Rule-Based Anomaly Pattern Detection for Detecting Disease Outbreaks

Weng-Keen Wong
Dept of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213
wkw@cs.cmu.edu

Andrew Moore
Dept of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213
awm@cs.cmu.edu

Gregory Cooper
Ctr for Biomedical Informatics
University of Pittsburgh
Pittsburgh, PA 15213
gfc@cbmi.upmc.edu

Michael Wagner
Ctr for Biomedical Informatics
University of Pittsburgh
Pittsburgh, PA 15213
mmw@cbmi.upmc.edu

Abstract

This paper presents an algorithm for performing early detection of disease outbreaks by searching a database of emergency department cases for anomalous patterns. Traditional techniques for anomaly detection are unsatisfactory for this problem because they identify individual data points that are rare due to particular combinations of features. When applied to our scenario, these traditional algorithms discover isolated outliers of particularly strange events, such as someone accidentally shooting their ear, that are not indicative of a new outbreak. Instead, we would like to detect anomalous patterns. These patterns are groups with specific characteristics whose recent pattern of illness is anomalous relative to historical patterns. We propose using a rule-based anomaly detection algorithm that characterizes each anomalous pattern with a rule. The significance of each rule is carefully evaluated using Fisher's Exact Test and a randomization test. Our algorithm is compared against a standard detection algorithm by measuring the number of false positives and the timeliness of detection. Simulated data, produced by a simulator that creates the effects of an epidemic on a city, is used for evaluation. The results indicate that our algorithm has significantly better detection times for common significance thresholds while having a slightly higher false positive rate.

Introduction

Multidimensional data with a temporal component is available from numerous disciplines such as medicine, engineering, and astrophysics. This data is commonly used for monitoring purposes by a detection system. These systems inspect the data for anomalies and raise an appropriate alert upon discovery of any deviations from the norm. For example, in the case of an intrusion detection system, an anomaly would indicate a possible breach of security (Lane & Brodley 1999; Eskin 2000; Maxion & Tan 2001).

We would like to tackle the problem of early disease outbreak detection in a similar manner. In our situation, we have a database of emergency department (ED) cases from several hospitals in a city. Each record in this database contains information about the individual who was admitted to the ED. This information includes fields such as age, gender, symptoms exhibited, home location, work location, and time admitted. (To maintain patient confidentiality, personal identifying information, such as patient names, addresses, and identification numbers were not in the dataset used in this research.) Clearly, when an epidemic sweeps through a region, there will be extreme perturbations in the number of ED visits. While these dramatic upswings are easily noticed during the late stages of an epidemic, the challenge is to detect the outbreak during its early stages and mitigate its effects. Different diseases cause different signals to appear in temporal, spatial and demographic data (Wagner *et al.* 2001). In order for any anomaly detection algorithm to be successful in early detection of disease outbreaks, it must be able to detect abnormalities in these three aspects of ED data.

Although we have posed our problem in an anomaly detection framework, the majority of anomaly detection algorithms are inappropriate for this domain. In this section, we will illustrate the shortcomings of traditional anomaly detection techniques in our task of early epidemic detection.

A simplistic first approach would be to report an ED case as an anomaly if it has a rare value for some attribute. As an example, we would signal an anomaly if we encountered a patient over a hundred years old. While this method detects the outliers for a single attribute, it fails to identify anomalies that occur due to combinations of features which by themselves might not be abnormal but together would certainly be unusual. For instance, the first technique would not find anomalies in cases where the patients were male and under the age of thirty but exhibited symptoms that were associated with a disease that affects primarily female senior citizens. Fortunately, there are plenty of anomaly detection algorithms that can identify outliers in multidimensional feature space. Typically these detection algorithms build a probabilistic model of the "normal" data using a variety of techniques such as neural nets (Bishop 1994) and a mixture of naive Bayes submodels (Hamerly & Elkan 2001)

However, even that kind of sophisticated outlier detection is insufficient for our purposes. Outlier detection succeeds at finding data points that are rare based on the underlying density, but these data points are treated in isolation from each other. Early epidemic detection, on the other hand, hinges on identifying anomalous groups, which we will refer to as *anomalous patterns*. Specifically, we want to know if the recent proportion of a group with specific characteristics is anomalous based on what the proportion is normally. Traditional outlier detection will likely return isolated irregularities that are insignificant to the early detection system.

