS

Pattern mining can be used in the field of telecommunications for mining of group patterns from mobile user movement data, for customer behavior prediction, for predicting future location of a mobile user for location based services and for mining patterns useful for mobile commerce. We discuss these works briefly in this sub-section.

Patterns in mobile user movement data: (Wang, Lim, & Hwang, 2006) present a new approach to derive groupings of mobile users based on their movement data. User movement data are collected by logging location data emitted from mobile devices tracking users. This data is of the form $D = (D_1, D_2 \ldots D_M)$, where D_i is a time series of tuples (t, (x, y, z)) denoting the x, y and z coordinates of user u_i at time t. A set of consecutive time points $[t_a, t_b]$ is called a valid segment of G (where G is a set of users) if all the pair of users are within dist max_dis for time $[t_a,t_b]$, at least one pair of users has distance greater than max_dis before time t_a, at least one pair of users has distance greater than max_dis after time t_b and t_b-t_a+1 >=min_dur. Given a set of users G, thresholds max_dis and min_dur, these form a group pattern, denoted by P = <G,max_dis,min_dur>, if G has a valid segment. Thus, a group pattern is a group of users that are within a distance threshold from one another for at least a minimum duration.

In a movement database, a group pattern may have multiple valid segments. The combined length of these valid segments is called the weight-count of the pattern. Thus the significance of the pattern is measured by comparing its weight-count with the overall time duration.

Since weight represents the proportion of the time points a group of users stay close together, the larger the weight is, the more significant (or interesting) the group pattern is. Furthermore, if the weight of a group pattern exceeds a threshold min_wei, it is called a valid group pattern, and the corresponding group of users a valid group.

To mine group patterns, they first propose two algorithms, namely AGP (based on Apriori) and VG-growth (based on FP-growth). They show that when both the number of users and logging duration are large, AGP and VG-growth are inefficient for the mining group patterns of size two. Therefore, they propose a framework that summarizes user movement data before group pattern mining. In the second series of experiments, they show that the methods using location summarization reduce the mining overheads for group patterns of size two significantly.

Patterns for customer behavior prediction: Predicting the behavior of customers is challenging, but important for service oriented businesses. Data mining techniques are used to make such predictions, typically using only recent static data. (Eichinger, Nauck, & Klawonn) propose the usage of sequence mining with decision tree analysis for this task. The combined classifier is applied to real customer data and produces promising results.

They use two sequence mining parameters: maxGap, the maximum number of allowed extra events in between a sequence and maxSkip, the maximum number of events at the end of a sequence before the occurrence of the event to be predicted.
They use an Apriori algorithm to detect frequent patterns from a Sequence tree and hash table based data structure. This avoids multiple database scans, which are otherwise necessary after every generation of candidate sequences in Apriori based algorithms.

The frequent sequences are combined with decision tree based classification to predict customer behavior.

Patterns for future location prediction of mobile users: Future location prediction of mobile users can provide location-based services (LBSs) with extended resources, mainly time, to improve system reliability which in turn increases the users' confidence and the demand for LBSs. (Vu, Ryu, & Park, 2009) propose a movement rule-based Location Prediction method (RLP), to guess the user's future location for LBSs. They define moving sequences and frequent patterns in trajectory data. Further, they find out all frequent spatiotemporal movement patterns using an algorithm based on GSP algorithm. The candidate generating mechanism of the technique is based on that of GSP algorithm with an additional temporal join operation and a different method for pruning candidates. In addition, they employ the clustering method to control the dense regions of the patterns. With the frequent movement patterns obtained from the preceding subsection, the movement rules are generated easily.

Patterns for mobile commerce: To better reflect the customer usage patterns in the mobile commerce environment, (Yun & Chen, 2007) propose an innovative mining model, called mining mobile sequential patterns, which takes both the moving patterns and purchase patterns of customers into consideration. How to strike a compromise among the use of various knowledge to solve the mining on mobile sequential patterns, is a challenging issue. They devise three algorithms for determining the frequent sequential patterns from the mobile transaction sequences.

INTRUSION DETECTION

Sequential pattern mining has been used for intrusion detection to study patterns of misuse in network attack data and thereby detect sequential intrusion behaviors and for discovering multistage attack strategies.

