Temporal Association Rules: A Survey

Written Comprehensive Examination
Inés Fernando Vega López
Department of Computer Science
University of Arizona, Tucson, AZ 85721
ifvega@cs.arizona.edu

Abstract

Data mining has been an area of research interest in recent years. In particular, the problem of discovering association rules has been widely studied. Here, we presented a small survey of different research approaches on the mining of association rules. We pay particular attention to mining association rules in temporal and sequential data sets. In addition we compare the problem of similarity search to the problem of mining of association rules. While the problem of mining association rules focus on discovering all interesting patterns, similarity search deals with the problem of finding the objects that follow a specific pattern. These two approaches complement each other as after a set of interesting rules have been discovered, we can apply similarity search to look for the objects that satisfy a particular rule.

1 Introduction

The discovery of interesting information from large databases has been a topic of research for the last decade. Traditional examples involve *market basket data*, in most cases, obtained from retailer stores. Given the massive amount of data that can be obtained from these applications (on Friday 11/29/02 alone, one of these chain of stores accounted sales for over 1 billion dollars), it is very difficult analyze data *by hand*. Data mining is an important step towards extracting information from massive databases. It deals with the discovery o frequent patterns or associations in data. The term *association rule* has been used to describe a specific form of patterns.

Given a database D, of transactions, where each transaction $T \in D$ is a set of items, an association rule $X \Rightarrow Y$ is a statement of the form ”for $s\%$ of the transactions, if the value of an item set X is a particular value, then the value of item set Y tends to have another particular value”. The rule confidence can be understood as the conditional probability $p(Y \subseteq T | X \subseteq T)$. Association rules aim at identifying the patterns of co-occurrences of the attributes in the database [6, 11]. The problem of discovering association rules was first explored in [1] on a supermarket basket data. In this pioneering work, mining of association rules was decomposed into two subproblems; the first one is to discover all frequent patterns, and the second is to generate the association rules from the sets representing those frequent itemsets. The second problem is straightforward and can be done efficiently in a reasonable time. However, the first subproblem is tedious and computationally expensive for very large databases [11]. This complexity increases if we consider continuous domains (instead of items and itemsets), or temporal data.

The rest of the paper is organized as follows: in section 2, we define the classical problem of mining for association rules. In section 3, we include a small survey of papers addressing the problem of mining association rules from temporal databases. In section 4, we focus on papers mining frequent patterns from time series. In section 5, we establish a connection between the problem of mining for association rules and similarity search. Finally, in section 6, we provide concluding remarks and some research questions.

2 Traditional Association Rules

The problem of discovering association rules is traditionally defined as follows: given fixed thresholds for the permisible minimum support and confidence, find all association rules that hold with more than the given support and confidence. Formally, an association rule is defined as [6, 8, 11, 12]:

Definition 1 *Let $I = \{x_1, \ldots, x_n\}$ be a set of distinct literals called items. A set $X \subseteq I$ with $k = |X|$ is called a k-itemset. Let a database D be a multi-set of subsets of I. Each $T \in D$ is called a transaction. We say that a transaction $T \in D$ supports an itemset $X \subseteq I$ if $X \subseteq T$ holds. An association rule is an expression $X \Rightarrow Y$, where X, Y are itemsets and $X \cap Y = \emptyset$. The fraction of transactions T supporting an itemset X with respect to the database D is called the suport of X, $supp(X) = |\{T \in D | X \subseteq T\}|/|D|$. The support of a rule $X \Rightarrow Y$ is defined as $supp(X \cup Y)$. The confindence of this rule is defined as $conf(X \Rightarrow Y) = supp(X \cup Y)/supp(X)$.*

The seminal work by Agrawal *et al.* [1] decomposed the problem of mining for association rules into two parts:

1. Generate all combinations of items that have a support above the user-specified threshold. These combinations of items are also called large itemsets.
2. Use the large itemsets to generate association rules.

The key problem here is to effficiently solve the first subproblem. Given a data set with n distinct items, there are potentially $2^n - 1$ large itemsets. Therefore effective algorithms need to prune most of the itemsets as early as possible in order to keep computational expenses down to a minimum. The most common prunning technique was proposed by Agrawal and it is based on the *a priori* princliple, which indicates that if if one subset of an itemset is not frequent, then the itemset cannot be frequent. For a complete reference of the different algorithms for finding large itemsets please refer to [6].

