Activity Monitoring: Noticing interesting changes in behavior

Tom Fawcett
Bell Atlantic Science and Technology
500 Westchester Ave
White Plains, New York 10604
tfawcett@acm.org

Foster Provost
Bell Atlantic Science and Technology
500 Westchester Ave
White Plains, New York 10604
provost@acm.org

Abstract

We introduce a problem class which we term *activity monitoring*. Such problems involve monitoring the behavior of a large population of entities for interesting events requiring action. We present a framework within which each of the individual problems has a natural expression, as well as a methodology for evaluating performance of activity monitoring techniques. We show that two superficially different tasks, news story monitoring and intrusion detection, can be expressed naturally within the framework, and show that key differences in solution methods can be compared.

1 Introduction

In this paper we introduce a problem class which we term *activity monitoring*. Such problems typically involve monitoring the behavior of a large population of entities for interesting events requiring action. Examples include the tasks of fraud detection, computer intrusion detection, network performance monitoring, crisis monitoring, some forms of fault detection, and news story monitoring. These applications may differ greatly in representation. For example, the data streams being monitored may be streams of numbers, mixed-type feature vectors, or free-text documents. However, problems in this class share significant characteristics that differentiate them from other KDD problems.

The goal of activity monitoring is to issue alarms accurately and in a timely fashion. Standard KDD techniques such as classification, regression and time series analysis are useful as solution components, but these techniques do not completely address the goal. Although activity monitoring applications have received much attention individually, to our knowledge they have never been generalized and analyzed in a common framework. We present a framework within which these tasks have a natural expression. This framework codifies similarities of the tasks and highlights significant differences.

We begin by discussing briefly cellular phone fraud detection as a domain to illustrate some of the issues in activity monitoring. We define the problem formally and present an evaluation methodology. Then we explain some important differences between types of activity monitoring tasks. The resulting framework comprises the problem definition, the evaluation methodology, and the categorization of activity monitoring techniques. We believe this framework is a significant contribution because it helps to understand activity monitoring better in general. To demonstrate this, we show that it aids in understanding how methods from fraud detection can apply directly to the seemingly different problem of creating stock market alerts based on news stories, yet the same methods do not seem to work well on the problem of computer intrusion detection, which seems closely related to fraud detection.

2 Cellular phone fraud detection

Cellular phone fraud detection is a typical activity monitoring problem. The task is to scan a large set of accounts, examining the calling behavior of each, and to issue an alarm when an account appears to have been defrauded. This can be done in many ways, such as profiling users' behavior, looking for known fraud patterns, etc. The goal is to identify fraudulent activity as soon as possible without creating too many false alarms [7].

Calling activity may be represented in various ways, but is usually described with call records. Each call record is a vector of features, both continuous (*e.g.*, `CALL_DURATION`) and discrete (*e.g.*, `CALLING_CITY`). However, there is no inherent primitive representation in this domain. Calls can be aggregated by time, for example into call-hours or call-days. Though it is not obvious, they can also be broken up into a

finer grainsize: more detailed intra-call information is available, such as the delays between button presses, switch hand-offs during each call, etc.

The task is to monitor all accounts continuously and to identify fraudulent activity as soon as possible. In one intervention scenario, when an alarm is issued, a human fraud analyst must check the account carefully and decide what to do. Therefore, it is important that the system not create too many false alarms. The precise definitions of "as soon as possible" and "too many false alarms" vary with the amount of fraud, the size of the workforce, and other factors.

In evaluating solutions to this problem, several issues arise:

- **Granularity:** Because of the different possible problem representations, it is difficult to evaluate and compare different solutions. A method that classifies individual calls cannot be compared easily to one that classifies account-days or individual segments of calls. Both the false alarm rate and the true alarm rate must be normalized before solutions can be compared.
- **Multiple alarms:** Alarms do not have equal value. For example, a monitor might generate several alarms in a row for a given defrauded account. The first alarm is important, but subsequent alarms occurring closely thereafter contribute virtually nothing. Evaluation should take into account the diminished importance of multiple alarms.
- **Benefit of timely alarms:** Fraud should be detected as soon as possible and evaluation should reflect this. The benefit of an earlier alarm can often be quantified. In cellular phone fraud detection, the cost of delaying an alarm may be measured in real time, the number of fraudulent calls missed, the amount of uncompensated airtime or the total expense incurred by the fraud.

These characteristics are common among activity monitoring domains. Perhaps surprisingly, most prior work in such domains ignores one or more of these issues, even though they are exhibited in the corresponding application. Section 5 argues that standard evaluation metrics such as classification accuracy, cost, and even ROC analysis are insufficient for evaluating activity monitoring systems. In the next section we introduce a formalism and a framework that can accommodate these complexities.

3 Activity Monitoring

We define the general problem as follows. Let $\mathcal{D}$ be a set of *data streams*, where each data stream $D_i \in \mathcal{D}$ is an ordered set of data items $\{d_{i,1}, \ldots, d_{i,k_i}\}$ (k_i need not equal k_j, for $i \neq j$—that is, the length of the episodes need not be equal). We will be liberal in what are considered data—numeric measurements, vectors of symbolic features, text, etc. Each data stream represents information about a single entity in the population being monitored, for example, customers, users, network equipment, nuclear reactor components, or even potential investment opportunities. For the rest of this paper we only refer to one data stream at a time, so for simplicity we omit the subscript i, and refer instead to a data stream D and its elements $\{d_j\}, j = 1, \ldots, k$.