We might then argue that aggregate daily counts of a single attribute or combination of attributes should be monitored in order to detect an anomalous group. For instance, we could monitor the daily number of people appearing in the ED with respiratory problems. A naive detector would determine the mean and variance of the monitored signal over a training set which is assumed to capture the normal behaviour of the system. Then, a threshold would be established based on these values. Whenever the daily count exceeds this threshold, an alert is raised. This technique works well if the monitored features are known. However, the spatial, temporal, and demographic signatures of diseases are simply too wide a space for us to know a priori what features to monitor. We could well miss some combination of features that would indicate an outbreak of a particular disease. Thus, we need an algorithm that is able to detect anomalous patterns rather than pre-defined anomalies.

Our approach to this problem uses a rule-based anomaly pattern detector. Each anomalous pattern is summarized by a rule, which in our current implementation consists of one or two components. Each component takes the form $X_i = V_i^j$, where X_i is the ith feature and V_i^j is the jth value of that feature. Multiple components are joined together by a logical AND. For example, a two component rule would be Gender = Male and Age_Decile = 4. One benefit to a rule-based system is that the rules are easily understood by a non-statistician.

However, we need to be wary of the pitfalls of rule-based anomaly pattern detection. Since we are finding anomalous patterns rather than isolated anomalies, we will be performing multiple hypothesis tests. When multiple hypothesis tests are performed, the probability of a false positive becomes inflated unless a correction is made (Benjamini & Hochberg 1995). In addition, as we add more components to a rule, overfitting becomes a serious concern. A careful evaluation of significance is clearly needed. Furthermore, temporal healthcare data used for disease outbreak detection, are frequently subject to “seasonal” variations. As an example, the number of influenza cases is typically higher during winter than summer. Additionally, the number of ED visits vary between weekends and weekdays. The definition of what is normal will change depending on these variations.

Rule-based Anomaly Pattern Detection

The basic question asked by all detection systems is whether anything strange has occurred in recent events. This question requires defining what it means to be recent and what it means to be strange. Our algorithm considers all patient records falling on the current day under evaluation to be recent events. Note that this definition of recent is not restrictive – our approach is fully general and recent can be defined to include all events within some other time period. In order to define an anomaly, we need to establish the concept of something being normal. Our algorithm is intended to be applied to a database of ED cases and we need to account for environmental factors such as weekend versus weekday differences in the number of cases. Consequently, normal behaviour is assumed to be captured by the events occurring on the days that are exactly five, six, seven, and eight weeks prior to the day under consideration. The definition of what is normal can be easily modified to another time period without major changes to our algorithm. We will refer to the events that fit a certain rule for the current day as C_{today}. Similarly, the number of cases matching the same rule from five to eight weeks ago will be called C_{other}.

From this point on, we will refer to our algorithm as WSARE, which is an abbreviation for “What’s strange about recent events”. WSARE operates on discrete data sets with the aim of finding rules that characterize significant patterns of anomalies. Due to computational issues, the number of components for these rules is two or less. The description of the rule-finding algorithm will begin with an overview followed by a more detailed example.

Overview of WSARE The best rule for a day is found by considering all possible one and two component rules over events occurring on that day and returning the one with the best “score”. The score is determined by comparing the events on the current day against events in the past. Following the score calculation, the best rule for that day has its p-value estimated by a randomization test. The p-value for a rule is the likelihood of finding a rule with as good a score under the hypothesis that the case features and date are independent. The randomization-based p-value takes into account the effect of the multiple testing that went on during the rule search. If we were running the algorithm on a day-by-day basis we would end at this step. However, if we were looking at a history of days, we would need the additional step of using the False Discovery Rate (FDR) method (Benjamini & Hochberg 1995) to determine which of the p-values are significant. The days with significant p-values are returned as the anomalies.