3 Temporal Association Rules

A temporal association rule is an association rule that holds during specific time intervals. A temporal association rule can be considered as a pair $<AR, TF>$, where AR is an statement of the form $X \Rightarrow Y$ (i.e. an association rule), and TF is a temporal feature (e.g. a valid period, a periodicity or a specific calendar), for which AR is valid. Three interesting tasks have been identified for mining temporal association rules. The discovery of valid time periods during which the association rules hold, the discovery of possible periodicities that association rules have, and the discovery of association rules with temporal features [3, 7].

Tansel *et al.* [11] address the problem of mining temporal association rules and propose an approach aimed to detect changes in the association rules along the time dimension. To find sequences of association rules, they decompose the problem of observing changes in association rules into three subproblems. In a pre-processing step, the time line is partitioned into consecutive disjoint intervals. At the same time, subsets of data valid at each interval are generated. In a second step, they find all the association rules for each time interval. In doing so, they can use any algorithm for the traditional association rule mining problem. The last step consist in the presentation of rules and their fluctuations to the users [11].

Li *et al.* [7] address this problem from a slightly different perspective. Instead of finding sequences of association rules along the time line, they look for patterns in the association rules. In particular they look at calendar patterns. A calendar pattern is defined in the context of a calendar schema. A calendar schema is a relational schema $R = (Gn : D_n, G_{n-1} : D_{n-1}, \ldots, G1 : D_1)$, where each attribute G_i is a granularity name like *year, month, and week* and each domain D_i is a finite subset of the positive integers.

Given a calendar schema R, a simple calendar-based pattern on R is a tuple of the form $< d_n, \ldots, d_1 >$, where each d_i is in D_i or is the wildcard symbol *. They call a calendar pattern with k wildcard symbols a k-star calendar pattern. They call a calendar pattern with no wildcard symbols a basic time interval.

Given a temporal database and calendar schema, they want to discover all interesting association rules with all their calendar. In this context, mining temporal association rules is a two step process quite similar to the mining of traditional association rules:

1. Find all large itemsets for all star calendar patterns on the given calendar schema
2. Generate temporal association rules using the large itemsets and their calendar patterns

To find large itemsets, Li's algorithm starts by processing all calendar patterns without wildcards in them (i.e basic intervals). During the processing of each basic time interval, e_0, the set of large k-itemsets $L_k(e_0)$ is first computed. $L_k(e_0)$ is then used to update the large k-itemsets for all the k-star calendar patterns that cover the basic time interval e_0.

Rainsford *et al.* [9] consider the temporal relationships between objects for developing temporal association rules. They extend traditional association rules to include temporal predicates. These predicates involve a temporal relationship between the items in the association rule. They use a taxonomy with 13 different temporal relationship classes, namely: *before, meets, overlaps, finished by, contains, starts, equals, started by, during, finishes, overlapped by, met by, and after*. In their approach, they first find all association rules using the traditional approach and ignoring any temporal information. When all the association rules have been identified, they find all the possible pairs of items on each association rule. In the third step, the database is scanned to identify the kind of temporal relationship between every pair of items. Each tuple is checked to see if it supports a given rule. The fourth step is the derivation of temporal association rules based upon the initial set of association rules and the aggregation of temporal relationships found to exist during the third step. Since the final set of temporal association rules can be difficult to understand by a human, Rainsford *et al.* [10] developed an schema for visualizing and effectively presenting temporal association rules to human users.

4 Mining Time Sequences

The problem of mining sequential patterns is to find the maximal sequences among all sequences that have certain user-specified minimum support. Each such maximal sequence represents a sequential pattern [2]. Or, using more a more general description, the sequence mining task is to discover a set of attributes, shared across time among a large number of objects in a given database [13]. An example could be that "70% of the people who buy Octavio Paz's *El Laberinto de la Soledad (The Labyrinth of Solitude)* also buy *Águila o Sol? (Eagle or Sun?)* within a month".

Wang *et al.* [12] generalize the association rule discovery problem to encompass numerical attributes and their temporal evolutions. They view the database as a sequence of snapshots of objects and their values taken at some frequency. In their approach, the main idea is to discretize both the value domain and the time dimension so traditional mining of association rules can be used. Traditional association rules assume a discrete domain of attributes. In this case, the authors consider that the domain of a numerical attribute might contain a large number of distinct values or could even be continuous. Thus, the number of possible rules can be very large to infinity (since the number of different itemsets for an attribute with m values is $2^m - 1$). To minimize the space of potential association rules, each attribute domain is quantized into a set of equal-length intervals. Similarly, the time line is implicitly partitioned into disjoint consecutive intervals as defined by the different snapshots of the database (i.e a time interval is defined by two consecutive snapshots and information is valid for that interval).