The d_j's are information related to the entity being monitored, and comprise the evidence upon which alarms are to be issued. The data may be measurements taken directly from the entity, such as the temperature of an engine part at a point in time. They may be descriptions of the entity's behavior, such as transactions billed to an account. They may also be indirect descriptions of the behavior from a third party, such as news stories about a particular company. The d_j's are totally ordered in time, so $j > k \Leftrightarrow time(d_j) > time(d_k)$. The d_j's are not necessarily evenly spaced in time.

Activity monitoring is the task of analyzing the data streams in order to detect the occurrence of interesting behavior, which we refer to as *positive activity*. Episodes of positive activity need not be similar to each other, nor do is the non-positive activity in one data stream necessarily similar to that of any other. Let τ be a point in time denoting the onset of positive activity. For this paper, we assume that each D contains at most one τ, and that if there is a period of positive activity, it is at the end of D. Although real data streams may contain multiple periods of positive activity, we will see that this assumption is not overly restrictive. For the data stream D, τ designates the beginning of a contiguous subsequence $D_\tau = \langle d_{p_1}, \ldots, d_{p_m} \rangle$ such that $time(d_i) \geq \tau \Leftrightarrow d_i \in D_\tau$. Because the positive activity is necessarily at the end of D, $p_m = k$. Note that the positive activity is separate from the existence of the activity's evidence: D_τ may be the empty sequence. The goal of an activity monitor is to give an indication that the sequence is exhibiting positive activity; such an indication is called an *alarm*. Formally, an alarm α represents the point in time when it is issued.

Figure 1 illustrates a data stream with two alarms. A monitor may issue multiple alarms for any D. For completeness, a D that contains no positive activity implies $\tau = \infty$; if a monitor never alarms on a D, we consider $\alpha = \infty$.

3.1 Costs and benefits

There is a crucial difference between this problem formulation and those used in previous work. The goal

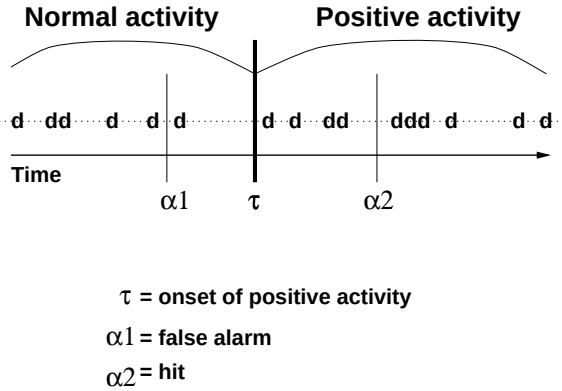


Figure 1: A data stream D with alarms α_1 and α_2.

in activity monitoring is not to identify completely all the positive activity, nor to classify each d_j as positive or negative. Rather, the goal is to identify *in a timely fashion* that positive activity has begun. Alarming earlier may be more beneficial, but after a first alarm, a second alarm on the same sequence may add no value.

To accommodate these considerations, we define two task-dependent functions: a score function s and a false alarm function f. Let $s_{\mathcal{D}}(\tau, \alpha, H, D)$ be a score function which returns the value (benefit) of an alarm α on a given subsequence, with respect to a given τ. The variable H is a sequence of the alarms that have already occurred on the sequence. We can now define the positive activity in terms of s: it is the subsequence of D for which $s_{\mathcal{D}}(\tau, time(d_j), \langle\rangle, D) > 0$.

A symmetric function $f_{\mathcal{D}}$ can be defined for false alarm penalties. Let $f_{\mathcal{D}}(\alpha, H, D)$ be the cost of a false alarm on sequence D, with H being the sequence of false alarms already issued.

We make several simplifications to these functions for presentation. Within a given domain we can eliminate the $\mathcal{D}$ subscript, although it is important to remember that the score function depends on the domain and on the particular sequence of activity. For this paper we will assume that for most domains the first alarm is significant but subsequent alarms add no value, so the score function returns a non-zero value only if H is the empty sequence. Thus we remove H from s and assume an implicit clause within s:

$$s(\tau, \alpha, H, D) = 0 \text{ if } H \neq \langle\rangle$$

Similarly, for this paper we assume that the cost of a false alarm is constant and does not depend on history, so f reduces to a variable representing this cost.

The goal of activity monitoring is two-fold. A technique should maximize the expected value of $s(\tau, \alpha, D)$ for any given alarm, while minimizing the number of false alarms. Adopting standard terminology, α is a *false alarm* if $\alpha < \tau$; α is a *hit* if $\alpha \geq \tau$ and $s(\tau, \alpha, D) > 0$. If $\tau < \infty$ and there does not exist an α such that $\tau \leq \alpha < \infty$ then the interesting period was *missed*.

It is important to understand that the concept of "true negative" is not well defined in activity monitoring. In practice, one can create definitions based on the particular representation used, but in principle there are infinitely many possible "true negatives," because of the continuity of time. This observation has been made before, for example in classification for visual pattern recognition [2], and we will return to it when we discuss the evaluation framework.