One component rules In order to illustrate this algorithm, suppose we have a large database of 1,000,000 ED records over a two-year span. This database contains roughly 1000 records a day, thereby yielding approximately 5000 records if we consider the cases for today plus those from five to eight weeks ago. We will refer to this record subset as DB_i, which corresponds to the recent event data set for day i. The algorithm proceeds as follows. For each day i, retrieve the records belonging to DB_i. We first consider all possible one-component rules. For every possible feature-value combination, obtain the counts C_{today} and C_{other} from the data set DB_i. As an example, suppose the feature under consideration is the Age_Decile for the ED case. There are 9 possible Age_Decile values, ranging from 0 to 8. We start with the rule Age_Decile $= 3$ and count the number of cases for the current day i that have Age_Decile $= 3$ and those that have Age_Decile $\neq 3$. The cases from five to eight weeks ago are subsequently examined to obtain the counts for the cases matching the rule and those not matching the rule. The four values form a two-by-two contingency table such as the one shown in Table 1.

Scoring each one component rule The next step is to evaluate the "score" of the rule using a hypothesis test in which the null hypothesis is the independence of the row and column attributes of the two-by-two contingency table. In effect, the hypothesis test measures how different the distribution for C_{today} is compared to that of C_{other}. This test will generate a p-value that determines the significance of the anomalies found by the rule. We will refer to this p-value as the *score* in order to differentiate this p-value from the p-value that is obtained later on from the randomization test. We use the Chi Squared test for independence of variables whenever the counts in the contingency table do not violate the validity of the Chi Squared test. However, since we are searching for anomalies, the counts in the contingency table frequently involve small numbers. In this case, we use Fisher's Exact Test (Good 2000) to find the score for each rule. Running Fisher's Exact Test on Table 1 yields a score of 0.00005058, which indicates that the count C_{today} for cases matching the rule Age_Decile $= 3$ are significantly different from the count C_{other}.

	C_{today}	C_{other}
$Age\_Decile = 3$	48	45
$Age\_Decile \neq 3$	86	220

Table 1: A Sample 2x2 Contingency Table

Two component rules At this point, the best one component rule for a particular day has been found. We will refer to the best one component rule for day i as BR_i^1. The algorithm then attempts to find the best two component rule for the day by adding on one extra component to BR_i^1. This extra component is determined by supplementing BR_i^1 with all possible feature-value pairs, except for the one already present in BR_i^1, and selecting the resulting two component rule with the best score. Scoring is performed in the exact same manner as before, except the counts C_{today} and C_{other} are calculated by counting the records that match the two component rule. The best two-component rule for day i is subsequently found and we will refer to it as BR_i^2

BR_i^2, however, may not be an improvement over BR_i^1. We need to perform further hypothesis tests to determine if the presence of either component has a significant effect. This can be accomplished by determining the scores of having each component through Fisher's Exact Test. If we label BR_i^2's components as C_0 and C_1, then the two 2-by-2 contingency tables for Fisher's Exact Tests are as follows:

Records from Today matching C_0 and C_1	Records from Other matching C_0 and C_1
Records from Today matching C_1 and differing on C_0	Records from Other matching C_1 and differing on C_0

Table 2: First 2x2 Contingency Table 1 for a Two Component Rule

Records from Today matching C_0 and C_1	Records from Other matching C_0 and C_1
Records from Today matching C_0 and differing on C_1	Records from Other matching C_0 and differing on C_1

Table 3: Second 2x2 Contingency Table 2 for a Two Component Rule

Once we have the scores for both tables, we need to determine if they are significant or not. We used the standard α value of 0.05 and considered a score to be significant if it was less than or equal to α. If the scores for the two tables were both significant, then the presence of both components had an effect. As a result, the best rule overall for day i is BR_i^2. On the other hand, if any one of the scores was not significant, then the best rule overall for day i is BR_i^1.