Once the data space is partitioned in this way, the problem of mining for association rules is reduced to:

1. Find all cubes in the space satisfying a user-defined density value.
2. Using this cubes, find all the valid rule sets using traditional algorithms.

Das *et al.* [4] use an approach similar to that of Wang *et al.* as they chose to discretize the data space for mining association rules on time series data. Their rule discovery method aims at finding local relationships from the series. They extract subsequences using a sliding window of size w. Then, create k clusters using all the subsequences and a distance function between time series. For each cluster C_i, they introduce an alphabet symbol a_i. The original time series is now represented by the string of alphabet symbols corresponding to the subsequences extracted from the initial series. Once the time series is discretized, traditional rule mining algorithms can be applied.

Another approach in mining time series was presented by Han *et al.* [5], in which they want to discover patterns. A pattern $s = s_1, \ldots, s_p$ of period p is a sequence of itemsets. An itemset in a patter can be replaced by a wildcard *. An i-pattern is a pattern with i itemsets that are not wildcards. The mining of frequent partial periodic patterns in a time series is to discover, possibly with some restrictions, all the frequent patterns of the time series for one period or a range of specified periods. Han's method for discovering patterns in time series can be summarized as:

1. Given a time series S, scan S to find the set of frequent 1-patterns of period p by accumulating the frequency count for each 1-pattern and selecting among them those with the minimum support. Combine all 1-patterns to find a candidate max-pattern
2. Scan the time series S. During the scan, for each period segment S_i, if its hit set is nonempty (i.e. if there are matching patterns for S_i), do the following: add the max-sub-pattern (i.e. the longest sub-pattern of the max-pattern that is satisfied by S_i) into the hit set buffer if it is not already there, otherwise increment its counter. The hit set is kept in a max-sub-pattern tree (a novel data structure). The max-sub-pattern tree can then be scanned to find the frequency counts of these candidate patterns and eliminate the non-frequent ones.
3. From the hit set, derive all the frequent patterns.

The problem of mining sequential patterns has also been addressed by Zaki [13]. In this work he proposes the SPADE (Sequential PAttern Discovery using Equivalence classes) algorithm. The SPADE algorithm works as follows. First, it identifies all frequent sequences with 1 and 2 items. Then it compute all the equivalent classes for 1-sequences (i.e. all sequences for which a 1-sequence is a prefix of). For each of these equivalent classes, enumerate frequent sequences [13]. Later, Zaki extended his own work to account for constraints on the mined sequences [14]. For example the user could be interested in subsequences that occur close together or far apart. For this, he proposes c-SPADE, an algorithm that can handle restriction on the size of the subsequence being mined, restriction on the time separation between subsequences, and restrictions on the items to be included in the subsequences.

5 Data Mining and Similarity Search

In the preceding sections we have described techniques for mining association rules. We paid special attention to mining temporal and sequential data. In general, data mining is the process of discovering interesting relations among the transactions stored in the database. We call these interesting relations association rules. Similarity search, on the other hand, deals with the problem of finding a specific pattern in the database. Therefore, similarity search cannot be used to discover association rules.

The keywords here are *discover* and *search*. We mine a particular data set to *discover* all possible patterns of interest, where interest is determined by the concepts of support and confidence. We *search* the data set looking for all the objects that follow a particular pattern. While searching we are interested in finding all objects that match a goal (or query) pattern within an error bound ϵ. The difference in the size of the search space for these two problems is significantly different as well. When mining for association rules, we need to consider all possible subset of rules, which is exponential on the number of items in the database. During similarity search, the space is linear on the number of association rules. On a knowledge discovery application, the number of interesting association rules is a very small fraction of the total number of possible rules (otherwise, there is simply no knowledge to discover).

While it is a different concept than mining, similarity search can still be used in knowledge discovery applications, In particular, once the association rules have been discovered. There is no use of having a set of association rules if we are not interested in finding the set of objects in the data base that have exactly the same pattern as an association rule, or that have a pattern similar to an association rule. Similarity search can then be used to find these objects of interest. In a marketing application, for example, we may be interested in targeting a certain group of customers based on the association rules indicating that they are likely to purchase a particular product. Since the data sets are usually large, efficient similarity search techniques should be used for this task.