3.2 Cellular phone fraud detection revisited

The problem formulation may be clarified by an example from cellular phone fraud detection [7].[1] Each $D \in \mathcal{D}$ represents a customer account, comprising a sequence of cellular phone calls (the d_j). Each d_j is a time-stamped vector of features representing details of the call. "Positive activity" is cellular cloning fraud.

In this domain there are several realistic ways to define a score function $s(\tau, \alpha, D)$ to measure the benefit of alarming at α. One possibility is to count the number of fraudulent calls that would have been made had the fraud not been caught. These are the calls after α:

$$s(\tau, \alpha, D) = \left| \{d_i \in D \mid callstart(d_i) \geq \alpha\} \right|$$

A more elaborate score function calculates the cost of the airtime of these fraudulent calls:

$$s(\tau, \alpha, D) = aircost \times \sum_{d_i \in \{c \in D \mid callstart(c) > \alpha\}} airtime(d_i)$$

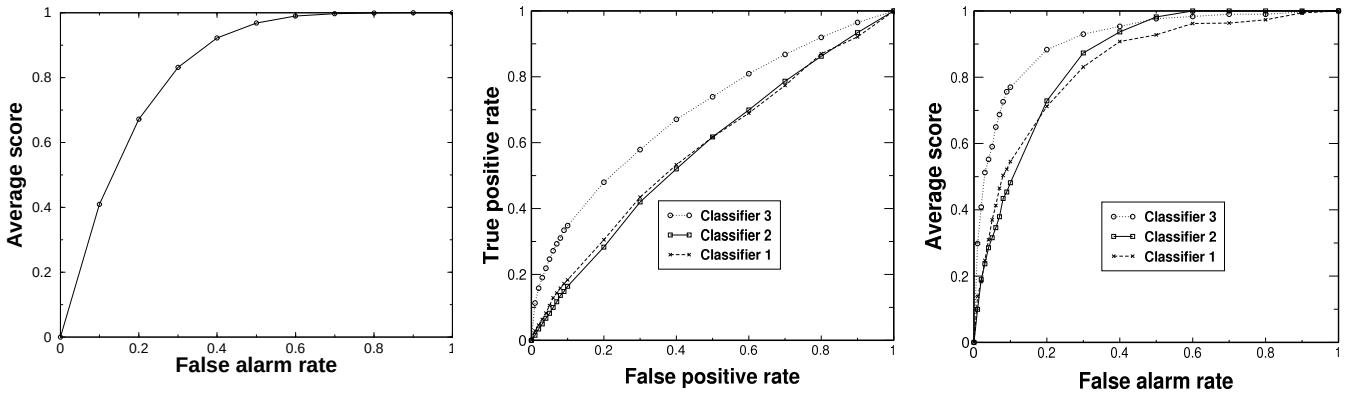
where *callstart* is the start time of a given call, and *aircost* is the (constant) cost of a unit of call airtime. Our previous work assumed a monetary cost of \$.40 per minute for airtime, and a fixed cost of \$5 for a false alarm, so $aircost = 0.4$ and $f = 5$;

Our assumption of at most one τ per episode is not problematic in practice. If a customer account is defrauded twice, the sequence can be decomposed into two single-τ sequences without affecting the evaluation. More generally, as long as the value of s depends on only one τ, the subsequence can be decomposed into separate episodes for the purpose of evaluation. There may be cases where s would combine value from multiple τ's, but we are aware of no non-contrived examples.

3.3 Evaluating activity monitoring performance

We have argued elsewhere [12, 13] that Receiver Operating Characteristic (ROC) analysis, a method

[1] We did not use this framework in our prior work, which we criticize later along with other prior work.



(a) (b) (c)

Figure 2: (a) AMOC curve of a random classifier, (b) ROC curves of three classifiers, (c) AMOC curves of the same three classifiers.

from signal detection theory, is the appropriate metric for analyzing classifiers under imprecision. ROC analysis depicts the tradeoff between true positive classifications and false positive ones, which is similar to the goal of activity monitoring. For activity monitoring we use ROC analysis with two minor modifications, which for this paper we call AMOC (Activity Monitor Operating Characteristic).

On the X-axis, ROC curves plot the false-positive rate, meaning here the percentage of the negative examples classified as positive. Since we have no fixed notion of a "negative example," we define the false-alarm rate to be the number of false alarms, normalized by the distance metric used in s, for example, the expected number of false alarms per unit time. This is analogous to the FROC technique found in visual pattern recognition [2], which also assumes no fixed notion of a negative example so false alarms per unit area are measured. Normalizing false-alarm rate allows us to compare methods that cannot be compared under standard ROC analysis. For example, in fraud detection a transaction classifier and an account-day classifier have different notions of negative example, so their standard false-positive rates cannot be compared directly. However, their normalized false-alarm rates can, because each produces a certain number of false alarms per hour. The second modification to ROC analysis is that on the Y-axis, rather than plotting the true positive rate, AMOCs plot the expected value of $s(\tau, \alpha, D)$.

AMOCs retain some of the advantages of ROC curves [12] but are tailored to activity monitoring. An example highlights the differences. Consider a fraud detection problem where fraud must be caught within five hours of its onset. The score function is binary, where the benefit of fraud detection is 1 only if detection is within five hours:

$$s(\tau, \alpha, D) = \begin{cases} 1 & \text{if } 0 \leq \alpha - \tau \leq 5 \\ 0 & \text{otherwise} \end{cases}$$

The false alarm rate is normalized per hour.

