Predictive Sequence Miner in ILP Learning

Carlos Abreu Ferreira[1], João Gama[2] and Vítor Santos Costa[3]

[1] LIAAD-INESC and ISEP - Polytechnic Institute of Porto, Porto, Portugal
[2] LIAAD-INESC and Faculty of Economics - University of Porto, Porto, Portugal
[3] CRACS-INESC and Faculty of Sciences - University of Porto, Porto, Portugal

Abstract. In this work we present an optimized version of `XMuSer`, an ILP based framework suitable to explore temporal patterns available in multi-relational databases. The main idea behind `XMuSer` consists of exploiting frequent sequence mining, an efficient and direct method to learn temporal patterns in the form of sequences. The efficiency of `XMuSer` comes from a new coding methodology and on the use of a predictive sequential miner, which finds discriminative frequent patterns. After finding the discriminative sequences, we map the most interesting ones into a new table that encodes the multi-relational temporal information. The original database is enlarged with a new table that encodes the temporal information in the form of sequences. The last step of our framework consists of applying an ILP algorithm to learn a theory on the enlarged relational database. We evaluate our framework by addressing three classification multi-relational problems. Overall, we observe clear advantages when exploiting temporal information.

1 Introduction

Multi-relational databases are widely used to represent and store data. A multi-relational database is often composed by a *target* table and by a number of *fact* tables. The target table will represent the main objects of interest (say, patients in a medical domain); fact tables will represent the information being accumulated about the entities in the target table (say, medical visits or drug usage in the medical domain). We expect target tables to be relatively stable or to grow slowly over time; in contrast, fact tables may grow quickly. Moreover, quite often the information stored in fact tables is time-based and consists of *sequences* that reflect the evolution of a phenomenon of interest.

The main goal of this work is to exploit heterogeneous temporal information stored in the multi-relational sequences of events. To do so, we propose an optimized version of the `XMuSer` (eXtended MUlti-relational SEquential patteRn knowledge learning) framework [3] that encodes *timed data*, stored in one or several fact tables, into a separate sequence relation, uses a predictive sequence learner to find the most interesting such sequences, maps back the sequences to the relational database, and then learns a theory on the extended multi-relational database. The extended database thus contains all primitive relations and an additional relation that stores temporal patterns for each example.

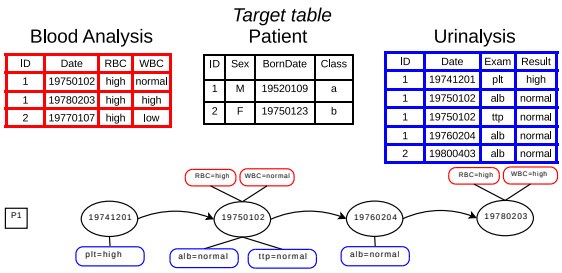

Fig. 1. Database Relations(up) and Temporal Patient Events for Patient One(down). ID is the patient ID, Date is in numeric format, RBC and WBC are blood parameters, and we show alb, plt and ttp urine exams.

This methodology allows us to explore multi-relational datasets that have different types of timed data, either sequence data or time-series data. On the one hand, we can benefit from computationally efficient sequential miners such as cSpade [6] to find the most predictive sequential patterns. On the other hand, we still have access to the original data and can take advantage of the flexibility of Inductive Logic Programming (ILP) to learn in the extended multi-relational dataset. Indeed, we argue that the first step provides a good insight into the search space, and may enable `XMuSer` to perform better than classical ILP based algorithms in large search spaces. We should observe that the sequence miner and ILP learning algorithm are decoupled and we can use other highly efficient algorithms such as CloSpan [5], that find closed sequential patterns.

We evaluate our framework by addressing three classification problems. Moreover, we map each one of three different types of sequential patterns: frequent, closed or maximal.

In the following section we introduce and evaluate the performance of the `XMuSer` algorithm. In the last section we present some conclusions.