Patterns in network attack data: (Wuu, Hung, & Chen, 2007) have implemented an intrusion pattern discovery module in Snort network intrusion detection system which applies data mining technique to extract single intrusion patterns and sequential intrusion patterns from a collection of attack packets, and then converts the patterns to Snort detection rules for on-line intrusion detection. Patterns are extracted both from packet headers and the packet payload. A typical pattern is of the form "A packet with DA port as 139, DgmLen field in header set to 48 and with content as 11 11". Intrusion

behavior detection engine creates an alert when a series of incoming packets match the signatures representing sequential intrusion scenarios.

Patterns for discovering multi-stage attack strategies: In monitoring anomalous network activities, intrusion detection systems tend to generate a large amount of alerts, which greatly increase the workload of post-detection analysis and decision-making. A system to detect the ongoing attacks and predict the upcoming next step of a multistage attack in alert streams by using known attack patterns can effectively solve this problem. The complete, correct and up to date pattern rule of various network attack activities plays an important role in such a system. An approach based on sequential pattern mining technique to discover multistage attack activity patterns is efficient to reduce the labor to construct pattern rules. But in a dynamic network environment where novel attack strategies appear continuously, the novel approach proposed by (Li, Zhang, Li, & Wang, 2007) to use incremental mining algorithm shows better capability to detect recently appeared attack. They remove the unexpected results from mining by computing probabilistic score between successive steps in a multistage attack pattern. They use GSP to discover multistage attack behavior patterns. All the alerts stored in database can be viewed as a global sequence of alerts sorted by ascending DetectTime timestamp. Sequences of alerts describe the behavior and actions of attackers. Multistage attack strategies can be found by analyzing this alert sequence. A sequential pattern is a collection of alerts that occur relatively close to each other in a given order frequently. Once such patterns are known, the rules can be produced for describing or predicting the behavior of the sequence of network attack.

OTHER APPLICATIONS

Apart from the different domains mentioned above, sequential pattern mining has been found useful in a variety of other domains. We briefly mention works in some of such areas in this sub-section. Besides the works mentioned below, there are some applications that may need to classify sequence data, such as based on sequence patterns. An overview on research in sequence classification can be found in (Xing, Pei & Keogh).

Patterns in earth science data: The earth science data consists of time series measurements for various Earth science and climate variables (e.g. soil moisture, temperature, and precipitation), along with additional data from existing ecosystem models (e.g. Net Primary Production). The ecological patterns of interest include associations, clusters, predictive models, and trends. (Potter, Klooster, Torregrosa, Tan, Steinbach, & Kumar) discuss some of the challenges involved in preprocessing and analyzing the data, and also consider techniques for handling some of the spatio-temporal issues. Earth Science data has strong seasonal components that need to be removed prior to pattern analysis, as Earth scientists are primarily interested in patterns that represent deviations from normal seasonal variation such as anomalous climate events (e.g., El Nino) or trends (e.g., global warming). They de-seasonalize the data and then compute variety of spatio-temporal patterns. Rules learned from the patterns

look like (WP-Hi) ⇒ (Solar-Hi) ⇒ (NINO34-Lo) ⇒ (Temp-Hi) ⇒ (NPP-Lo) where WP, Solar etc are different earth science parameters with values Hi (High) or Lo (Low).

Patterns for computer systems management: Predictive algorithms play a crucial role in systems management by alerting the user to potential failures. (Vilalta, Apte, Hellerstein, Ma, & Weiss, 2002) focus on three case studies dealing with the prediction of failures in computer systems: (1) long-term prediction of performance variables (e.g., disk utilization), (2) short-term prediction of abnormal behavior (e.g., threshold violations), and (3) short-term prediction of system events (e.g., router failure). Empirical results show that predictive algorithms based on mining of sequential patterns can be successfully employed in the estimation of performance variables and the prediction of critical events.

Patterns to detect plan failures: (Zaki, Lesh, & Mitsunori, 1999) present an algorithm to extract patterns of events that predict failures in databases of plan executions: PlanMine. Analyzing execution traces is appropriate for planning domains that contain uncertainty, such as incomplete knowledge of the world or actions with probabilistic effects. They extract causes of plan failures and feed the discovered patterns back into the planner. They label each plan as “good" or “bad" depending on whether it achieved its goal or it failed to do so. The goal is to find “interesting" sequences that have a high confidence of predicting plan failure. They use SPADE to mine such patterns.

