Discovering Temporal Rules from Temporally Ordered Data

Kamran Karimi and Howard J. Hamilton

Department of Computer Science
University of Regina
Regina, Saskatchewan
Canada S4S 0A2
{karimi, hamilton}@cs.uregina.ca

Abstract. We introduce a method for finding temporal and atemporal relations in nominal, causal data. This method searches for relations among variables that characterize the behavior of a single system. Data are gathered from variables of the system, and used to discover relations among the variables. In general, such rules could be causal or acausal. We formally characterize the problem and introduce RFCT, a hybrid tool based on the C4.5 classification software. By performing appropriate preprocessing and postprocessing, RFCT extends C4.5's domain of applicability to the unsupervised discovery of temporal relations among temporally ordered nominal data.

1. Introduction

We consider the problem of discovering relations among a set of variables that represent the state of a single system as time progresses. Given a sequence of temporally ordered records, with or without an explicit time variable, the goal is to identify as many cases as possible where two or more variables' values depend on each other. We may want to describe the system, predict future behavior, or control some of the variables by changing the values of other variables. For example, a description might be based on the observation that (y = 5) is always true when (x = 2). From this description, we could predict the value of y as 5 when we see that x is 2. This description is an example of *association* between two values, where observing the value of one variable allows us to predict the value another variable, without one necessarily causing the other. Alternatively, from the description, we could devise the rule: **if** {(x = 2)} **then** (y = 5), and use forward chaining to predict that setting the value of x to 2 will result in y becoming 5. This rule can be interpreted as a *causal relation*.

Previous research has emphasized causality mining, time series, and event sequences. A *causality miner* attempts to generate a description of the causal relations in the data. TETRAD [13] and CaMML [8, 16] are two causality miners based on Bayesian networks [3]. The suitability of using Bayesian networks for mining causality is a continuing source of debate [4, 10, 15]. In [11] the author claims that it is possible to discover and express causality with mathematical tools, while in [2] the claim is that the ability to extract correct causal relations from any given set of observed data is doubtful. At the least, considerable disagreement exists about the concept of causality.

Interpreting association relations as causal relations, as done by applications such as TETRAD, requires justification. The main trend in causality mining involves using the statistical concept of conditional independence as a measure of the control one variable may have over another. For example, given the three variables x, y, and z, if x is independent from y given z, that is, $P(x, y \mid z) = P(x \mid z)$, then we can conclude that x is not a direct cause of y, and y is not a direct cause of x. In other words, z separates x and y from each other. This basic concept is used in Bayesian Networks to build graphs that show the conditional dependence of the variables under observation. This graph is then *interpreted* as signifying causal relations. The notion of conditional independence is void of time. The proponents of this mainstream method use temporal information, if it is available, to place constraints on the relationships among the variables (e.g., if we know x always appears before y, then y cannot be a cause of x), but time is not essential to the working of their algorithms. Leaving out time when dealing with the notion of causality seemed counterintuitive to us.

In this paper, we introduce the principles of RFCT's operation. RFCT [9] is a new hybrid tool based on the C4.5 classification software [12]. The name RFCT is a loose acronym for "Rotate, Flatten, apply C4.5, and enforce Temporal constraints." By performing appropriate preprocessing (rotation, and flattening) and postprocessing (applying temporal constraints), RFCT extends C4.5's domain of applicability to the unsupervised discovery of temporal relations among nominal data. We chose C4.5 because it is available in source code, and has been used widely in the literature.

The remainder of this paper is organized as follows. Section 2 formally presents the problem and the notation used in this paper. Section 3 introduces the RFCT method and software, and Section 4 concludes the paper.

2. Formal Representation of the Problem

Given a set of temporally ordered observation records $\mathbf{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_T\}$, the problem is to find a set of relations, as described in more detail below. Each record $\boldsymbol{d}_t = \langle d_{t1}, \ldots, d_{tm} \rangle$ gives the values of a set of variables $V = \{v_1, \ldots, v_m\}$ observed at time step t. Each relation predicts or constrains the value of one variable v_j, $1 \le j \le m$. The data values are assumed to be nominal, that is, symbolic values such as "yes," or numeric values, such as "1," representing different categories with no ordering defined among the values. It is assumed that no variable explicitly holds a time value. Unlike the real-world data studied by many researchers [1, 2, 14], we assume that a temporal order exists among the records.