Figure 2a illustrates why even when simplified there is a fundamental difference between AMOC and ROC curves. The figure shows an AMOC curve for a hypothetical fraud detector. How good is the detection? In fact, this curve shows the performance of classifiers that alarm randomly with different frequencies (0.1 per hour, 0.2 per hour, etc.). Unlike ROC curves, for which the diagonal $y = x$ depicts the performance of random guessing, in AMOCs performance depends upon the definition of s. Even with a simple binary definition of s, an AMOC is not necessarily the same as an ROC.

Figures 2b and c show why the AMOC formulation is important for evaluating activity monitoring. The figures show ROC and AMOC curves for three classifiers on an activity monitoring domain. The ROC curves in b show that classifier 3 dominates the other two in ROC space [12], so its instance classification performance is generally superior. From this we might conclude that it is unconditionally best. However, the AMOC curves in c show that classifier 2 performs better for activity monitoring if maximal detection score is much more important than false alarm rate.

Figure 3 shows an algorithm for generating an AMOC curve from a probabilistic classifier. As with ROC curves, to generate AMOCs it is not necessary to specify thresholds on the classifier's continuous output. Given an assignment of probabilities to examples, the examples are sorted decreasing by their assigned probabilities. Then the sorted list is traversed, updating

```
Given: Alarms: Set of alarm tuples ⟨p, α, i⟩ where:
                p: probability calculated by the classifier
                α: alarm
                i: index of D_i
Output: R: Set of points on AMOC.

S = 0; /* Current cumulative score */
F = 0; /* False alarm rate */
R = {(0, 0)};
sort Alarms in decreasing order by p values;
for (⟨p, α, i⟩ ∈ Alarms) do
      if (α < τ_i) then /* α is a false alarm */
            F = F + f;
      else              /* α is in activity period */
            if (H_i is undefined) then /* First alarm */
                  S = S + s(τ_i, α, D);
                  H_i = α;
            else if (α < H_i) then
                  /* Earlier than previous first alarm */
                  S = S − s(τ_i, H_i, D);
                  S = S + s(τ_i, α, D);
                  H_i = α;
                  end if
            end if
      end if
      Add point (F, S) to R;
end for;
S_total = S;
F_total = F;
/* Scale F and S values to [0,1] */
for ((F,S) ∈ R) do
      F = F/F_total;
      S = S/S_total;
end for;
```

Figure 3: Algorithm for generating an AMOC curve from a set of alarms

the score and the false-alarm rate after each example. Each update produces a point on the AMOC curve; this point represents the performance of the classifier that would result from placing a threshold just below the corresponding probability. One list traversal generates points for all thresholds. Because we have made a simplifying assumption that any true alarms after the first have no contribution to the score, we need deal only with one other alarm (H_i) as history. This keeps the algorithm's time complexity to $O(n)$.

3.4 Key differences between methods

There are many similarities between activity monitoring problems. A framework for a problem class also should aid in understanding intraclass differences; for example, which differences are superficial and which are fundamental. We now focus on one distinction: the nature of the modeling and monitoring process. There are two high-level classes of methods:

- A *profiling* method builds a model of normal (non-positive) activity without reference to positive activity. A system can then use profiles to alarm on activity that deviates significantly from normal.
- A *discriminating* method builds a model of interesting (positive) activity, usually with reference to normal activity. This model is then used to scan for positive activity.

Similar distinctions have been made regarding methods for simple classification [14]. Here the distinction is based on a fundamental asymmetry of activity monitoring problems: normal activity is common and positive activity is rare. Therefore, methods other than supervised learning algorithms can play a significant role. For example, modeling normal activity with a regression curve or a probability distribution may be useful for judging typicality of subsequent activity.

An orthogonal distinction separates methods that consider activity to be *uniform* over all D_i from those that model each *individual* D_i. Adding this second distinction yields four distinct categories of methods:

1. *Uniform profiling* builds a profile of normal activity, assuming it is uniform across all D_i. For example, in a network surveillance task where only errors are reported, a general profile might be a stream of zeros, or a mean and standard deviation of errors over the population.

2. *Individual profiling* builds a profile of the normal activity of each D_i. For example, one type of intrusion detection creates a profile of each user's normal activity, and then looks for deviations from the profile.

3. *Uniform discriminating* builds a general model to distinguish positive from uniform normal activity. For example, creating investment alerts from company news bulletins requires forming a model of the type of news that indicates an interesting investment opportunity.

4. *Individual discriminating* builds a model to distinguish positive activity from the normal activity of each D_i. For example, examples of intrusions may be compared with each user's normal activity to build individualized scanning models.

We have considered the uniform/individual distinction only with respect to normal activity. For discriminating models, this distinction may be made with respect to the positive activity as well. For example, in monitoring device behaviors, particular devices may have idiosyncratic failure modes that must be modeled. However, such individualized positive activity is unusual; positive activity typically either is too scarce to be modeled individually, or simply is not related to any particular data stream.