TRIPS is an integrated system in which a person collaborates with a computer to develop a high quality plan to evacuate people from a small island. During the process of building the plan, the system simulates the plan repeatedly based on a probabilistic model of the domain, including predicted weather patterns and their effect on vehicle performance.

The system returns an estimate of the plan's success. Additionally, TRIPS invokes PlanMine on the execution traces produced by simulation, in order to analyze why the plan failed when it did. The system runs PlanMine on the execution traces of the given plan to pinpoint defects in the plan that most often lead to plan failure. It then applies qualitative reasoning and plan adaptation algorithms to modify the plan to correct the defects detected by PlanMine.

Patterns in automotive warranty data: When a product fails within a certain time period, the warranty is a manufacturer’s assurance to a buyer that the product will be repaired without a cost to the customer. In a service environment where dealers are more likely to replace than to repair, the cost of component failure during the warranty period can easily equal three to ten times the supplier’s unit price. Consequently, companies invest significant amounts of time and resources to monitor, document, and analyze product warranty data. (Buddhakulsomsiri & Zakarian, 2009) present a sequential pattern mining algorithm that allows product and quality engineers to extract hidden knowledge from a large automotive warranty database. The algorithm uses the elementary set concept and database manipulation techniques to search for patterns or relationships among occurrences of warranty claims over time. The sequential patterns are represented in a

form of IF–THEN association rules, where the IF portion of the rule includes quality/warranty problems, represented as labor codes, that occurred in an earlier time, and the THEN portion includes labor codes that occurred at a later time. Once a set of unique sequential patterns is generated, the algorithm applies a set of thresholds to evaluate the significance of the rules and the rules that pass these thresholds are reported in the solution. Significant patterns provide knowledge of one or more product failures that lead to future product fault(s). The effectiveness of the algorithm is illustrated with the warranty data mining application from the automotive industry.

Patterns in alarm data: Increasingly powerful fault management systems are required to ensure robustness and quality of service in today's networks. In this context, event correlation is of prime importance to extract meaningful information from the wealth of alarm data generated by the network. Existing sequential data mining techniques address the task of identifying possible correlations in sequences of alarms. The output sequence sets, however, may contain sequences which are not plausible from the point of view of network topology constraints. (Devitt, Duffin, & Moloney, 2005) presents the Topographical Proximity (TP) approach which exploits topographical information embedded in alarm data in order to address this lack of plausibility in mined sequences. Their approach is based on an Apriori approach and introduces a novel criterion for sequence selection which evaluates sequence plausibility and coherence in the context of network topology. Connections are inferred at run-time between pairs of alarm generating nodes in the data and a Topographical Proximity (TP) measure is assigned based on the strength of the inferred connection. The TP measure is used to reject or promote candidate sequences on the basis of their plausibility, i.e. the strength of their connection, thereby reducing the candidate sequence set and optimizing the space and time constraints of the data mining process.

Patterns for personalized recommendation system: (Romero, Ventura, Delgado, & Bra, 2007) describe a personalized recommender system that uses web mining techniques for recommending a student which (next) links to visit within an adaptable educational hypermedia system. They present a specific mining tool and a recommender engine that helps the teacher to carry out the whole web mining process. The overall process of Web personalization based on Web usage mining generally consists of three phases: data preparation, pattern discovery and recommendation. The first two phases are performed off-line and the last phase on-line. To make recommendations to a student, the system first, classifies the new students in one of the groups of students (clusters). Then, it only uses the sequential patterns of the corresponding group to personalize the recommendations based on other similar students and his current navigation. Grouping of students is done using k-means. They use GSP to get frequent sequences for each of the clusters. They mine rules of the form readme⇒install, welcome⇒install which are intuitively quite common patterns for websites.

Patterns in atmospheric aerosol data: EDAM (Exploratory Data Analysis and Management) is a joint project between researchers in Atmospheric Chemistry and Computer Science at Carleton College and the University of Wisconsin-Madison that

aims to develop data mining techniques for advancing the state of the art in analyzing atmospheric aerosol datasets.