The goal is to discover relationships among the past and present values of the variables. Of particular interest is any relation that can be specified as a rule that determines the value of some variable v_j at time t based on the previous or current values of the variables. The current value of v_j may not be used to determine its own value. If a rule predicts v_j's value at time step t based only on the values of other variables observed at time step t, then the rule is an *atemporal rule*. Alternatively, if the rule is based on the value of variables from previous time steps, it is called a *temporal rule*. Since one common sense definition of a causal relation involves the passage of time between the cause and the effect, we arbitrarily define such rules to specify causal relations. By this assumption, for a rule to be causal, the variables from the past must be indispensable; otherwise it will turn into an atemporal rule.

For any t, $1 \leq t \leq T$ we distinguish between the current time step t, and the previous time steps. We define the previous set $P(t) = \{d_{ki} \mid 1 \leq k \leq t, 1 \leq i \leq m\}$ to represent all observations made from time 1 to time t.

For practical reasons, we concentrate on a limited window of past observations. For any given time step t, the *window* includes the preceding w-1 time steps, plus the current time step, for a total of w time steps. We assume that only information in this window is relevant to predicting the value of some variable v_j at time t. The *window set* $P_w(t) = \{d_{ki} \mid w \leq t \ \& \ t\text{-}w\text{+}1 \leq k \leq t, 1 \leq i \leq m\}$ represents all observations in the window.

The *flattening* operator $F_w(t,\boldsymbol{D})$ takes as input a window size w, a current time t, $t \geq w$, and the input records $\boldsymbol{D}$, and gives a single flattened record $\boldsymbol{z} = \{z_{ki} \mid d_{pi} \in P_w(t) \ \& \ k = w\text{-}t\text{+}p \ \& \ z_{ki} = d_{pi}\}$. The flattened record contains the most recent w records. Given T input records, F_w can be applied with any t, $w \leq t \leq T$, thus turning the original T records into T-w+1 flattened records. Each flattened record contains mw fields.

The F_{w} operator renames the time index values so that in each record, time is measured relative to the start of that record only. In each flattened record, the time index ranges from 1 to w. The flattened records are thus independent of the time variable t. To create a variable corresponding to each member of a flattened record, we define the set $V_w = \{v_{ki} \mid 1 \leq k \leq w, 1 \leq i \leq m\}$. The variables in V_w correspond to w consecutive values of the variables in the set V. With mw members, V_w has a one to one correspondence with every flattened record $\boldsymbol{z}$.

To use tools that do not consider any temporal order to be present among the input records, we flatten the T records and use the new T-w+1 records as input. Each flattened record contains all information available in a window of w time steps. The flattened records can thus be processed by a tool that ignores temporal relationships. Before flattening, time goes "down" to the next record, but after flattening, time moves horizontally within the same record, hence the name "flattening."

We look for a relation R: $S \rightarrow \{v_j\}$. The only restriction is that $v_j \notin S$. We consider S to be minimal, that is, there is no S' such that $S' \subset S$ and R: $S' \rightarrow \{v_j\}$. We define R to be of two different types:

- Atemporal: R: $S \rightarrow \{v_j\}$, $S \subseteq V - \{v_j\}$.
- Temporal: R: $S \rightarrow \{v_{wj}\}$, $S \subseteq V_w - \{v_{wj}\} \ \& \ S \cap (V_w - \{v_{w1},\ldots, v_{wm}\}) \neq \varnothing$.

If S includes variables from only the current time step then we call R an atemporal relation. If S includes variables from previous time steps, then we call R a temporal relation. For such relations to be discovered, we assume that relations among data persist over time, and thus are repeatable. For the use of a time window to be justified, we assume that each relation is of limited duration. If we cannot prove that set S is minimal, then a specified temporal relation may actually be equivalent to an atemporal relation that can be found by eliminating unnecessary variables. In addition, we require the relation to obey the temporal order by referring to the variables according to the order of their appearance. This is clarified in Section 3.