6 Conclusion

In this paper, we have presented a small survey of research work on the mining of association rules. We have also compared this problem to similarity search. In the context of this work, association rules are classified as *classical*, *temporal*, or *sequential*. Different approaches have been presented for discovering such association rules. Here, we have presented a few of them in an effort to provide a framework for comparing the mining of association rules to the problem of similarity search. It is curious that, while there is a myriad of initial assumptions and perspectives for this problem, after a decade of research, most of the algorithms are variations of the seminal work presented by Agrawal in 1993 [1].

While trying to classify a specific work in either the temporal association rule or on the mining of sequences we often found ourselves in a difficult situation as there is no clear division among these two topics. Some of the papers included in this work could have been inserted in both classes. This is probably because temporal databases capture the evolution of a data set. At the same time, evolution can be considered as a sequence of events.

The final result of the mining process is a set of association rules. Independently of the domain, association rules are represented as strings. This happened either because the domain of items was discrete and finite, or because the domain of values had been discretized and represented by a finite set of symbols. From this scenario, we come to wonder on why there is no work using suffix trees as this could facilitate the search and generation of frequent string patterns. Unfortunately, due to time constraints we did not have the opportunity to explore on this issue. For now the possible use of suffix trees in the process of mining for association rules will remain a research question.

References

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM-SIGMOD Conference*, pages 207–216, Washington, DC, May 1993.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Mining Sequential Patterns. In *Proceedings of the 11th Inter. Conference on Data Engineering*, Taipei, Taiwan, March 1995.

[3] Xiaodong Chen and Ilias Petrounias. Discovering Temporal Association Rules: Algorithms, Language and System. In *Proceedings of the 16th Inter. Conference on Data Engineering*, page 306, San Diego, CA, March 2000.

[4] Gautam Das, King-Ip Lin, Heikki Mannila, Gopal Renganathan, and Padhraic Smyth. Rule Discovery from Time Series. In *Knowledge Discovery and Data Mining*, pages 16–22, 1998.

[5] Jiawei Han, Guozhu Dong, and Yiwen Yin. Efficient Mining of Partial Periodic Patterns in Time Series Database. In *Fifteenth International Conference on Data Engineering*, pages 106–115, Sydney, Australia, 1999. IEEE Computer Society.

[6] Jochen Hipp, Ulrich Guntzer, and Gholamreza Nakhaedizadeh. Algorithms for Association Rule Mining - A General Survey and Comparison. *SIGKDD Explorations*, 2(1):58–64, 2000.

[7] Yingjiu Li, Peng Ning, X. Sean Wang, and Sushil Jajodia. Discovering Calendar-based Temporal Association Rules. In *Proceedings of the 8th International Symposium on Temporal Representation and Reasoning*, 2001.

[8] R. J. Miller and Y. Yang. Association Rules over Interval Data. In *Proceedings of the 1997 ACM-SIGMOD Conference*, pages 452–462, Tucson, AZ, May 1997.

[9] C.P. Rainsford and J.F. Roddick. Adding Temporal Semantics to Association Rules. In *3rd. European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'99)*, pages 504–509, Prague, 1999.

[10] C.P. Rainsford and J.F. Roddick. Visualisation of Temporal Interval Association Rules. In *2nd. International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2000)*, pages 91–96, Hong Kong, 2000.

[11] Abdullah Uz Tansel and Necip Fazil Ayan. Discovery of Association Rules in Temporal Databases. In *4th International Conf. on Knowledge Discovery and Data Mining (KDD'98) Distributed Data Mining Workshop*, New York, USA, August 1998.

[12] Wei Wang, Jiong Yang, and Richard Muntz. Temporal Association Rules with Numerical Attributes. Technical Report 990011, UCLA CSD, 1999.

[13] Mohammed J. Zaki. Efficient Enumeration of Frequent Sequences. In *7th International Conference on Information and Knowledge Management*, pages 68–75, Washington, DC, November 1998.

[14] Mohammed J. Zaki. Sequence Mining in Categorical Domains: Incorporating Constraints. In *9th International Conference on Information and Knowledge Management*, pages 422–429, Washington, DC, November 2000.