For some tasks, discriminating models may be sufficient. In intrusion detection, *misuse detection* is a simple example: "defining intrusions ahead of time and watching for their occurrence" [9, p.26]. As another example, consider monitoring news stories for interesting investment opportunities. Noticing that today's news text differs from prior news text is not particularly promising, but a discriminating model of "useful investment news" could be quite valuable.

Are there cases where profiling would be more useful than discriminating? Detecting non-malicious intrusions is one example. For this task a model of positive activity may not exist, so discriminating may be impossible. An instance of profiling, *anomaly detection* "observes the activity of subjects and generates profiles for them that represent their behavior" [9, p.16]. Profiling is much more useful than discriminating in the burgeoning application of news story topic detection and tracking [1]. When positive activity is simply the introduction of new news topics, discriminating via modeling positive activity may be of little use.

Profiling and discriminating are two disjoint classes of techniques, but in practice they often are interwoven. For example, the DC-1 system [7] builds fraud detectors automatically by first using normal and positive activity to construct dimensions along which profiling will be effective. Then it creates profiling monitors that model each account's normal activity with respect to a single dimension. Finally it uses the outputs of the profiling monitors (indicating how far from normal the current activity is) as inputs to a discriminator that was trained to distinguish positive deviations from normal deviations.

DC-1 uses a particular type of discriminating method we call *change detection*, which is based on a model of transitions that occur from normal to positive activity in each individual account. To be useful, a change model must be able to refer to a profile of normal activity for each account. Change detection may be effective when there is little commonality within positive activity, but significant differences between normal and positive activity. For example, in fraud detection, there may be no times of day or locations that indicate fraud, but for a given account *changes* from its typical time of day or location usage may be very reliable indicators. Change detection capitalizes (in the modeling) on the temporal nature of activity monitoring problems.

Technically, uniform and individual modeling are two ends of a spectrum of groupings of data streams. In many applications, D_i's can be clustered into useful groups such that profiling or discriminating may be made more effective. For example, for phone-fraud detection, customers often can be grouped usefully as high-volume users, nine-to-five users, emergency-only users, etc. This paper does not address the problem of grouping data streams.

4 Empirical Demonstration

We have claimed that this framework allows activity monitoring tasks to be viewed as a single type of problem, and that the distinctions that arise help to understand this type of problem. We now provide a demonstration that the framework facilitates understanding of activity monitoring across different domains.

First we describe the task of creating investment-related news alerts, which is superficially very different from fraud detection, but similar as an activity monitoring task. Once representational issues are addressed, we can apply a system designed for fraud detection to this seemingly different problem. We then use the concepts of the framework to explain why this is the case.

Secondly, we consider a task of computer intrusion detection. Though it is superficially similar to fraud detection, the framework suggests that within the class they are fundamentally different. We show that, as would be predicted, the fraud detection system unmodified does not perform particularly well, but that a simple modification, suggested by the framework, performs better.

The goal here is not to advance the state of the art in either news monitoring or intrusion detection. Rather, these demonstrations serve to illustrate the value of viewing activity monitoring as a problem class.

4.1 An overview of the DC-1 system

Before discussing the two domains, we review briefly the DC-1 system which will be used in the experiments. DC-1 was designed previously to create detectors for cellular phone fraud by mining data on the behavior of cellular phone accounts. It is described in detail elsewhere [7]; only its salient aspects will be covered here, cast in the activity monitoring framework.

DC-1 constructs a fraud detector in three stages. First it generates rules that distinguish fraudulent calls from legitimate ones. This can be done using either uniform discriminating modeling (*e.g.*, learning a classifier from positive and negative calls) or change modeling (*e.g.*, learning rules to distinguish fraudulent changes in behavior within individual accounts). These rules are then used to create profiling monitors. Each monitor models the behavior of each account with respect to one rule and, when monitoring, describes how far the account's activity is from its typical behavior. Finally, DC-1 weights the monitor outputs to maximize the effectiveness of the resulting fraud detector. To do so, the outputs of the monitors are provided as features to a standard learning program along with the desired output (an account-day's correct class: fraud or non-fraud). The result is a weighted sum of the outputs of

said [it] expects	same period	revenues	increase over	per share
fourth	compare[d]	income	quarter	fiscal
earnings per	diluted fiscal	quarter ended	expenses	months ended
today reported	consensus	quarter earnings	year ended	repurchase
lower than	research [and] development	fourth-quarter	first call	Q[1234]
below analyst	for quarter	shortfall		

Table 1: Change indicators from news stories preceding stock spikes. Text in brackets was removed in lexical analysis.

the monitors, along with a threshold on the sum.

In use, the monitors view each day's calls from a given account, and each monitor generates a number indicating how unusual that account-day looks for the account. The numeric outputs from the monitors are treated as evidence and are combined by the detector. When the detector has enough evidence of fraudulent activity on an account, based on the indications of the monitors, it generates an alarm.

4.2 News Story Monitoring

The goal of this task is to scan news stories associated with a large number of companies and to issue alarms on specific companies when their stocks are about to exhibit positive activity. We collected stories and stock prices for approximately 6000 companies over a three month period. Each story was tagged with a list of companies to which it pertained; each company's stories constituted a different data stream, and an "interesting event" (positive activity) was defined to be a 10% change in the company's stock price (a price "spike") in either direction. The activity monitoring goal is to minimize the number of false alarms and to maximize the number of correctly predicted price spikes.