3. Method

In this section, we describe RFCT's method for finding relations among variables in a single system. For a practical comparison of this method with TETRAD, see [5]. RFCT is an unsupervised variant of C4.5, a well-known decision rule and tree discoverer. Another rule discoverer could be used in RFCT instead of C4.5.

C4.5 first creates a decision tree that can be used to predict the value of one variable (the *decision attribute*) from the values of other variables (the *condition attributes*). C4.5 uses a greedy algorithm with one look-ahead step, based on information entropy of the condition attributes, to build a decision tree. Decision rules are derived from the decision tree with a program called "c4.5rules." After the rules have been created, the *consult* program in the C4.5 package can be used to execute the rules. It prompts for the condition attributes and then outputs the appropriate value of the decision attribute.

Each decision rule generated by C4.5 is equivalent to a simple predicate, such as **if** {(a = 1) **and** (b = 2)} **then** {(*class* = 4)} [83.2%]. The variables a and b may be causing the value of *class*, or they may be associated with it because of some other reason. A certainty value, assigned to each rule, specifies the confidence of that rule. In the example rule just given, the certainty value is 83.2%. C4.5 has been modified to output its rules as Prolog statements, which can be executed by a Prolog Interpreter with little or no change [6, 7].

For our research, we created a modified version of C4.5, called C4.5T**,** which ensures that each decision rule references condition attributes in temporal order, and adds temporal annotations to both condition and decision attributes. An implicit assumption in C4.5 is that all condition attributes are available at the same time. As a decision tree is being built, the condition attributes may be used in any order. C4.5T performs two additional steps (listed below). To illustrate these steps, suppose the input consists of data records that contain the values of 4 variables: <a, b, c, d>, flattened with a time window of w = 3. Suppose C4.5 generates the decision rule: **if** {(a = 3) **and** (b = 2) **and** (c = 6)} **then** (d = 8), where condition attributes a and c are derived from the first unflattened record in the window, condition attribute b is derived from the second time step, and the decision attribute d is derived from the third. The two additional steps performed by C4.5T are as follows.

1. Reorder the arguments in the rule, so that the attributes appear in the same temporal order as their respective unflattened records. For example, reorder the rule given above as: **if** {(a = 2) **and** (c = 6) **and** (b = 3)} **then** (d = 8).
2. Add "At Time n:" before the attributes that happen at time step n in the flattened record. Intuitively, this can be considered the reverse of the flattening operation. For example, annotate the rule given above as follows:
   if {At Time 1: (a = 2) **and** (c = 6) **and** At Time 2: (b = 3)} **then** At Time 3: (d = 8).

C4.5T has been implemented as a modification to c4.5rules, a program in the C4.5 Release 8 package that generates rules from decision trees, to output temporal rules. The user can specify the number of records involved in the flattening process via a new option -T (Time window). The modified c4.5rules program then generates the rules as usual, but before outputting them, it sorts the decision attributes of each rule according to their time of occurrence. It then prints out the rules, along with the temporal information as outlined in the example given above. We have also modified the c4.5 program (C4.5's decision tree builder) to consider the temporal order while building the tree [9].

The RFCT algorithm is introduced here.

```
Algorithm RFCT.
Input: a set of $m$ variables $V$; a data set $\boldsymbol{D}$ consisting of $T$ records with $m$ values;
and a window size $w$.
Output: a set of decision rules $R$.
for $j$ = 1 to $m$
    $\boldsymbol{D'}$ = rotate the values in each record $\boldsymbol{d} \in \boldsymbol{D}$ such that value in $v_j$ is the last
    member of $\boldsymbol{d}$.
    Flat = the sequence of values yielded by $F_w(t,\boldsymbol{D'})$, for all $t$, $w \le t \le T$.
    $\boldsymbol{R} = \boldsymbol{R} \cup$ C4.5T(Flat)
end for
return $\boldsymbol{R}$
```

Since C4.5 is a supervised algorithm, the user must specify the decision attribute. To avoid providing this supervision, RFCT is set to apply C4.5T to every possible decision attribute in turn. Alternatively, the user can choose exactly which attributes should be considered as decision attributes. To allow temporal relations of up to *w* steps to be discovered, preprocessing using the flattening operator with window size *w*, is applied to the input. RFCT provides the user with temporal rules as its output.