The traditional approach for particle measurement, which is the collection of bulk samples of particulates on filters, is not adequate for studying particle dynamics and real-time correlations. This has led to the development of a new generation of real-time instruments that provide continuous or semi-continuous streams of data about certain aerosol properties. However, these instruments have added a significant level of complexity to atmospheric aerosol data, and dramatically increased the amounts of data to be collected, managed, and analyzed. (Ramakrishnan, et al., 2005) experiment with a dataset consisting of samples from aerosol time-of-flight mass spectrometer (ATOFMS).

A mass spectrum is a plot of signal intensity (often normalized to the largest peak in the spectrum) versus the mass-to-charge (m/z) ratio of the detected ions. Thus, the presence of a peak indicates the presence of one or more ions containing the m/z value indicated, within the ion cloud generated upon the interaction between the particle and the laser beam. In many cases, the ATOFMS generates elemental ions. Thus, the presence of certain peaks indicates that elements such as Na+ (m/z = +23) or Fe+ (m/z = +56) or O- (m/z = -16) ions are present. In other cases, cluster ions are formed, and thus the m/z observed represents that of a sum of the atomic weights of various elements.

For many kinds of analysis, what is significant in each particle's mass spectrum is the composition of the particle, i.e., the ions identified by the peak labels (and, ideally, their proportions in the particle, and our confidence in having correctly identified them). While this representation is less detailed than the labeled spectrum itself, it allows us to think of the ATOFMS data stream as a time-series of observations, one per observed particle, where each observation is a set of ions (possibly labeled with some additional details). This is precisely the market-basket abstraction used in e-commerce: a time-series of customer transactions, each recording the items purchased by a customer on a single visit to a store. This analogy opens the door to applying a wide range of association rule and sequential pattern algorithms to the analysis of mass spectrometry data. Once these patterns are mined, they can be used to extrapolate to periods where filter-based samples were not collected.

Patterns in individuals' time diaries: Identifying patterns of activities within individuals' time diaries and studying similarities and deviations between individuals in a population is of interest in time use research. So far, activity patterns in a population have mostly been studied either by visual inspection, searching for occurrences of specific activity sequences and studying their distribution in the population, or statistical methods such as time series analysis in order to analyze daily behavior. (Vrotsou, Ellegård, & Cooper) describe a new approach for extracting activity patterns from time diaries that uses sequential data mining techniques. They have implemented an algorithm that searches the time diaries and automatically extracts all activity patterns meeting user-defined criteria of what constitutes a valid pattern of interest. Amongst the many criteria which can be applied are: a time window containing the pattern, and minimum and maximum

number of people that perform the pattern. The extracted activity patterns can then be interactively filtered, visualized and analyzed to reveal interesting insights using the VISUAL-TimePAcTS application. To demonstrate the value of this approach they consider and discuss sequential activity patterns at a population level, from a single day perspective, with focus on the activity “paid work” and some activities surrounding it.

An activity pattern in this paper is defined as a sequence of activities performed by an individual which by itself or together with other activities, aims at accomplishing a more general goal/project. When analyzing a single day of diary data, activity patterns identified in a single individual (referred to as an individual activity pattern) are unlikely to be significant but those found amongst a group or population (a collective activity pattern) are of greater interest. Seven categories of activities that they consider are: care for oneself, care for others, household care, recreation/reflection, travel, prepare/procure food, work/school. {“cook dinner”; “eat dinner”; “wash dishes”} is a typical pattern. They also incorporate a variety of constraints like min and max pattern duration, min and max gap between activities, min and max number of occurrences of the pattern and min and max number of people (or a percentage of the population) that should be performing the pattern. The sequential mining algorithm that they have used for the activity pattern extraction is an “AprioriAll” algorithm which is adapted to the time diary data.

Two stage classification using patterns: (Exarchos, Tsipouras, Papaloukas, & Fotiadis, 2008) present a methodology for sequence classification, which employs sequential pattern mining and optimization, in a two-stage process. In the first stage, a sequence classification model is defined, based on a set of sequential patterns and two sets of weights are introduced, one for the patterns and one for classes. In the second stage, an optimization technique is employed to estimate the weight values and achieve optimal classification accuracy. Extensive evaluation of the methodology is carried out, by varying the number of sequences, the number of patterns and the number of classes and it is compared with similar sequence classification approaches.

CONCLUSION

We presented selected applications of the sequential pattern mining methods in the fields of healthcare, education, web usage mining, text mining, bioinformatics, telecommunications, intrusion detection, etc. We envision that the power of sequential mining methods has not yet been fully exploited. We hope to see many more strong applications of these methods in a variety of domains in the years to come.

