International Journal of Engineering, Management, Humanities and Social Sciences Paradigms (IJEMHS)
Volume 30, Issue 04, Quarter 04 (Oct-Nov-Dec 2018)
ISSN (Online): 2347-601X
www.ijemhs.com

Predicting Telecommunication Equipment Failures from Sequences of Network Alarms

Bhagaban Sri Ramakrishna[1], Janmoni Borah[2] and Prangya Padhi[3]

[1]Aryan Institute of Engineering & Technology, Bhubaneswar, Odisha

[2]NM Institute of Engineering and Technology, Bhubaneswar, Odisha

[3]Capital Engineering College, Bhubaneswar, Odisha

Abstract

The computer and telecommunication industries rely heavily on knowledge-based expert systems to manage the performance of their networks. These expert systems are developed by knowledge engineers, who must first interview domain experts to extract the pertinent knowledge. This knowledge acquisition process is laborious and costly, and typically is better at capturing qualitative knowledge than quantitative knowledge. This is a liability, especially for domains like the telecommunication domain, where enormous amounts of data are readily available for analysis. Data mining holds tremendous promise for the development of expert systems for monitoring network performance since it provides a way of automatically identifying subtle, yet important, patterns in data. This case study describes a project in which a temporal data mining system called Time weaver is used to identify faulty telecommunication equipment fromlogs of network alarm messages.

Introduction

One such expert system is AT&T's ANSWER system (Weiss, Ros and Singhal 1998). ANSWER (Automated Network Surveillance with Expert Rules) is responsible for monitoring the 4ESS switches that route the majority of the traffic in the AT&T network. When a component within a switch experiences a problem, the switch generates an alarm and forwards it to one of AT&T's two technical control centers. At the technical control center, the alarm is inserted into a relational database and forwarded to ANSWER. ANSWER then analyzes the alarm using rules acquired from domain experts. If ANSWER determines that the alarm requires some action be taken, it forwards an alert describing the problem to a technician for further processing. Because the 4ESS switches in the AT&T network generate in excess of 100,000 alarms per week, it is critical that alarms corresponding to non-recurring transient problems be filtered so they do not result in an alert. One of the most important functions of ANSWER is to distinguish between recurring and non-recurring faults.

A key **business goal** of ANSWER is to minimize the number of service affecting incidents—such as blocked or lost calls—while keeping development, personnel and maintenance costs at an acceptable level. Unfortunately, expert systems such as ANSWER require the use of highly trained knowledge engineers to extract knowledge from domain experts and to encode the

knowledge in a useable form. This knowledge acquisition process is time-consuming and expensive, and often does not capture important quantitative relationships. For example, a human expert may know that the occurrence of several **`type-A`** messages indicates a problem, but may not know specifically how many should be required, or within what time interval, before action is required. Data mining offers an intriguing solution, by making it possible to acquire knowledge directly from the network alarm data. The **data-mining task** in this case study is to identify patterns in network alarm logs that can be used to predict telecommunication equipment failures. These patterns can then be incorporated as rules into the existing ANSWER system.

The data mining team for this project was comprised of a single individual who was familiar with the telecommunication domain through previous involvement with the ANSWER system. The project took approximately two years, of which most of the time was spent developing new data mining software.

The KDD Process

The KDD process employed in this case study is very similar to the process used in many other projects. The steps in this process, as they relate to our project, are described in this section.

Understanding the Data Mining Problem

The first step in the process involves gaining an understanding of the application domain and the goals of the KDD task. In this case, such an understanding had been acquired through previous work on the ANSWER expert system. The goal of the project was also influenced by a previous effort that attempted to use data mining techniques to predict *catastrophic* failures of 4ESS switches (Weiss, Eddy and Weiss 1998). A catastrophic failure is one in which the functioning of the entire 4ESS switch is compromised. The previous effort was seriously hampered by the rarity of catastrophic failures—only 50 such failures had been recorded and were available for study. Because of this experience, and because we wanted our results to be more widely applicable, we focused on the more general goal of predicting individual component failures.