This domain seems very different from fraud detection. The data items are represented as free text rather than as predefined feature vectors of numeric and discrete attributes. The news stories are not as clearly associated with positive activity as are cellular phone calls, which are carefully tagged by a cellular company's billing system. Also, the alarms of this domain are fundamentally "opportunistic" rather than failure- or crisis-driven. A researcher attacking this problem would likely see it as very dissimilar to fraud detection, and would probably would seek an information retrieval solution, using precision and recall as evaluation metrics.

However, generating stock alerts is easily cast as an activity monitoring problem. Each news story constitutes a d_j of its associated company (D). We heuristically associate with a price spike any story appearing from midnight the day before the spike up until 10:30 AM of the day of the spike.[2] With the prior midnight as τ, the scoring function s for an alarm α can be defined as:

$$s(\tau, \alpha, D) = \begin{cases} 1 & \text{if } 0 \leq \alpha - \tau \leq 34.5 \text{ hours} \\ 0 & otherwise \end{cases}$$

The news story domain is similar to fraud detection with respect to building discriminating models. Specifically, "positive" news stories can be differentiated from "normal" stories using DC-1's data mining. Because news story text is superficially quite different from cellular call records, the representation had to be adjusted before DC-1 could be applied. Each story was lexically analyzed and reduced to its constituent words. The words were stemmed, then a stop list was applied to remove common noise words. The final representation for a story was a set of its processed words and bi-grams. The reduced stories were each labeled as positive or negative using the heuristics discussed above. DC-1 then learned indicators of positive activity (discriminating models) from these stories. Some specific indicators are shown in Table 1. These indicators were used to form a DC-1 monitor (as described above).

Figure 4 shows the performance of different activity monitors applied to this domain. Each curve was generated by 10-fold cross-validation. **Random** shows the performance of monitors that alarm randomly on stories with varying probabilities (each probability gives one curve point). **DC-1** shows the performance of the DC-1 monitor.

Consider this problem in terms of the distinctions made in Section 3.4. Because of the large number of features (approximately 10^5 words and bi-grams appeared more than once), and because the news is always changing, we would expect little behavioral consistency at the word level. Consequently, we would predict that profiling would add little compared to discriminating; specifically, that the profiling done by DC-1 adds little to the activity monitoring performance on this task. The curve labeled **DC-1 without profiling** demonstrates this: the performance indeed decreases little when profiling is turned off, at which point DC-1 just scans using the learned classifier.

4.3 Intrusion Detection

Intrusion detection is a field of computer security concerned with detecting attacks on computers and

[2]The actual heuristics used are more complex than this, in order to filter secondary stories reporting on the stock's activity. This does not affect the definition of τ.

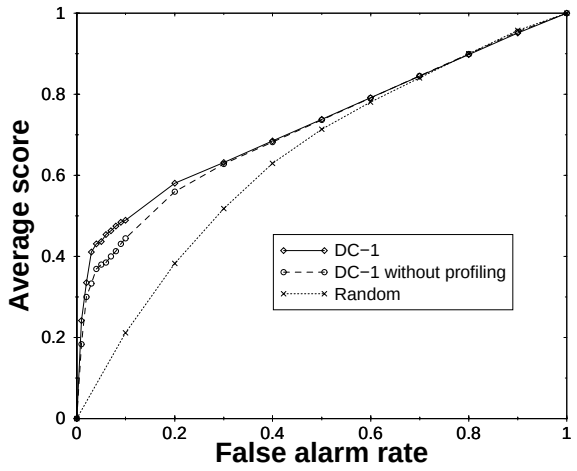


Figure 4: Performance of several methods for news story monitoring

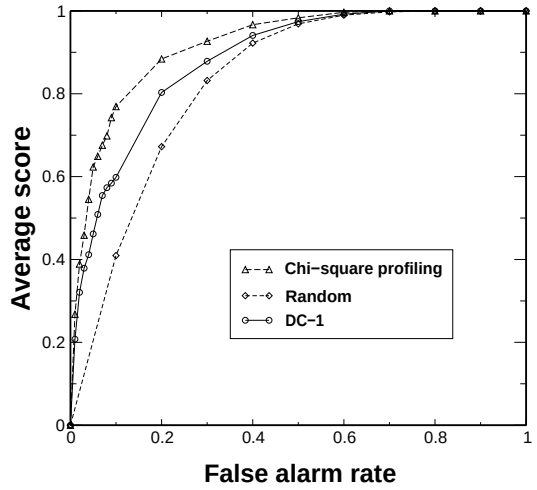


Figure 5: Performance of several methods for intrusion detection

computer networks [8, 9]. Within intrusion detection, *anomaly detection* systems characterize behavior of individual users and issue alarms of intrusions, based on anomalies in behavior.

