Event Discovery in Medical Time-Series Data

Christine L. Tsien, PhD
Massachusetts Institute of Technology, Laboratory for Computer Science, Cambridge, MA
Harvard Medical School, Boston, MA

*Vast amounts of clinical information are generated daily on patients in the health care setting. Increasingly, this information is collected and stored for its potential utility in advancing health care. Knowledge-based systems, for example, might be able to apply rules to the collected data to determine whether a patient has a certain condition. Often, however, the underlying knowledge needed to write such rules is not well understood. How could these clinical data be useful then? Use of machine learning is one answer. We present a pipeline for discovering the knowledge needed for event detection in medical time-series data. We demonstrate how this process can be applied in the development of intelligent patient monitoring for the intensive care unit (ICU). Specifically, we develop a system for detecting Ôtrue alarmÕ situations in the ICU, wherecurrently as many as 86% of bedside monitor alarms are fals*__e.__

INTRODUCTION

As information technology continues to expand into all areas of health care, we need to understand how to take advantage of the clinical information being made available. Vast amounts of clinical data are being generated and collected daily on patients in the health care setting. These data, however, can only be as helpful as we know how to use them. Knowledge-based systems, for example, might be able to apply rules to data to determine whether a patient has a certain condition. To build knowledge-based systems, however, assumes that we both understand the underlying knowledge and know how to encode that knowledge into usable rules. Often, though, the underlying knowledge is not well understood. In those cases, is there a way we can still take advantage of the available clinical information?

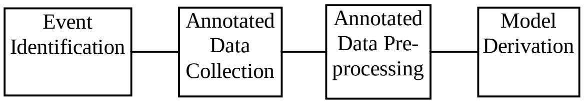
Machine learning methods, such as neural networks and decision tree classifiers, are being used increasingly for knowledge discovery in other areas of society. Examples of their application are in loan advising, speech recognition, and robot vision.[1] In medicine as well, machine learning methods have been explored. A common target area for these methods is the classification of patients as having or not having a disease condition (e.g., myocardial infarction) based upon patient characteristics (e.g., age, gender, smoking history, and symptoms). An area of medicine that has not received as much attention for machine learning is data-intensive bedside monitoring. Patients in the operating room, intensive care unit (ICU), emergency room, labor and delivery department, coronary care unit, as well as other areas of the health care setting, are usually connected to several lines, tubes, and probes that continuously monitor vital signs such as heart rate, blood pressure, and respiratory rate. While the classification of a patient as having a myocardial infarction or not is relatively easy to fit into the framework of machine learning, it is less clear how to formulate these bedside monitoring situations as machine learning questions.

We present a process, or pipeline, that can be used for knowledge discovery of events in medical time-series data. Events of interest can range from high-level clinical events, such as a patientÕs development of low blood pressure, to low-level events, such as sensor artifact. We then demonstrate use of this pipeline for development of ÔintelligentÕ patient monitoring. Specifically, we develop a system for detecting Ôtrue alarmÕ situations in the ICU, where currently as many as 86% of bedside monitor alarms are false alarms.[2,3]

EVENT DISCOVERY PIPELINE

Four fundamental parts comprise the pipeline for event discovery in medical time-series data. These are: identification of the event(s) of interest, annotated data collection, annotated data preprocessing, and derivation of an event detection model. This system is depicted in Figure 1. Performance evaluation and prospective verification are also necessary.

Figure 1. Components of the pipeline for event discovery in medical time-series data.



Event Identification

The first step in the event discovery pipeline is identification of the event or events of interest for knowledge discovery. An event in this context should be an entity that is thought to effect changes in available monitored time-series values; the exact nature of those changes is what we would like to better understand. Examples of events include disconnection of an electrocardiogram (ECG) lead (or other sensor), apnea (lack of breathing), false alarm due to patient motion, and blood pressure decrease warranting clinical attention. Candidate events for knowledge discovery should either occur frequently, or, if not, occur in environments amenable to prolonged monitoring and observation such that adequate numbers of those events may eventually be observed. Candidate events should also be such that it is clear to an observer when they are or are not occurring.

Annotated Data Collection

The second step is to collect a large amount of numerical time-series data along with annotations of event occurrences and event Ônon-occurrencesÕ. Time-series data can usually be stored to computer disk either in a central data repository or via a laptop computer. Annotations need to be made prospectively, at the time of event occurrence. Retrospective chart review, for example, is not adequate for these purposes. Annotations furthermore need to be Ôtime-stampedÕ for accurate correlation with the data, which are usually in a separate file or files. One way to meet these criteria is by using a custom-built program in which a human observer can easily record time periods of event occurrence and event non-occurrence. An observer, if not already knowledgeable about the area, can be trained to recognize which events to look for (with verbal verification from medical staff). An alternative to a custom-designed annotation program is a custom-formatted data entry interface for a readily available commercial spreadsheet program.