The data mining task, as described in the previous section, is not well defined since neither the term "prediction" nor "pattern" were defined. To be meaningful, a prediction must apply to a specific time period. A prediction of a component failure is said to be correct if and only if the prediction occurs more than *warning time* and less than *monitoring time* before the actual failure. For example, given a warning time of one minute and a monitoring time of eight hours, a prediction must occur between one minute and eight hours prior to the actual failure for the failure to be considered successfully predicted. The warning time parameter ensures that there is sufficient time to respond to a failure prediction prior to the actual failure and the monitoring time parameter allows the user to control the specificity of the prediction. A prediction is "issued" if one of a pre-specified set of patterns (i.e., patterns identified by the data mining process) occurs in the stream of alarms coming from the 4ESS switch.

Patterns are described using a pattern language developed for this project. Key features of this language were identified based on knowledge of the telecommunication domain. First, because alarms can be generated as a result of unrelated problems, the pattern language must provide a way of specifying a pattern so that the *presence* of an alarm does not prevent a pattern from matching. Secondly, because a fault may manifest itself in slightly different ways at different times—perhaps due to different activity occurring outside of the faulty component—there must be a way of specifying a pattern without fixing the order of all alarms within the pattern. Finally, because time is such an important factor, and because the behavior of the system will change in response to a fault, it must be possible to associate a time period with each pattern. Given these three requirements, it should be possible to specify the pattern "3 **`type-A`** alarms and 2 **`type-B`** alarms occur, in any order, within a 5 minute period". The pattern language is described in additional detail later in this chapter.

Selecting a Target Dataset

The target dataset was formed by collecting two weeks worth of alarms from the database at one of the two technical control centers. The resulting dataset contained 148,886 alarms, where each alarm contained approximately 20 variables. Based on knowledge of the domain and what variables are most important for diagnosis, five variables were selected to describe each alarm. These variables are 1) the time the alarm was generated, 2) a unique identifier for the device associated with the alarm, 3) the type of device, 4) the diagnostic code associated with the alarm, and 5) the severity of the alarm. Thus, an alarm is represented by the tuple <time, device-id, device-type, diag-code, severity>. There are several dozen types of devices in a 4ESS switch, hundreds of diagnostics codes, and three severity levels (warning, minor, and major). Because the 4ESS switch recognizes component failures and automatically generates alarms for such failures, no additional effort was required in order to record these failures in the alarm stream.

Preprocessing and Transforming the Data

Components can fail for a variety of reasons. Because there is no need to distinguish between different types of failures, the various failure alarms were replaced with a common failure alarm. Also, because routine maintenance testing will cause failed components to generate additional failure alarms, a simple software program was applied to the target dataset to prune the "redundant" failure alarms. The resulting dataset yielded 1045 failure alarms corresponding to 1045 *distinct* component failures. The data mining task is then to predict these failure alarms.

Data Mining

The next major step in the KDD process involves selecting the data mining task and the algorithm to be applied. For this project, the data mining task is clearly a *prediction* task. Because each alarm record represents an event—an observation that occurs at a specific instant in time—we refer to this as an *event prediction* task. A temporal data mining algorithm is required because the dataset contains events, not examples, and because predicting component failures will require the identification of temporal relationships in the data. Other important considerations in selecting a data mining method include the fact that the component failures are *rare* and known to be *difficult* to predict. Given these characteristics, the data mining algorithm may need to find patterns that occur very infrequently in the data and have relatively low predictive accuracy—perhaps well below 50%. Most data mining methods are not well suited to problems with these characteristics. The genetic-based data mining algorithm developed to solve this data mining task is described in later in this chapter.

Interpretation of Results

The data mining algorithm produces a set of patterns for predicting component failures (i.e., each pattern x can be viewed as a rule of the form "$x \Rightarrow$ device failure"). Recall and precision values are computed for each pattern. For this domain, a pattern's recall is the percentage of the total component failures that it predicts and its precision is the percentage of times a prediction is correct. Note that if we were to view these patterns/rules as association rules, then recall corresponds to support and precision to confidence.