Since traces of actual computer intrusions are rare and difficult to obtain, we chose a variant of this task common in computer intrusion research [6, 10, 15]. In this variant, the goal is to predict, based on commands typed, when the user of a given account is not the actual legitimate user. With such a task, typically one user at a time is chosen to be the "legitimate" user and one to be the "intruder." The task then becomes one of characterizing the behavior of each user. For this domain we used a dataset of Unix commands taken from about 8000 login sessions collected from a population of 77 users.[3]

Intrusion detection is easily cast as activity monitoring. Each user constitutes a D with an associated stream of sessions d_j. Each session is a set of Unix commands. False positives were normalized per session. We consider an alarm effective only if it occurs within the first five intrusion sessions, so the s function was defined as:

$$s(\tau, \alpha, D) = \begin{cases} 1 & \text{if } 0 \leq \left| \{d_i \in D_\tau \mid time(d_i) \leq \alpha\} \right| < 5 \\ 0 & otherwise \end{cases}$$

Change modeling can be used to discover differences between command sets used by various users. Table 2 shows some of the change indicators extracted from the Davison-Hirsh dataset. Although these commands may seem mundane, some indicate common distinctions among Unix users: `emacs` vs `vi`, `exit` vs `logout`, and

[3]These data were provided by Davison and Hirsh [5] who used them in a study on command sequence prediction.

`gs`	`kill`	`ps`	`emacs`
`a.out`	`project6.out`	`ftp`	`exit`
`Chat_Client`	`gcc`	`finger`	`more`

Table 2: Indicators from change-modeling Unix users.

text processing (`gs`) vs code development (`gcc` and `a.out`).

Figure 5 shows the performance of several classifiers. Each curve was generated by 10-fold cross-validation. **Random** represents the effect of issuing random alarms with varying probabilities (as above). **DC-1** generated indicators such as those above to generate features for distinguishing users, and used these indicators to form its profiling monitor. As shown in Figure 5, **DC-1** detects intrusions better than random.

However, given our experience with the effectiveness of DC-1 for fraud detection, its performance on intrusion detection was disappointing. We believe that the reasons for the performance are clarified by viewing the two problems within the activity monitoring framework. In cellular fraud detection, for which DC-1 was designed, discriminating positive activity is an important component because fraudulent behavior often is quite different from legitimate behavior. Superficially, detection intrusions is very similar to detecting cellular fraud. However, note that there is no distinguishable intruder behavior because any user can fill the role of an intruder with respect to some other user. Discriminative modeling is impracticable because there is no distinct positive activity. Profiling should be much more important.

To (partially) test this hypothesis we created a profiler based on the χ^2 statistical test, which models each

user's normal behavior as a probability distribution across the commands they use. As predicted, the χ^2 technique outperformed the DC-1 technique. A comparison of this simple profiler to existing intrusion detection methods might be revealing.

5 Prior work

Much work in KDD has involved monitoring activity for unusual behavior; for example, fraud detection [3, 4, 7], intrusion detection [6, 10, 11], and network monitoring [16, 17]. Indeed it is this wealth of prior work on closely related problems that makes the formulation of a general class worthwhile. It is important to examine how activity monitoring problems have been framed in prior work, in order to compare them with the framework we propose.

Much prior work frames activity monitoring as a classification problem in which the effectiveness of alarms is evaluated based on simple classification accuracy or error rate. However, such measures are particularly inappropriate for problems such as activity monitoring where one class is rare and where the cost of a miss and the cost of a false alarm are not equal [13].

Other related work addresses unequal costs by framing the task as a cost-sensitive classification problem [3, 7]. Unfortunately, in activity monitoring domains it is usually difficult to specify costs precisely. More importantly, cost-sensitive classification relies on a specification of class priors, which can be even more difficult to determine precisely. In fact, the work cited uses priors known to be inaccurate. It is difficult to generalize from such results without special studies weakening the basic, framing assumption of knowledge of class priors.

Some prior work relies on classification frameworks that make explicit the tradeoff between hits and false alarms, including frameworks that use ROC curves, precision/recall curves, and others [6, 10]. If done well, this addresses the problem of imprecision in knowledge of costs and of class distributions.

However, all these classification-based frameworks share a basic flaw, which our prior work on fraud detection admitted:

> In this work we have dealt with differing costs of false positive and false negative errors. However, we have still glossed over some complexity. For a given account, the only false negative fraud days that incur cost to the company are those prior to the *first* true positive alarm. After the fraud is detected, it is terminated. Thus, our analysis overestimates the costs ... [7, p. 308]

The flaw is that they ignore the fundamental sequential nature of the problem and, thereby, the goal of *timely* classification. We have concentrated on classification-based frameworks because they are the most prevalent; corresponding simplifications are found in activity monitoring work rooted in other disciplines.

Weiss and Hirsh [17] created an activity monitoring framework for their study of activity preceding failures in a telecommunications network. Their framework models the sequential nature of the problem as time-stamped data streams, defines the goal as identifying a "rare event," notes that there is a window of the data stream where (in our terminology) $s(\tau, \alpha, D) > 0$, and looks at the tradeoff between hits and false alarms. Their framework can be expressed within ours by appropriate definitions of τ and $s(\tau, \alpha, D)$, but ours applies more generally. In particular, their framework requires that the event to be identified be one of the d_j. Our framework allows the event initiating the activity to be separate from the data; for example, some action of a company triggers news agencies to report on the action. More fundamentally, Weiss and Hirsh require that the activity *precede* the event in question. While this is appropriate for their application (network performance degrades before a failure), we found it hard to extend in general (predictive activity does not precede a case of fraud). We prefer to recast such problems: τ marks the beginning of the period prior to the failure for which detection will be useful, and activity contained therein would be considered positive in the application. For example, there may be an underlying occurrence that causes (and therefore precedes) the performance degradation; the failure itself, just a further manifestation of the underlying cause, may be a key factor in the definition of s. In other cases, such as those described by Weiss and Hirsh, τ can be defined as a fixed offset prior to the hard failure, representing "the maximum amount of time prior to the target event for which a prediction is considered correct" [17, p. 359].