Finding the p-value for a rule The algorithm above for determining scores is extremely prone to overfitting. Even if data were generated randomly, most single rules would have insignificant p-values but the best rule would be significant if we had searched over 1000 possible rules. In order to illustrate this point, suppose we follow the standard practice of rejecting the null hypothesis when the p-value is $< \alpha$, where $\alpha = 0.05$. In the case of a single hypothesis test, the probability of making a false discovery under the null hypothesis would be α, which equals 0.05. On the other hand, if we perform 1000 hypothesis tests, one for each possible rule under consideration, then the probability of making a false discovery could be as bad as $1 - (1 - 0.05)^{1000} \approx 1$, which is much greater than 0.05 (Miller *et al.* 2001). Thus, if our algorithm returns a significant p-value, we cannot accept it at face value without adding an adjustment for the multiple hypothesis tests we performed. This problem can be addressed using a Bonferroni correction (Bonferroni 1936) but this approach would be unnecessarily conservative. Instead, we turn to a randomization test in which the date and each ED case features are assumed to be independent. In this test, the case features in the data set DB_i remain the same for each record but the date field is shuffled between records from the current day and records from five to eight weeks ago. The full method for the randomization test is shown below.

```
Let $UCP_i$ = Uncompensated p-value ie. the score
              as defined above.

For j = 1 to 1000
    Let $DB_i^{(j)}$ = newly randomized dataset
    Let $BR_i^{(j)}$ = Best rule on $DB_i^{(j)}$
    Let $UCP_i^{(j)}$ = Uncompensated p-value of $BR_i^{(j)}$ on
                  $DB_i^j$
Let the compensated p-value of $BR_i$ be $CPV_i$ ie.
```

$$CPV_i = \frac{\text{\# of Randomized Tests in which } UCP_i^j > UCP_i}{\text{\# of Randomized Tests}}$$

It is clear from this procedure that CPV_i is an estimate of the chance that we would have seen an uncompensated p-value as good as UCP_i if in fact there was no relationship between date and case features. In practice, for computational reasons, we involve the old idea of "racing" (Maron & Moore 1997) during the randomization procedure. If BR_i is highly significant, we run the full 1000 iterations but we stop early if we can show with very high confidence that CPV_i is going to be greater than 0.05.

Using FDR to determine which p-values are significant This algorithm can be used on a day-to-day basis similar to an online algorithm or it can operate over a history of several days to report all significantly anomalous patterns. When using our algorithm on a day-to-day basis, the compensated p-value CPV_i obtained for the current day through the randomization tests can be interpreted at face value. However, when analyzing historical data, we need to compare the CPV values for each day in the history. Comparison of multiple CPV values results in a second overfitting opportunity analogous to that caused by performing multiple hypothesis tests to determine the best rule for a particular day. As an illustration, suppose we took 500 days of randomly generated data. Then, approximately 5 days would have a CPV value less than 0.01 and these days would naively be interpreted as being significant. Two approaches can be used to correct this problem. Again, the Bonferroni method (Bonferroni 1936) aims to reduce the probability of making at least one false positive to be no greater than α. However, this tight control over the number of false positives causes many real discoveries to be missed (Miller *et al.* 2001). The other alternative is the False Discover Rate (FDR) method (Benjamini & Hochberg 1995; Miller *et al.* 2001), which guarantees that the fraction of the number of false positives over the number of tests in which the null hypothesis was rejected will be no greater than α. The FDR method is more desirable as it has a higher power than the Bonferroni method but still has reasonable control over the number of false positives. We incorporate the FDR method into our rule-learning algorithm by first providing an α value and then using FDR to find the cutoff threshold for determining which p-values are significant.