To help select the most appropriate subset of the generated patterns, our data mining software orders the patterns from most to least precise and then uses this ordering to generate a precision/recall curve. This curve is generated by viewing the most precise pattern as a solution, the two most precise patterns as another solution, etc., and then plotting the precision and recall for each solution. Such a precision/recall curve is shown later in Figure 1. Each point on the curve can then be evaluated given the cost of a component failure that is not predicted (false negative), the cost of incorrectly diagnosing a component as going to fail (false positive) and the value of correctly identifying a failure (true positive). If these values are all known, the optimal

point on the curve can be determined. For this domain, as for most domains in the real world, these costs are not precisely known. In this case, these values may be estimated. Alternatively, several of the solutions can be given to a domain expert, who could then choose the most attractive solution based on the precision and recall values.

The KDD process is an iterative process and it is common to use results as feedback into earlier stages in the KDD process. In this project the warning and monitoring times are parameters to the data mining software, and preliminary results were used as feedback to modify these parameter values. The values selected for evaluation were partly based on the "sensitivity" of the results to these parameters. For example, if a small increase in the monitoring time significantly improved our ability to correctly predict a failure, then we explored additional points in this neighborhood. This exploration also provided insight into the nature of the prediction problem. For example, as is shown later, decreasing the warning time enables much better predictions to be made. This indicates that there is information useful for predicting failures that occurs shortly before the actual failure.

Consolidating Knowledge

The final step involves consolidating the discovered knowledge. This was accomplished by documenting and distributing the key results. More importantly, these results were then to be incorporated into the ANSWER expert system in the form of rules. Changes were proposed but were not carried out because of a decision, unrelated to this effort, to dramatically change the way the 4ESS switches were monitored.

The Data Mining Algorithm

The data mining task requires an algorithm that can identify predictive sequential and temporal patterns in the network alarm data. Because existing methods and software packages were not suitable for performing this task, we developed the Timeweaver data mining software package (Weiss 1999). Timeweaver is a genetic-based data mining system that evolves populations of prediction patterns in order to solve event prediction problems. The decision to utilize a genetic algorithm was based on several factors. One important consideration was that most existing methods are not readily applicable to our data mining task. For example, rule and decision tree learners are not suited to the task because they operate on classified examples, whereas our task involves streams of events. The requirement to find patterns *between* events caused us to consider methods that could search directly in the space of possible patterns. Given the nature of the data mining task and the requirements on the pattern language, the space of possible patterns is enormous. Given the difficulty of the prediction task and our belief that it is the relationship between alarms that is critical in identifying faults, we felt that a greedy algorithm would not sufficiently explore the search space. For these reasons a genetic algorithm was chosen—a method that is widely known for its ability to efficiently search through large search spaces.

The first issue in designing a genetic algorithm is how to encode each individual in the population. In this case, each individual represents a pattern. Patterns in Timeweaver are represented as a sequence of pattern-events, where each pattern-event roughly corresponds to an event in the data set (i.e., an alarm in the context of this project). Patterns also have the following extensions:

- the feature values in each pattern-event may take on a wildcard value,
- ordering constraints are specified between successive pattern-events in the pattern, and
- a pattern duration is associated with each pattern

A pattern matches a sequence of events if: 1) each pattern-event in the pattern matches an event in the sequence, 2) the ordering constraints specified in the pattern are obeyed, and 3) all events

involved in the match occur within a time period that does not exceed the pattern duration. For this telecommunication domain, each pattern-event is of the form <device-id, device-type, diag-code, severity>. A sample pattern, generated by Timeweaver, is: "351:<id20, TMSP,?, Major> * <id20, TMSP, ?, Major> *<id20, TMSP, ?, Minor>". This pattern is matched if, on the same TMSP device, two major alarms occur, followed by a minor alarm, all within 351 seconds. The "?" represents a wildcard, indicating that the diagnostic code does not matter, and the "*" represents the *after* constraint, which specifies the relative ordering of the events. Note that the presence of other alarms in the alarm stream will not prevent this pattern from matching.