2 The `XMuSer` Algorithm

Our framework has five main steps. In the first phase, if needed, we code the temporal data into a sequence database. In a second phase, we run a sequence miner to find all predictive sequential patterns. In a third phase, we sort the predictive sequential patterns using the chi-square statistic. In a fourth phase, we select the top-k most interesting sequential patterns and, for each example in the target table, we built a relation where the example is characterized by presence or absence of the k most interesting sequential patterns. Last, we learn a theory on the enlarged database, where enlarged database is the union of the original database with the new *sequence relation*.

Next, we explain each one of the major components in self-contained subsections. Throughout, we follow an illustrative example, a classification problem,

at Figure 1. This example is inspired on the relational Hepatitis dataset. The example has three tables registering the follow-up of two patients. One of the tables is the target table, named *Patient*, where each record describes each patient, identified by a masked ID, and registers the class of each patient. The other two tables are fact tables registering timed blood analysis and urinalysis examinations.

Data Coding. Our algorithm takes a multi-relational dataset as input, represented as a database of Prolog facts. To explore the richness of this representation, and namely temporal patterns, we introduce a strategy that converts multi-relational timed data into an amenable sequence database. First, we find all relations that have temporal records. Second, we sort the records in these relations by time order. We thus obtain a chronological sequence of multiple events for each example. Figure 1 shows patient one event sequence. The sequence includes a sequence of blood and urine analysis. Third, we build a temporal attribute-value sequence for each example. In this new sequence each itemset corresponds to all records registered at a given date/time. We have, for patient one, *(plt=high) (rbc=high, wbc=normal, alb=normal, ttp=normal) (alb=normal) (rbc=high, wbc=high)*. We then define a one-to-one coding map $f : Attributes \times Values \longrightarrow \mathbb{N}$. This mapping associates an *unique* number to each attribute-value pair. In the example, we use the map to code the attribute-value sequence into an integer number sequence. The mapping assumes discrete attributes (continuous attributes will be discretized beforehand). In Table 1 we present the transformed sequence database registering the coded sequence of patients one and two and four other patients, that were not present in the example database at Figure 1, but will be useful in the subsequence of the algorithm description. In this table, each sequence tuple corresponds to an example in the target table and each itemset in the sequence corresponds to all one-time events. In the third column we show the labels of each sequence, we have two classes, class **a** and **b**. In the example, we define the one-to-one map as $f(rbc, low) = 1$, $f(rbc, normal) = 2$, $f(rbc, high) = 3$, $f(wbc, low) = 4$, $f(wbc, normal) = 5$, $f(wbc, high) = 6$, $f(alb, normal) = 7$, $f(ttp, normal) = 8$, $f(plt, high) = 9$ and patient one sequence of events is thus coded as (9) (3 5 7 8) (7) (3 6).

Finding Frequent Sequential Patterns. When mining the sequence database we aim at finding interesting patterns that augment the descriptive power of the raw data and that are useful to build highly accurate models.

Table 1. Sequence Database

ID	Sequence	Class
1	(9) (3 5 7 8) (7) (3 6)	a
2	(3 4) (7)	b
3	(1 5) (4)	b
4	(3 5) (7)	a
5	(5 8) (7)	a
6	(4)	b

Table 2. Predictive Sequential Patterns found

		Support Value			
Rank	Seq. Patt.	Overall	Class **a**	Class **b**	Chi-square
1	**(5)(7)**	3	3	0	**6.000**
2	**(4)**	3	0	3	**6.000**
3	**(3 5)**	2	2	0	**3.000**
4	(5)	4	3	1	3.000
5	(7)	4	3	1	3.000
6	(3)(7)	3	2	1	0.667
7	(3)	3	2	1	0.667

Table 3. Sequence Relation

ID	S_1	S_2	S_3
1	1	0	1
2	0	1	0

Our first step is to run a sequence miner on the sequence database to find predictive frequent sequences, the ones having, in at least one class partition, a support value equal or higher than a user defined threshold. Following the illustrative example, if we set the support value to 50% and run cSPADE algorithm to find sequential patterns available in the sequence database (see Table 1), we get the sequential patterns presented in Table 2. cSPADE algorithm outputs the overall support of each pattern and the support of each pattern in each class partition. We slightly modify cSPADE to compute and output a measure of interest for each pattern. Here, we set cSPADE to output the chi-square statistic for each pattern.