An important benefit of an activity monitoring framework is to allow fundamentally different approaches to be compared. For example, many fraud detection methods focus on individual transactions [7, 3], and evaluation typically measures classifications of individual transactions. Other methods [7] aggregate transactions into account-days and measure differences in activity between them. A classification-based framework makes performance on different representations (transactions versus account days) difficult to compare. However, they can be compared within the activity monitoring framework.

Activity monitoring tries to identify where in a sequence an interesting change in behavior has occurred. This differs from the typical goals of time series analysis, such as predicting the next value of a series or characterizing the functional form of the series. However, we are not claiming that any of these methods is inappropriate as part of the solution to a activity mon-

itoring problem. Indeed the opposite is true: *all* these methods, standard time-series prediction, HMMs, classification models, and others, are candidate tools for solving an activity monitoring problem.

6 Conclusions

Activity monitoring is worth studying as a general KDD problem. Explicitly differentiating this class of problems should benefit KDD both scientifically and practically. We believe it is an ideal problem for KDD because it requires contributions from (at least) the three main constituent subfields: statistics, for expertise on time-series problems; databases, because of the tremendous amount of data typically involved; and machine learning, because of the representational complexity. We hope this paper has laid valuable groundwork.

7 Acknowledgments

We thank Tom Dietterich for his encouraging words regarding generalizing our work on fraud detection. We thank Brian Davison for providing the Unix command data used in the computer intrusion experiments.

References

[1] Allan, J., Papka, R., and Lavrenko, V. Online new event detection and tracking. In *Proceedings of the 21st ACM-SIGIR International Conference on Research and Development in Information Retrieval* (August 1998).

[2] Burl, M., Asker, L., Smyth, P., Fayyad, U., Perona, P., Aubele, J., and Crumpler, L. Learning to recognize volcanoes on venus. *Machine Learning 30*, 2/3 (1998), 165–194.

[3] Chan, P., and Stolfo, S. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD-98* (1998), Agrawal, Stolorz, and Piatetsky-Shapiro, Eds., AAAI Press, pp. 164–168.

[4] Cox, K., Eick, S., Wills, G., and Brachman, R. Visual data mining: Recognizing telephone calling fraud. *Data Mining and Knowledge Discovery 1*, 2 (1997), 225–231.

[5] Davison, B. D., and Hirsh, H. Predicting sequences of user actions. In *Predicting the Future: AI Approaches to Time Series Problems* (July 1998), AAAI Press, pp. 5–12. WS-98-07.

[6] DuMouchel, W., and Schonlau, M. A fast computer intrusion detection algorithm based on hypothesis testing of command transition probabilities. In *KDD-98* (1998), Agrawal, Stolorz, and Piatetsky-Shapiro, Eds., AAAI Press, pp. 189–193.

[7] Fawcett, T., and Provost, F. Adaptive fraud detection. *Data Mining and Knowledge Discovery 1*, 3 (1997), 291–316.

[8] Frank, J. Machine learning and intrustion detection: Current and future directions. In *National Computer Security Conference* (October 1994), vol. 1, pp. 22–33.

[9] Kumar, S. *A Pattern Matching Approach to Misuse Intrusion Detection*. PhD thesis, Purdue University, Department of Computer Sciences, August 1995.

[10] Lane, T., and Brodley, C. Approaches to online learning and concept drift for user identification in computer security. In *KDD-98* (1998), Agrawal, Stolorz, and Piatetsky-Shapiro, Eds., AAAI Press, pp. 259–263.

[11] Lee, W., Stolfo, S., and Mok, K. Mining audit data to build intrusion detection models. In *KDD-98* (1998), Agrawal, Stolorz, and Piatetsky-Shapiro, Eds., AAAI Press, pp. 66–72.

[12] Provost, F., and Fawcett, T. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *Proceedings of KDD-97* (1997), AAAI Press, pp. 43–48.

[13] Provost, F., Fawcett, T., and Kohavi, R. The case against accuracy estimation for comparing induction algorithms. In *ICML-98* (1998), AAAI Press. Available: `http://www.croftj.net/~fawcett/papers/ICML98-final.ps.gz`.

[14] Rubinstein, Y. D., and Hastie, T. Discriminative vs informative learning. In *KDD-97* (1997), AAAI Press, pp. 49–60.

[15] Ryan, J., Lin, M.-J., and Miikkulainen, R. Intrusion detection with neural networks. In *AI Approaches to Fraud Detection and Risk Management* (1997), Fawcett, Haimowitz, Provost, and Stolfo, Eds., AAAI Press.

[16] Sasisekharan, R., Seshadro, V., and Weiss, S. M. Proactive network maintenance using machine learning. In *KDD-94* (1994), pp. 453–461.

[17] Weiss, G., and Hirsh, H. Learning to predict rare events in event sequences. In *KDD-98* (1998), Agrawal, Stolorz, and Piatetsky-Shapiro, Eds., AAAI Press, pp. 359–363.