New patterns are generated by combining existing patterns using a crossover operator or by modifying existing patterns via a mutation operator. The patterns selected for combination and mutation are selected proportional to their fitness. The fitness of the prediction pattern is based on both its precision and recall and is computed using the F-measure, defined in equation 1, where • controls the importance of precision relative to recall.

$$\text{Fitness} = \frac{(\beta^2 + 1)\,\text{precision} \cdot \text{recall}}{\beta^2 \text{precision} + \text{recall}} \qquad \textbf{(1)}$$

Any fixed value of • was found to lead to poor performance of the genetic algorithm, so the value of • is continuously varied between 0 and 1. This allows "specific" patterns, with high precision and low recall, to evolve, as well as "general" patterns, with high recall but low precision. Over time, patterns evolve that tend to do well with respect to both measures, although some of the general and specific patterns will continue to remain in the population. By employing both precision and recall in the fitness function the genetic algorithm is able to find patterns that perform well at predicting rare events. Had we only used precision in the fitness function, then the genetic algorithm would have quickly converged to a few extremely specific patterns, and most of the search space would have remained unexplored. The details of the genetic algorithm are described in Weiss (1999). The interested reader may find a general introduction to genetic algorithms in Mitchell (1998) and Goldberg (1989).

Results

The target dataset was split into disjoint training and test sets by placing all alarms associated with 70% of the 4ESS switches into the training set and the remaining alarms into the test set. Timeweaver was applied to the training set and the "mined" patterns were then evaluated on the test set. Figure 1 displays the precision/recall curves generated when the monitoring time is held constant at eight hours while the warning time is varied. As expected, the precision of the predictions decreases as the recall of the predictions increases. The figure demonstrates that the performance of the rules is heavily affected by the warning time parameter—the shorter the warning time the better the predictions. This indicates that the alarms that occur near the actual failure are especially useful in predicting the failure. The mined patterns are not listed since they are only meaningful to a domain expert. However, each curve was formed from approximately thirty patterns, most of which contain 3-5 pattern-events.

International Journal of Engineering, Management, Humanities and Social Sciences Paradigms (IJEMHS)
Volume 30, Issue 04, Quarter 04 (Oct-Nov-Dec 2018)
ISSN (Online): 2347-601X
www.ijemhs.com