Considering the nature of the sequence miner, the type and the dimension of problem being addressed we usually get a large number of sequential patterns and some of these patterns can be redundant and/or uninteresting.

Sort Sequential Patterns. To select the most interesting patterns and class correlated patterns to build the final classification model, we sort all the sequential patterns found using a metric (see Table 2). In this example, we sort the patterns according to the chi-square statistic value. The k most interesting patterns (the top ones) will be used to build the final classification model.

Mapping Back Interesting Sequences. In order to best explain our procedure we present Table 3, the sequence relation that was obtained by applying our mapping procedure on the example data. This table has four attributes, the example ID and the three most interesting sequences (the top three patterns of Table 2). If we get ties we select the longest pattern.

We develop a mapping strategy that maps each one of the selected sequential patterns into a Boolean attribute and, portray this mapping as a new table. This new table, named *sequential relation*, has an entry for each example on the target table and has $k+1$ attributes: the top-k most interesting features and the example ID, usually the target table primary key. To compute the value of each attribute we use the information coded in the sequence database and subsequence definition. If the sequence associated with the new attribute is a subsequence of the example sequence at the sequence database, the new attribute takes value one. Otherwise, it takes the value zero.

Learning a Theory. In this last step we learn a theory. We take all the primitive tables and the new sequential relation as input to an ILP algorithm, such as Aleph [4], to learn a set of clauses. These clauses can therefore use the primitive tables and the new one, that encodes the time information.

Table 4. Mean Generalization Accuracy: `XMuSer` against Stand-alone Aleph

	λ	`XMuSer` (With Aleph)			Stand-alone Aleph	Wilcoxon p-value		
	λ	freq.	clo.	max.		freq.	clo.	max.
Hepatitis Subtype	0.9	0.78(0.10)	0.78(0.10)	0.78(0.10)	0.78(0.11)	0.539	0.539	0.919
	0.8	0.79(0.10)	0.80(0.10)	0.79(0.08)		0.083	0.020	0.322
Hepatitis Fibrosis	0.9	0.64(0.04)	0.63(0.05)	0.63(0.05)	0.58(0.09)	0.032	0.032	0.105
	0.8	0.58(0.11)	0.60(0.08)	0.62(0.06)		0.760	0.919	0.262
Financial	0.9	0.74(0.06)	0.74(0.06)	0.74(0.05)	0.71(0.07)	0.041	0.041	0.020
	0.8	0.77(0.06)	0.77(0.06)	0.75(0.06)		0.008	0.008	0.041

Following the illustrative example, an example of a clause we can find is: *patient_info(A,B,C,a):- blood_analysis(A,D,high,normal), urinalysis(A,E,plt,high), sequence_relation(A,1,F,G).*

This clause has the predicate *patient_info* at the head, and a call to the predicate *blood_analysis*, the predicate *urinalysis* and the predicate associated with the sequence relation, predicate *sequence_relation*, as the clause body. The clause explains (or covers) patient number one in the database.

Experimental evaluation. We evaluated our algorithm in two real-life datasets obtained from the PKDD data-mining competitions[4], the Hepatitis dataset and the Financial dataset. We experimented three classification problems: Hepatitis Subtype and Hepatitis Fibrosis Degree (we translate this multi-class problem into a binary problem, see [3]), in the Hepatitis dataset, and the prediction of successful loans, in the Financial dataset.