REFERENCES

Berendt, B. A. (2000). Analysis of navigation behaviour in web sites integrating multiple information systems. *The VLDB Journal, 9*(1), 56-75.
Buchner, A. G., & Mulvenna, M. D. (1998). Discovering Internet marketing intelligence through online analytical web usage mining. *SIGMOD Record, 27*(4), 54-61.

Buddhakulsomsiri, J., & Zakarian, A. (2009). Sequential pattern mining algorithm for automotive warranty data. *Journal of Computers and Industrial Engineering, 57*(1), 137-147.

Chen, Y.-L., & Huang, T. C.-K. (2008). A novel knowledge discovering model for mining fuzzy multi-level sequential patterns in sequence databases. *Data and Knowledge Engienering, 66*(3), 349-367.

Cooley, R., Mobasher, B., & Srivastava, J. (1999). Data preparation for mining World Wide Web browsing patterns. *Knowledge and Information Systems, 1*(1), 5-32.

Devitt, A., Duffin, J., & Moloney, R. (2005). Topographical proximity for mining network alarm data. *MineNet '05: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data* (pp. 179-184). Philadelphia, PA: ACM.

Eichinger, F., Nauck, D. D., & Klawonn, F. (n.d.). *Sequence mining for customer behaviour predictions in telecommunications*.

Exarchos, T. P., Papaloukas, C., Lampros, C., & Fotiadis, D. I. (2008). Mining sequential patterns for protein fold recognition. *Journal of Biomedical Informatics, 41*(1), 165-179.

Exarchos, T. P., Tsipouras, M. G., Papaloukas, C., & Fotiadis, D. I. (2008). A two-stage methodology for sequence classification based on sequential pattern mining and optimization. *Data Knowl. Eng., 66*(3), 467-487.

Ferreira, P. G., & Azevedo, P. J. (2005). Protein sequence classification through relevant sequence mining and bayes classifiers. *Proc. 12th Portuguese Conference on Artificial Intelligence (EPIA)* (pp. 236-247). Springer-Verlag.

Garboni, C., Masseglia, F., & Trousse, B. (2005). *Sequential pattern mining for structure-based XML document classification*. Workshop of the INitiative for the Evaluation of XML Retrieval.

Guan, J. W., Liu, D., & Bell, D. A. (2004). Discovering motifs in DNA sequences. *Fundam. Inform., 59*(2-3), 119-134.

Icev, A. (2003). *Distance-enhanced association rules for gene expression.* BIOKDD'03, in conjunction with ACM SIGKDD.

Ishio, T., Date, H., Miyake, T., & Inoue, K. (2008). Mining coding patterns to detect crosscutting concerns in Java programs. *WCRE '08: Proceedings of the 2008 15th Working Conference on Reverse Engineering* (pp. 123-132). Washington, DC: IEEE Computer Society.

Jaillet, S., Laurent, A., & Teisseire, M. (2006). Sequential patterns for text categorization. *Intell. Data Anal., 10*(3), 199-214.

Kay, J., Maisonneuve, N., Yacef, K., & Zaïane, O. (2006). *Mining patterns of events in students' teamwork data*. In Educational Data Mining Workshop, held in conjunction with Intelligent Tutoring Systems (ITS), (pp. 45-52).

Kum, H.-C., Chang, J. H., & Wang, W. (2007). Benchmarking the effectiveness of sequential pattern mining methods. *Data Knowl. Eng., 60*(1), 30-50.

Kum, H.-C., Chang, J. H., & Wang, W. (2006). Sequential Pattern Mining in Multi-Databases via Multiple Alignment. *Data Min. Knowl. Discov., 12*(2-3), 151-180.

Kuo, R. J., Chao, C. M., & Liu, C. Y. (2009). Integration of K-means algorithm and AprioriSome algorithm for fuzzy sequential pattern mining. *Appl. Soft Comput., 9*(1), 85-93.

Lau, A., Ong, S. S., Mahidadia, A., Hoffmann, A., Westbrook, J., & Zrimec, T. (2003). Mining patterns of dyspepsia symptoms across time points using constraint association rules. *PAKDD'03: Proceedings of the 7th Pacific-Asia conference on Advances in knowledge discovery and data mining* (pp. 124-135). Seoul, Korea: Springer-Verlag.