4. Concluding Remarks

We introduced a new unsupervised learning tool called RFCT, which is based on C4.5 and relies on straightforward methods to extend its abilities. It is specifically meant for cases where data is generated sequentially by a single source. RFCT is not meant to replace software such as TETRAD, as its domain of applicability is more restrained (temporal output from a single source vs. data generated by many sources with no regard to the order in which they were gathered). RFCT is written in Java and runs in any environment that supports the Java runtime environment and has a graphical user interface, including Microsoft Windows and XWindow. The package includes full source code and online help, and is freely available from http://www.cs.uregina.ca/~karimi/downloads.html or by contacting the authors.

References

1. Bowes, J., Neufeld, E., Greer, J. E. and Cooke, J., A Comparison of Association Rule Discovery and Bayesian Network Causal Inference Algorithms to Discover Relationships in Discrete Data, *Proceedings of the Thirteenth Canadian Artificial Intelligence Conference (AI'2000)*, Montreal, Canada, 2000.
2. Freedman, D. and Humphreys, P., *Are There Algorithms that Discover Causal Structure?*, Technical Report 514, Department of Statistics, University of California at Berkeley, 1998.

3. Heckerman, D., *A Bayesian Approach to Learning Causal Networks*, Microsoft Technical Report MSR-TR-95-04, Microsoft Corporation, May 1995.
4. Humphreys, P. and Freedman, D., The Grand Leap, *British Journal of the Philosophy of Science 47*, pp. 113-123, 1996.
5. Karimi, K. and Hamilton, H.J., Finding Temporal Relations: Causal Bayesian Networks vs. C4.5, *The Twelfth International Symposium on Methodologies for Intelligent Systems (ISMIS'2000)*, Charlotte, NC, USA, October 2000.
6. Karimi, K. and Hamilton, H.J., Learning With C4.5 in a Situation Calculus Domain, *The Twentieth SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence (ES2000)*, Cambridge, UK, December 2000.
7. Karimi, K. and Hamilton, H.J., Logical Decision Rules: Teaching C4.5 to Speak Prolog, *The Second International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2000)*, Hong Kong, December 2000.
8. Kennett, R.J., Korb, K.B., and Nicholson, A.E., Seabreeze Prediction Using Bayesian Networks: A Case Study, *Proc. Fifth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'01)*. Hong Kong, April 2001.
9. Karimi, K. and Hamilton, H.J., RFCT: An Association-Based Causality Miner, *The Fifteenth Canadian Conference on Artificial Intelligence (AI'2002)*, Calgary, Alberta, Canada, May 2002.
10. Korb, K. B. and Wallace, C. S., In Search of Philosopher's Stone: Remarks on Humphreys and Freedman's Critique of Causal Discovery, *British Journal of the Philosophy of Science 48*, pp. 543-553, 1997.
11. Pearl, J., *Causality: Models, Reasoning, and Inference*, Cambridge University Press. 2000.
12. Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
13. Scheines, R., Spirtes, P., Glymour, C. and Meek, C., *Tetrad II: Tools for Causal Modeling*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.
14. Silverstein, C., Brin, S., Motwani, R. and Ullman, J., Scalable Techniques for Mining Causal Structures, *Proceedings of the 24th VLDB Conference*, pp. 594-605, New York, USA, 1998.
15. Spirtes, P. and Scheines, R., Reply to Freedman, In McKim, V. and Turner, S. (editors), *Causality in Crisis*, University of Notre Dame Press, pp. 163-176, 1997.
16. Wallace, C. S., and Korb, K. B., Learning Linear Causal Models by MML Sampling, *Causal Models and Intelligent Data Management*, Springer-Verlag, 1999.