The Simulator

Validation of our algorithm is a difficult task due to the type of data required. Data consisting of ED cases during a disease outbreak is extremely limited and there are few available databases of ED cases during a bioagent release. To make matters more difficult, evaluation of our anomaly pattern detector requires a large amount of data that has records that are labeled as either anomalies or normal events. [1] In most cases, this task requires a human to perform the labelling by hand, resulting in an insufficient amount of data. As a result of these limitations, we resort to evaluating our algorithm using data from a simulator.

The simulator is intended to simulate (to a first approximation) the effects of an epidemic on a population. The world in this simulator consists of a grid in which there are three types of objects – places, people, and diseases. These three objects interact with each other in a daily routine for a fixed number of days. Each of these objects will be described in detail below.

Places The three types of places in the simulator include homes, businesses, and restaurants. Their roles are evident from what they represent in real life. People reside in homes, work in businesses and eat in restaurants.

People Each person in the simulation has a specified gender and age. Genders for the population are distributed uniformly between male and female while ages follow a normal distribution with mean 40 and standard deviation of 15. People have a home location, a work location, a list of restaurants that they eat at and a list of homes of friends that they like to visit. The locations of work, restaurants, and friends' homes are chosen to be in close proximity to a person's home. On each day, a schedule is generated for a person. In this schedule, people sleep at home until it is time to go to work. They go to work, stop for a lunch break at a restaurant, and then return to work. After work, they spend some time at home before going to a restaurant for dinner. Following dinner, they visit a random selection of friends at their houses. Finally they return home to sleep.

Diseases Diseases are the most complex objects in the simulator as they are designed to allow the creation of a large variety of disease models. People, places and grid cells can all serve as infection agents since they can all carry a disease. With infected places, we can create diseases that spread by a contaminated food supply while with infected grid cells, we can model airborne infections. Associated with each disease is a spontaneous generation probability which corresponds to how likely the disease is to appear in the population at each timestep. Typically, this probability is extremely small. Each disease also progresses through several stages at different rates. On each stage, the infected person can exhibit a variety of symptoms. The current simulation chooses randomly from a list of symptoms at each stage of the disease. At the final stage, an infected agent can either recover or die. The deceased are removed from the simulation.

The entire infection process revolves around the infection probability, which controls how easily an infected person can pass the disease on to another on each timestep. A radius parameter determines how close a person needs to be to catch the disease. The simulator only allows a person to have one disease at a time. Should more than one disease infect a person, the priority of an epidemic arbitrates which disease is assigned to the person. Diseases can be designed to spread from one particular type of agent to another for example place to person, person to person, or grid cell to person. Additionally, each disease has a specific demographic group that it infects. Whenever it has an opportunity to spread to a person outside of this demographic group, the

[1] Of course, labelled data is only needed for evaluation and validation. In regular deployment, WSARE is applied to unlabelled data

infection probability is reduced to a small percentage of its original value.

We do not have hospitals in the simulation. Instead, when people exhibit a certain symptom, we create an ED case by adding an entry to a log file. This entry contains information such as the person id, the day, the time, the current location of the person, the home location of the person, and any demographic information about the individual. Most importantly, we add to each entry the actual disease carried by that person, though this last piece of information is hidden from the anomaly detector.

Results

Simulation Settings Our results were obtained by running the simulator on a 50 by 50 grid world with 1000 people, 350 homes, 200 businesses, and 100 restaurants. The simulation ran for 180 simulated days with the epidemic being introduced into the environment on the 90th day. There are nine background diseases that spontaneously appeared at random points in the simulation. At certain stages, these background diseases caused infected people to display the monitored symptom. These background diseases had low infection probabilities as they were intended to provide a baseline for the number of ED cases. The epidemic, on the other hand, had a higher priority than the background diseases and it had a relatively high infection probability, making it spread easily through its target demographic group.

The epidemic that we added to the system will be referred to as Epidemic0. This disease had a target demographic group of males in their 50s. Additionally, the disease is permitted to contaminate places. Epidemic0 had 4 stages with each stage lasting for two days. The disease was contagious during all four stages. At the final stage, we allowed the person to recover instead of dying in order to keep the total number of people in the simulation constant. Epidemic0 also exhibited the monitored symptom with probability 0.33 on the third stage, probability 1.0 on the final stage, and probability 0 on all other stages. This disease was designed to produce a subtle increase in the number of daily ED counts rather than causing extreme perturbations that could easily be picked up by the naive algorithm.