Laur, P.-A., Symphor, J.-E., Nock, R., & Poncelet, P. (2007). Statistical supports for mining sequential patterns and improving the incremental update process on data streams. *Intell. Data Anal., 11*(1), 29-47.

Lent, B., Agrawal, R., & Srikant, R. (1997). Discovering trends in text databases. *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining, KDD* (pp. 227-230). AAAI Press.

Li, Z., Zhang, A., Li, D., & Wang, L. (2007). Discovering novel multistage attack strategies. *ADMA '07: Proceedings of the 3rd international conference on Advanced Data Mining and Applications* (pp. 45-56). Harbin, China: Springer-Verlag.

Lin, N. P., Chen, H.-J., Hao, W.-H., Chueh, H.-E., & Chang, C.-I. (2008). Mining strong positive and negative sequential patterns. *W. Trans. on Comp., 7*(3), 119-124.

Mannila, H., Toivonen, H., & Verkamo, I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery, 1*(3), 259-289.

Masseglia, F., Poncelet, P., & Teisseire, M. (2009). Efficient mining of sequential patterns with time constraints: Reducing the combinations. *Expert Syst. Appl., 36*(2), 2677-2690.

Masseglia, F., Poncelet, P., & Teisseire, M. (2003). Incremental mining of sequential patterns in large databases. *Data Knowl. Eng., 46*(1), 97–121.

Mendes, L. F., Ding, B., & Han, J. (2008). Stream sequential pattern mining with precise error bounds. *Proc. 2008 Int. Conf. on Data Mining (ICDM'08),* Italy, Dec. 2008.

Mobasher, B., Dai, H., Luo, T., & Nakagawa, M. (2002). Using sequential and non-sequential patterns in predictive Web usage mining tasks. *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining* (pp. 669-672). Washington, DC: IEEE Computer Society.

Nicolas, J. A., Herengt, G., & Albuisson, E. (2004). Sequential pattern mining and classification of patient path. *MEDINFO 2004: Proceedings Of The 11th World Congress On Medical Informatics.*

Parthasarathy, S., Zaki, M., Ogihara, M., & Dwarkadas, S. (1999). Incremental and interactive sequence mining. In *Proc. of the 8th Int. Conf. on Information and Knowledge Management (CIKM'99).*

Perera, D., Kay, J., Yacef, K., & Koprinska, I. (2007). Mining learners' traces from an online collaboration tool. *Proceedings of Educational Data Mining workshop*, (pp. 60-69). Marina del Rey, CA, USA.

Pinto, H., Han, J., Pei, J., Wang, K., Chen, Q., & Dayal, U. (2001). Multi-dimensional sequential pattern mining. *CIKM '01: Proceedings of the Tenth International Conference on Information and Knowledge Management* (pp. 81-88). New York, NY: ACM.

Potter, C., Klooster, S., Torregrosa, A., Tan, P.-N., Steinbach, M., & Kumar, V. (n.d.). *Finding spatio-temporal patterns in earth science data*.

Ramakrishnan, R., Schauer, J. J., Chen, L., Huang, Z., Shafer, M. M., Gross, D. S., et al. (2005). The EDAM project: Mining atmospheric aerosol datasets: Research articles. *Int. J. Intell. Syst., 20*(7), 759-787.

Romero, C., Ventura, S., Delgado, J. A., & Bra, P. D. (2007). *Personalized links recommendation based on data mining un adaptive educational hypermedia systems*.