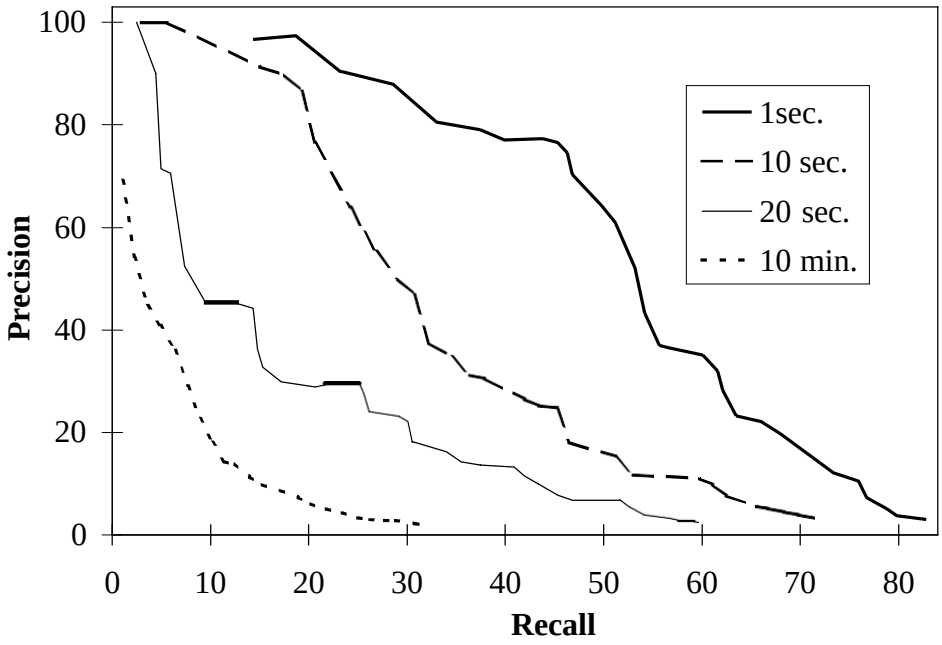

Figure 1: Predictive Performance of Prediction Patterns

Alternative Approaches and Related Work

Other approaches toward solving the data mining task were considered. Most of these approaches involve reformulating the event prediction problem into a classification problem by transforming the event sequence data into classified examples (Dietterich and Michalski 1985). These approaches typically involve sliding a window over the data. Events that fall within the window are encoded as a single, fixed-length example. The example is classified based on the proximity of the events in the window to the event to be predicted. A drawback of this approach is that some sequence and temporal information is inevitably lost in the transformation process—and a-priori we do not know what information is important to preserve. However, one advantage of this approach is that a wealth of existing classification methods can be applied once the transformation has been performed. This approach was evaluated on the telecommunication prediction problem. The network alarm data was transformed into classified examples using a sliding window and the resulting examples were then fed into C4.5 (Quinlan 1993) and RIPPER (Cohen 1995), two popular classification systems. The learned rules were not competitive with those generated by Timeweaver (Weiss and Hirsh 1998), most likely because of the limited amount of sequential and temporal information that was preserved in the transformation process. A similar transformation-based approach, but with a more sophisticated encoding scheme, was employed by Weiss, Eddy and Weiss (1998) to predict catastrophic 4ESS failures and by Sasisekharan, Sesahdri and Weiss (1996) to identify chronic network problems.

A very different data mining approach has also been used to analyze network alarm data. The Telecommunication Network Alarm Sequence Analyzer (TASA) uses specialized data mining algorithms to formulate rules that describe frequently occurring patterns in sequences of network alarms (Hätönen, Klemettinen, Mannila, Ronkainen, and Toivonen 1996). These rules have been used to filter redundant alarms, locate network problems and predict network faults. However, the *common* patterns identified by TASA will not necessarily be useful for prediction. In fact, for our specific data mining task, where the event to be predicted is rare, it is extremely unlikely that common patterns in the alarm data would be helpful in predicting the component failures.

References

Cohen, W. 1993. "Fast effective rule induction." In *Proceedings of the Twelfth International Conference on Machine Learning (Lake Tahoe, Nevada)*, edited by A. Prieditis, S. Russell, pp. 115-123, Menlo Park, Calif.: AAAI Press.

Dietterich, T, and Michalski, R. 1985. "Discovering patterns in sequences of events." Artificial Intelligence **25**:187-232.

Goldberg, D. E. 1989. *Genetic algorithms in search, optimization, and machine earning*. Reading, Massachusetts: Addison-Wesley.

Hätönen, K., Klemettinen, M., Mannila, H., Ronkainen, P., and Toivonen, H. 1996. "Knowledge discovery from telecommunication network alarm databases." In *Proceedings of the Twelfth International Conference on Data Engineering (New Orleans, Louisiana)*, pp. 115-122, IEEE Computer Society Press.

Mitchell, M. 1996. *An introduction to genetic algorithms*. Cambridge, Massachusetts: MIT Press.

Quinlan. J. R. 1993. *C4.5: Programs for machine learning*. San Mateo, Calif.: Morgan Kaufmann.

Sasisekharan, R., Seshadri, V, and Weiss, S. 1996. "Data mining and forecasting in large-scale telecommunication networks." IEEE Expert 11(1): 37-43.

Weiss, G. M., Eddy, J. and Weiss, S. 1998. "Intelligent telecommunication technologies." In *Knowledge-based Intelligent Techniques*, edited by L. C. Jain, R. D. Johnson, Y. Takefuji, and L. A. Zadeh, pp. 249-275. Boca Raton, Florida: CRC Press.

Weiss, G. M., Ros, J. P., and Singhal, A. 1998. "ANSWER: network monitoring using object-oriented rules." *In Proceedings of the Tenth Conference on Innovative Applications of Artificial Intelligence (Madison, Wisconsin)*, pp. 1087-1093. Menlo Park, Calif.: AAAI Press.

Weiss, G. M. 1999. "Timeweaver: a genetic algorithm for identifying predictive patterns in sequences of events." *In Proceedings of the Genetic and Evolutionary Computation Conference (Orlando, Florida)*, edited by W. Banzhaf, J. Daida, A. Eiben, M. Garzon, V. Honavar, M. Jakiela, pp. 719-725. San Francisco, Calif.: Morgan Kaufmann.

Weiss, G. M. and Hirsh, H. 1998. "Learning to predict rare events in event sequences." In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (New York, New York)*, edited by R. Agrawal, P. Stolorz and G. Piatetsky-Shapiro, pp. 359-363. Menlo Park, Calif.: AAAI Press.