Data collection and annotation can proceed for as long as is feasible to capture multiple occurrences of the event of interest. Typically, machine learning programs are more robust when presented with more samples of the event of interest. Initial model development can also be tried periodically, with return to data collection if inadequate models (due to insufficient event samples) result.

Annotated Data Preprocessing

Preprocessing is much more important than generally recognized. It enables us to apply traditional machine learning methods to less traditional application areas such as medical monitoring. The two major components of the preprocessing step are feature attribute derivation and class labeling. Feature attribute derivation refers to the selection and calculation of mathematical quantities, such as moving mean or median, which can describe the time-series data and which are thought to be potentially different for events versus non-events. The quantities are calculated over a specified time interval (e.g., 10 seconds). The same quantities can also be calculated over multiple time intervals (e.g., 10 seconds, 1 minute) and then used as two different data attributes. The time intervals may be chosen to reflect a very general understanding of the problem. For example, very short time intervals might be chosen for spurious false alarms. The derived values are calculated not only for just one physiological signal type, but also for all available data signals being collected. The various quantities calculated for each signal, for all monitored signals of interest, together comprise the set of feature attributes that describe a time period of bedside monitoring.

Each set of multi-signal feature attributes is then given a class label of ÔeventÕ or Ônon-eventÕ according to the recorded annotations. All collected data are similarly transformed into sets of class-labeled feature attributes. Time intervals spanning a transition from an event to a non-event or vice versa can electively be disregarded for initial model development experiments. Alternatively, these transition periods can themselves become the event of interest for detection.

Model Derivation

Class-labeled sets of feature attributes are then divided into two or three sets: a training set, a test set, and an optional evaluation set. The training set is used for deriving candidate event detection models. The evaluation set is used to determine how well candidate models perform relative to each other. Once a final model is selected, it is then run on the reserved test set to determine the modelÕs performance. A training set consisting of approximately 70% of the available data is often chosen, while the remaining data can be further split to create the other two data sets.

For the techniques described thus far, ÔsupervisedÕ machine learning methods, such as neural networks or decision trees, can now easily be employed. These machine learning methods facilitate development of models from training data, which can then be used to classify unseen data as events or non-events.

Model performance is evaluated by comparing the areas under the receiver operating characteristic (ROC) curves4 for different models. The ROC curve is a plot of sensitivity versus one minus specificity, where sensitivity measures the number of correct model-labeled event cases out of the total number of actual event cases, while specificity measures the number of correct model-labeled non-event cases out of the total number of actual non-event cases. Because sensitivity and specificity can be inversely varied simply by altering the threshold at which to categorize a case as one class or the other, the area under the ROC curve more effectively describes a modelÕs discriminatory ability. Final models should additionally be evaluated prospectively in the clinical setting to better assess actual performance in detecting events of interest.

APPLICATION TO ICU MONITORING

We now demonstrate how the described event discovery process can be used for ICU monitoring to decrease false alarms. Previous studies have shown that as many as 86% of alarm soundings in the ICU are actually false.2,3 Current systems for monitoring vital signs typically sound an alarm any time the monitored signal surpasses a high threshold limit or falls below a low threshold limit. This simplistic rule, however, usually results in a large number of spurious readings that cause false alarms. This can lead to several problems,5,6 the most important end result being compromised patient care. Knowledge-based7,8 and other approaches have been proposed for improving various aspects of patient monitoring,9,10 but none has seen widespread clinical application.

Our approach is to develop multi-signal, machine-learned models able to detect Ôtrue alarmÕ events from bedside time-series data. As an example, we choose our event of interest to be true alarms that are clinically relevant (of any cause) occurring in the ICU on the systolic blood pressure signal.

Methods

Having identified an event of interest (true alarms on the arterial lineÕs systolic blood pressure signal), the next step was to collect annotated data. Over the course of 12 weeks, bedside monitor data along with prospectively recorded annotations of event and non-event occurrences were recorded in the multidisciplinary ICU (MICU) of a pediatric hospital. Monitoring devices for each patient were connected to a SpaceLabs bedside monitor (SpaceLabs Medical, Redmond, WA). A laptop computer placed at the bedside recorded raw values transmitted via a serial line from the SpaceLabs monitor approximately every five seconds. Available raw values included ECG heart rate, ECG respiratory rate (measured by impedance pneumography), pulse oximeter oxygen saturation, and arterial line mean and systolic blood pressure. A trained human observer recorded annotations into a custom-designed data entry interface to an Access database program (Microsoft, Redmond, WA) running on the laptop. For each occurrence of a clinically relevant systolic blood pressure true alarm, the trained observer created a time-stamped note indicating the true alarm occurrence. False alarm soundings, as well as periods of appropriate alarm silence (Ôtrue negative alarmsÕ), were also recorded. The bedside nurse moreover verbally verified each annotation.