Creating New Learning Experiences on a Global Scale. Second European Conference on Technology Enhanced Learning, EC-TEL 2007 (pp. 293-305). Crete, Greece: Springer.
Seno, M., & Karypis, G. (2002). SLPMiner: An algorithm for finding frequent sequential patterns using length-decreasing support constraint. In *Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM)*, (pp. 418-425).
Srikant, R., & Agrawal, R. (1996). *Advances in Database Technology EDBT '96*., (pp. 3-17).
Terai, G., & Takagi, T. (2004). Predicting rules on organization of cis-regulatory elements, taking the order of elements into account. *Bioinformatics, 20*(7), 1119-1128.
Tsuboi, Y. (2002). *Authorship identification for heterogeneous documents*.
Vilalta, R., Apte, C. V., Hellerstein, J. L., Ma, S., & Weiss, S. M. (2002). Predictive algorithms in the management of computer systems. *IBM Syst. J., 41*(3), 461-474.
Vrotsou, K., Ellegård, K., & Cooper, M. (n.d.). *Exploring time diaries using semi-automated activity pattern extraction*.
Vu, T. H., Ryu, K. H., & Park, N. (2009). A method for predicting future location of mobile user for location-based services system. *Comput. Ind. Eng., 57*(1), 91-105.
Wang, J. L., Chirn, G., Marr, T., Shapiro, B., Shasha, D., & Zhang, K. (1994). Combinatorial pattern discovery for scientific data: Some preliminary results. *Proc. ACM SIGMOD Int'l Conf. Management of Data*, (pp. 115-125).
Wang, K., Xu, Y., & Yu, J. X. (2004). Scalable sequential pattern mining for biological sequences. *CIKM '04: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management* (pp. 178-187). Washington, DC: ACM.
Wang, M., Shang, X.-Q., & Li, Z.-H. (2008). Sequential pattern mining for protein function prediction. *ADMA '08: Proceedings of 4th International Conference on Adv Data Mining and Applications* (pp. 652-658). Chengdu, China: Springer-Verlag.
Wang, Y., Lim, E.-P., & Hwang, S.-Y. (2006). Efficient mining of group patterns from user movement data. *Data Knowl. Eng., 57*(3), 240-282.
Wong, P. C., Cowley, W., Foote, H., Jurrus, E., & Thomas, J. (2000). Visualizing sequential patterns for text mining. *Proc. IEEE Information Visualization, 2000* (pp. 105-114). Society Press.
Wuu, L.-C., Hung, C.-H., & Chen, S.-F. (2007). Building intrusion pattern miner for Snort network intrusion detection system. *Journal of Systems and Software, 80*(10), 1699-1715.
Xing, Z., Pei, J., & Keogh, E. (2010). A brief survey on sequence classification. *SIGKDD Explorations Newsletter, 12*(1), 40-48.
Yun, C. H., & Chen, M. S. (2007). Mining mobile sequential patterns in a mobile commerce environment. *IEEE Transactions on Systems, Man, Cybernetics,* 278-295.
Yun, U. (2008). A new framework for detecting weighted sequential patterns in large sequence databases. *Know.-Based Syst., 21*(2), 110-122.
Zaki, M. J. (2001). SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning*, *42*(1-2), 31-60.
Zaki, M. J., Lesh, N., & Mitsunori, O. (1999). PlanMine: Predicting plan failures using sequence mining. *Artificial Intelligence Review, 14*(6), 421-446.

ADDITIONAL READING

Adamo, J.-M. (2001). *Data Mining for Association Rules and Sequential Patterns: Sequential and Parallel Algorithms.* Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Alves, R., Rodriguez-Baena, D. S., Aguilar-Ruiz, & S., J. (2009). Gene association analysis: a survey of frequent pattern mining from gene expression data. *Briefings in Bioinformatics* , 210-224.

Han, J., & Kamber, M. (2006). *Data Mining: Concepts and Techniques, Second edition.* Morgan Kaufmann Publishers.

Li, T.-R., Xu, Y., Ruan, D., & Pan, W.-m. Sequential pattern mining. In R. Da, G. Chen, E. E. Kerre, & G. Wets, *Intelligent data mining: techniques and applications* (pp. 103-122). Springer.

Lu, J., Adjei, O., Chen, W., Hussain, F., & Enachescu, C. (n.d.). Sequential Patterns Mining.

Srinivasa, R. N. (2005). Data mining in e-commerce: A survey. *Sadhana* , 275-289.

Teisseire, M., Poncelet, P., Scientifique, P., Besse, G., Masseglia, F., Masseglia, F., et al. (2005). Sequential pattern mining: A survey on issues and approaches. *Encyclopedia of Data Warehousing and Mining, nformation Science Publishing* (pp. 3-29). Oxford University Press.

Yang, L. (2003). Visualizing frequent itemsets, association rules, and sequential patterns in parallel coordinates. *ICCSA'03: Proceedings of the 2003 international conference on Computational science and its applications* (pp. 21-30). Montreal, Canada: Springer-Verlag.

Zhao, Q., & Bhowmick, S. S. (2003). Sequential Pattern Matching: A Survey.