Throughout, we used cSPADE algorithm to find all frequent and predictive sequences and Aleph algorithm to learn a logical theory. We applied the following predefined parameter configuration: we map the 5 most interesting sequential patterns, the top-5, and we test two different support values $\lambda = 90\%$ and $\lambda = 80\%$. For comparison purposes we also run the stand-alone Aleph algorithm to solve each problem. Furthermore, besides presenting XMuSer results using all sequential patterns found by cSPADE, we present results using two subsets of sequential patterns, the closed set and the maximal set. The closed and maximal sequences were obtained using a naive post-processing strategy. We evaluate our framework using a ten-fold cross-validation procedure and compute: the mean generalization accuracy and standard deviation. We also compute the Wilcoxon hypothesis test p-value. We use the same background knowledge and bias when running the Aleph algorithm, either as the last component of `XMuSer` or as stand-alone. The major difference is that in `XMuSer` we introduce an extra relation, the *sequence relation*. The bias is relatively simple, relying on the predefined tables available in the enlarged database.

We address the three problems using the original multi-relational datasets without performing preprocessing steps, such as balancing the datasets. In Table 4 we present experimental results for $\lambda = 90\%$ and $\lambda = 80\%$. Lower values of λ cause a memory explosion. Further experiments with other values of k do not show significant changes in performance.

[4] see http://lisp.vse.cz/challenge/CURRENT/

We believe that we obtained very good results. In all three classification problems, `XMuSer` obtained better or equal results than the stand-alone Aleph algorithm. In each one of the three problems we get, for some support value, a significant win over the stand-alone Aleph algorithm, at a confidence level 5%. Moreover, the best results were obtained when we map a subset of the frequent or a subset of the closed sequential patterns. Furthermore, we run `XMuSer` by mapping a different number of patterns, other than 5, and we can say that the best results were obtained when we map less than 25 sequential patterns. These results are among the best that we could find in related work (see [2]).

3 Conclusions

We presented an optimized version of `XMuSer`, a multi-relational framework that explores temporal information stored in a database. The methodology is an effective alternative to previous ILP-based approaches that incorporate timed data in the final first-order theory by either using time aggregation strategies [1], like time window aggregation, or by refining ILP clauses or predicates that explicit explore time. Differently from these approaches, our methodology, can take advantage of the strengths of sequential miners to explore efficiently any kind of timed data.

Moreover, we develop a new methodology to translate any multi-relational timed database into a sequence database and that we develop a new strategy to explore efficiently timed patterns in an ILP framework. Our ILP based framework gains both from the descriptive power of the ILP algorithms and the efficiency of the sequential miners.

Acknowledgments: This work was supported by the Portuguese Foundation for Science and Technology (FCT) under the projects KDUS (PTDC/EIA-EIA/098355/2008) and HORUS (PTDC/EIA-EIA/100897/2008).

References

1. Ferreira, C.A., Gama, J., Costa, V.S.: RUSE-WARMR: Rule Selection for Classifier Induction in Multi-relational Data-Sets. In: 20th International Conference on Tools with Artificial Intelligence. vol. 1, pp. 379–386. Ohio, USA (2008)
2. Ferreira, C.A., Gama, J., Costa, V.S.: Sequential pattern mining in multi-relational datasets. In: Proc. of the 13th Conference of the Spanish Association for Artificial Intelligence, LNCS 5988. pp. 121–130. Seville, Spain (2010)
3. Ferreira, C.A., Gama, J., Costa, V.S.: Constrained sequential pattern knowledge in multi-relational learning. In: 15th Portuguese Conference on Artificial Intelligence, LNCS. p. (accepted for publication) (2011)
4. Srinivasan, A.: The Aleph Manual (2003), `http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/aleph.html`
5. Yan, X., Han, J., Afshar, R.: Clospan: Mining closed sequential patterns in large datasets. In: Proc. of the Int. Conf. on Data Mining. pp. 166–177. CA, USA (2003)
6. Zaki, M.J.: Sequence mining in categorical domains: Incorporating constraints. In: CIKM. pp. 422–429 (2000)