Preprocessing first involved calculation of eight different mathematical quantities for each successively overlapping group of raw data values. These calculated quantities included moving mean, median, maximum value, minimum value, range, linear regression slope, absolute value of linear regression slope, and standard deviation. These eight quantities were furthermore calculated for each of three different time intervals. The time intervals chosen were 10, 20, and 45 seconds, corresponding to feature derivation over two, four, and nine raw values, respectively. These time intervals were chosen with the general knowledge that false alarms tend to occur fleetingly, while true alarms tend to develop more slowly; the exact numbers themselves were otherwise chosen arbitrarily. The 24 described values (8 different quantities for each of 3 different time intervals) were calculated for each of the five recorded data signals, resulting in sets of 120 feature attributes (120-dimensional feature vectors). Each multi-signal feature vector was next labeled according to the annotations. Feature vectors whose attributes were derived from raw values labeled Ôtrue alarmÕ were given the true alarm class label. Feature vectors whose attributes were derived from raw values occurring during false alarm or true negative (no alarm) periods were labeled Ôno alarmÕ (meaning that the desired result was to have no alarm sound at those times). Feature vectors whose attributes were derived from raw values spanning more than one label type were not used in model derivation for this set of experiments.

Data in the form of labeled feature vectors were divided as follows: 70% for training set, 21% for test set, and 9% for evaluation set. The training data were then given to both a decision tree induction system (c4.5)11 and a neural network classifier system (LNKnet) (Lincoln Laboratory, Lexington, MA). The decision tree system allows for model experimentation in various ways, such as changing the ÔselectivityÕ of growing a tree or the amount of ÔpruningÕ of a tree. Decision tree models were preferred if they had fewer errors when run on the evaluation set, and/or smaller size with little to no increase in the number of errors when run on the evaluation set. No special decision tree features (e.g., boosting, bagging) were used. The neural network system allows for model variation also, for example, by changing the number of layers of hidden nodes to be included in the network structure, or by changing the number of hidden nodes per layer. All networks explored here used a back propagation algorithm to perform a gradient descent that minimizes the error seen at the outputs. Networks with simpler structure and fewer hidden nodes, having similar performance on the evaluation set compared to more complicated networks, were preferred. Networks had two output nodes, one for each class (events and non-events); the one with the maximum output was returned. Final tree and network models were run on the same test set. For decision trees, ROC curves were determined by first assigning to each tree leaf the probability of being an event for a set of derived values that percolates to that point. These probabilities are based upon the ratio of events to (events + non-events) that fall into each leaf during training. The threshold for considering a case to be event or non-event was then set at each leaf probability value. The resulting sensitivity-specificity pairs were used to plot corresponding ROC curves, from which the area under the curves could then be calculated by trapezoidal method. For neural networks, ROC curve areas were calculated internally by LNKnet. A threshold is similarly moved over the event class output, with patterns below the threshold being rejected and patterns above the threshold being labeled as events.

Results

Over the 12-week data collection period, approximately 585 hours of bedside signal values were recorded along with annotations of alarm and no-alarm periods. Only monitored data containing all five signals of interest (heart rate, oxygen saturation, respiratory rate, and mean and systolic blood pressure) were further used in this study. Data were preprocessed by the described methodology. There were 86,062 training cases, 25,952 test cases, and 10,906 evaluation cases, collectively consisting of 1550 true-alarm cases and 121,350 no-alarm cases. (The no-alarm cases included 2109 false-alarm cases.) The training and evaluation sets were then given to c4.5 and LNKnet.

The final decision tree model chosen for detection of true alarms on the systolic blood pressure signal is shown in Figure 2. Class labels are represented by Ô1Õ for the true-alarm class and Ô0Õ for the no-alarm class. Parentheses after a class label indicate the number of training data cases which arrived at that node, followed by the number of training cases which were incorrectly classified at that node. Attribute names are a concatenation of the abbreviation of the signal name, an abbreviation of the derived feature name, and the number of values over which the derived feature was calculated. For example, the first line in the decision tree model, Òsbp_avg9 <= 136.9 : 0 (71141.0/137.8),Ó means: Òif the average value over nine raw values of systolic blood pressure is less than or equal to 136.9, the case will be labeled no-alarm. During training, 71141.0 training cases arrived at this node and were labeled no-alarm; 137.8 of those cases were incorrectly labeled.Ó (Fractional numbers of cases can arise due to pruning of the tree.) The final model used a ÔselectivityÕ of 2% (meaning very selective about whether or not to add a particular test node to the tree), and a pruning factor of 45 (meaning that a test node on the tree was only kept if at least 45 cases were classified by one outcome branch of that node. The decision tree model achieved an area under the ROC curve of 94.34% when run on its test set.