Evaluation of performance We treated our algorithm as if it ran on a day-by-day basis. Thus, for each day in the simulation, WSARE was asked to determine if the events on the current day were anomalous. We evaluated the performance of WSARE against a standard anomaly detection algorithm that treated a day as anomalous when the daily count of ED cases for the monitor symptom exceeded a threshold. The standard detector was allowed to train on the ED case data from day 30 to day 89 in the simulation to obtain the mean μ and variance σ^2. The threshold was calculated by the formula below, in which Φ^{-1} is the inverse to the cumulative distribution function of a standard normal.

$$\text{threshold} = \mu + \sigma * \Phi^{-1}(1 - \frac{\text{p-value}}{2})$$

In order to illustrate the standard algorithm, suppose we trained on the data from day 30 to 89. The mean and variance of the daily counts of the monitored symptom on this training set were determined to be 20 and 8 respectively. Given a p-value of 0.05, we calculate the threshold as $20 + 1.96 * \sqrt{8} = 25.54$. After training, the standard algorithm is run over all the days of data from day 0 to day 179. Any day in which the daily count of the particular symptom exceeds 25.54 is considered to contain anomalous events.

Both the standard algorithm and WSARE were tested using five levels of p-values (0.1, 0.05, 0.01, 0.005, and 0.001). In order to evaluate the performance of the algorithms, we measured the number of false positives and the number of days until the epidemic was detected. Note that there were two files used in this evaluation step. The first file is the database of ED cases produced by the simulator, which we will refer to as DB_{ED}. The second file is the list of anomalous days reported by the algorithm, which we will refer to as DB_{Anom}. We will call the subset of anomalies having a p-value below the ith p-value level as DB^i_{Anom}.

1. **Counting the number of false positives**
   The number of false positives for the ith p-value level was determined by checking each day in DB^i_{Anom} against DB_{ED}. If a case of the epidemic was not reported in DB_{ED} for that day, then the false positive count was incremented. However, since WSARE relies on data from five to eight weeks prior to the current day, detection does not begin until Day 56. In order to be fair, any false positives found before Day 56 in the standard algorithm were not included.

2. **Calculating time until detection**
   The detection time for the ith p-value level was calculated by searching for the first day in DB^i_{Anom} in which an epidemic case appeared in DB_{ED}. If no such days are found, the detection time was set to be 90 days ie. the maximum length between the introduction of the epidemic until the end of the simulation.

Figures 1 and 2 plot the detection time in days versus the number of false positives for five different p-value thresholds used in both the standard algorithm and WSARE. In Figure 1, the error bars for detection time and false positives are shown. Figure 2 fills in the lines to illustrate the asymptotic behaviour of the curves. These values were generated by taking the average over 100 runs of the simulation.

Results from Simulated Data

These results indicate that for p-value thresholds above 0.01, the detection time for WSARE is significantly smaller than that of the standard algorithm. On the other hand, as the p-value threshold decreases, the detection time for WSARE is somewhat worse than that of the standard algorithm. However, choosing an extremely low threshold would be unprofitable since all anomalies except those at an unusually high significance level would be ignored. For example, using a threshold of 0.01 corresponds to a 99% significance level.

The results also demonstrate that WSARE signals more false positives for higher p-value thresholds. While this behaviour is not desirable, it is tolerable since the number of false positives produced by WSARE differs by a small amount from the count generated by the standard algorithm. In this particular graph, there are at most 3 more false positives identified by WSARE that were not identified by the standard algorithm.

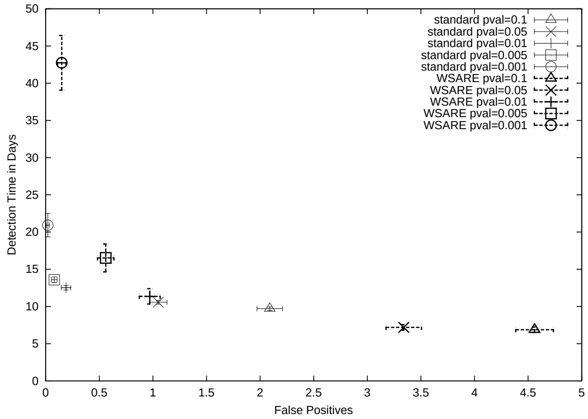

Figure 1: Scatterplot of Detection Time vs False Positives with Error Bars for Detection Time and False Positives

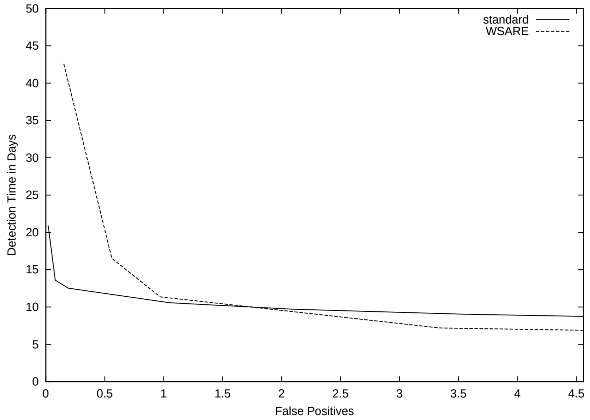

Figure 2: Plot of Detection Time vs False Positives

We now show some of the rules learned by WSARE. The rules below were obtained from one of the result generating simulations.

```
### Rule 1: Sat Day97 (daynum 97, dayindex 97)
SCORE = -0.00000011 PVALUE = 0.00249875
 33.33% ( 16/ 48) of today's cases have Age Decile = 5 and Gender = Male
  3.85% (  7/182) of other cases have Age Decile = 5 and Gender = Male

### Rule 2: Tue Day100 (daynum 100, dayindex 100)
SCORE = -0.00001093  PVALUE = 0.02698651
 30.19% ( 16/ 53) of today's cases have Age Decile = 5 and Col2 less than 25
  6.19% ( 12/194) of   other cases have Age Decile = 5 and Col2 less than 25
```

In rule 1, WSARE demonstrates that it is capable of finding the target demographic group that Epidemic0 infects. This rule proves to be significant above the 99% level. On the other hand, Rule 2 discovers something that was not deliberately hardcoded into Epidemic0. Rule 2 states that on Day 100, there is an unusually large number of cases involving people in their fifties that were all in the left half of the grid. Since we had designed the people in the simulation to interact with places that are in close geographic proximity to their homes, we suspected that the locality of interaction of infected individuals would form some spatial clusters of ED cases. Upon further inspection of the log files, we discovered that 12 of the 16 cases from the current day that satisfied this rule were in fact caused by Epidemic0. This example illustrates the capability of WSARE to detect significant anomalous patterns that are completely unexpected.