The final neural network model for systolic blood pressure alarm detection contained 120 input nodes, one hidden layer with 15 nodes, and two output nodes (one for each class type). During network training, a step size of 0.2 was chosen for updating network weights during error propagation. The training process updated weights during each of 20 cycles. The final neural network achieved an ROC curve area of 98.98% when run on its test set.

DISCUSSION

The results of applying the described event discovery pipeline to the problem of detecting true alarms in the ICU are promising. Both the decision tree and neural network models performed well on their test sets. These results still need to be validated prospectively in the clinical setting. The actual thresholds present in the decision tree model, for example, may reflect those patients whose vital signs were used for training. These thresholds would likely need to be refined for each different population of patients (e.g., neonatal babies, young children, or adults).

The ICU case study is not without limitations. First, the data collected were only available at a frequency of once per five seconds. This limits our choice in selecting appropriate time intervals for feature derivation; for example, a feature attribute derived from a time interval that is not a multiple of five seconds may in fact be the most accurate predictor for an event. The infrequency of data values from which to learn alarm patterns also may pose a problem if higher frequencies of data are later available when these models are tested prospectively.

Another limitation of the case study was that annotations were only recorded to the nearest minute, while raw data values were collected every five seconds. This difference mandates that we will incorrectly label some cases simply because we are not sure precisely when, during the recorded minute, the true alarm actually occurred. Future work in this area should pay caution to recording annotations with the same time granularity as available raw data.

Annotations are additionally subject to inter-observer and intra-observer biases. For the described ICU case study, two trained observers recorded all of the annotations; for each annotation, the bedside nurse present that day validated the annotation. It is not possible that all of the MICU nurses and both trained observers interpreted or recorded all alarm occurrences in the same manner. Moreover, the same nurse or the same trained observer would also likely not record every alarm occurrence in the same manner.

Despite its limitations, however, the ICU alarm example has provided a useful demonstration of how data-intensive medical time-series data may be useful, even when the underlying knowledge about how they relate to particular events is not well understood. Especially at a time when information technology is making available enormous amounts of clinical data, methods for taking advantage of these data need to be explored. The event discovery paradigm may be one technique that can assist in learning from these data.

Acknowledgments

The author would like to thank I. Kohane, J. Fackler, P. Szolovits, and H. Fraser for advice, J. Doyle for support; and M. Curley and the ChildrenÕs Hospital MICU nurses in Boston for assistance. This work was supported in part by the National Library of Medicine, AAUW Educational Foundation, and DARPA contract F30602-99-1-0509.

References

1. Mitchell TM. Machine learning. McGraw-Hill, 1997.
2. Lawless ST. Crying wolf: false alarms in a pediatric intensive care unit. Crit Care Med 1994; 22:981-985.
3. Tsien CL, Fackler JC. An annotated data collection system to support intelligent analysis of intensive care unit data. In: Advances in intelligent data analysis. Springer-Verlag, 1997, pp. 111-121.
4. Hanley JA, McNeil BJ. The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 1982; 143:29-36.
5. Meredith C, Edworthy J. Are there too many alarms in the intensive care unit? An overview of the problems. J Advanced Nursing 1995; 21:15-20.
6. Sara CA and Wark HJ. Disconnection: an appraisal. Anesthesia and Int Care 1986; 14:448-452
7. Fukui Y, Masazawa T. Knowledge-based approach to intelligent alarms. J Clin Monit 1989; 5:211-216.
8. Koski EMJ, Sukuvaara T, Makivirta A et al. A knowledge-based alarm system for monitoring cardiac operated patientsÑassessment of clinical performance. Int J Clin Mon Comp 1994; 11:79-83.
9. Uckun S. Intelligent systems in patient monitoring and therapy management. Int J Clin Mon Comp 1994; 11:241-253.
10. Orr JA, Westenskow DR. A breathing circuit alarm system based on neural networks. J Clin Monit 1994; 10:101-109.
11. Quinlan JR. C4.5 Programs for machine learning. San Mateo, Morgan Kaufman Publishers, 1993.