Results from Real ED data

We also ran WSARE on an actual ED data collected from hospitals in a major US city. This database contained approximately 70000 records collected over a period of 505 days. Since we are looking at historical data, we need to use FDR to determine which of the p-values are significant. The results are shown below with α for FDR equal to 0.1.

```
### Rule 1: Tue 05-16-2000 (daynum 36661, dayindex 18)
SCORE = -0.00000000  PVALUE = 0.00000000
 32.84% ( 44/134) of today's cases have Time Of Day4 after 6:00 pm
 90.00% ( 27/ 30) of   other cases have Time Of Day4 after 6:00 pm

### Rule 2: Fri 06-30-2000 (daynum 36706, dayindex 63)
SCORE = -0.00000000  PVALUE = 0.00000000
 19.40% ( 26/134) of today's cases have Place2 = NE and Lat4 = d
  5.71% ( 16/280) of   other cases have Place2 = NE and Lat4 = d

### Rule 3: Wed 09-06-2000 (daynum 36774, dayindex 131)
SCORE = -0.00000000  PVALUE = 0.00000000
 17.16% ( 23/134) of today's cases have Prodrome = Respiratory
and age2 less than 40
  4.53% ( 12/265) of other cases have Prodrome = Respiratory
and age2 less than 40

### Rule 4: Fri 12-01-2000 (daynum 36860, dayindex 217)
SCORE = -0.00000000  PVALUE = 0.00000000
 22.88% ( 27/118) of today's cases have Time Of Day4
after 6:00 pm and Lat2 = s
  8.10% ( 20/247) of   other cases have Time Of Day4
after 6:00 pm and Lat2 = s

### Rule 5: Sat 12-23-2000 (daynum 36882, dayindex 239)
SCORE = -0.00000000  PVALUE = 0.00000000
 18.25% ( 25/137) of today's cases have ICD9 = shortness of breath
and Time Of Day2 before 3:00 pm
  5.12% ( 15/293) of   other cases have ICD9 = shortness of breath
and Time Of Day2 before 3:00 pm

### Rule 6: Fri 09-14-2001 (daynum 37147, dayindex 504)
SCORE = -0.00000000  PVALUE = 0.00000000
 66.67% ( 30/ 45) of today's cases have Time Of Day4 before 10:00 am
 18.42% ( 42/228) of   other cases have Time Of Day4 before 10:00 am
```

Rule 1 notices that there are fewer cases after 6:00 pm quite possibly due a lack of reporting by some hospitals. Rule 6 correctly identifies a larger volume of data being collected before 10:00 am on Day 504. Since Day 504 was the last day of this database, this irregularity was the result of the database being given to us in the morning.

We are currently beginning the process of using input from public health officials of the city concerned to help us validate and measure WSARE's performance.

Future work

The algorithm described is computationally intensive, particularly when performing the many randomization tests required to obtain a good estimate of a rule's true p-value. Future research involves speeding up the randomization tests by using data structures that can be efficiently updated when the database is randomized. In addition, we would like to automatically model the "normal" database rather than using an arbitrary selection process of using data from five to eight weeks prior to the current date.

Related Work

Our approach is closely related to the work done by Bay and Pazzani (Bay & Pazzani 1999) in mining contrast sets. Contrast sets are conjunctions of attributes and values whose support differs significantly between groups. In (Bay & Pazzani 1999), the authors perform multiple hypothesis tests while searching for significant contrast sets. A Bonferroni correction is used to control the probability of a Type I error. The paper also prunes all contrast sets that cease to yield a valid chi-square test due to insufficient data points. Our approach is also somewhat similar to itemset mining (Brin *et al.* 1997). Other papers that deal with early disease outbreak detection include (Wagner *et al.* 2001) and (Goldenberg 2001).

Conclusion

WSARE has been demonstrated to be successful at identifying anomalous patterns in the data. From our simulation results, WSARE has significantly lower detection times than a standard detection algorithm provided the p-value threshold is not at at extremely low level. This condition should not be a problem since most anomalies are reported at a significance level of 95% or 99%, corresponding respectively to p-value thresholds of 0.05 and 0.01. WSARE also has a slightly higher false positive rate than the standard algorithm. However, this difference was shown to be about 3 more false positives in the worst case for our particular simulation.

We believe the three main innovations in this paper are:

1. Turning the problem of "detect the emergence of new patterns in recent data" into the question "is it possible to learn a propositional rule that can significantly distinguish whether records are most likely to have come from the recent past or longer past?"
2. Incorporating several levels of significance tests into rule learning in order to avoid several levels of overfitting caused by intensive multiple testing
3. Examining the interesting domain of early outbreak detection by means of machine learning tools

Acknowledgements

This software was devised, implemented and tested through a research grant to the University of Pittsburgh and Carnegie Mellon University funded by the DARPA BIOALIRT program.

References

Bay, S. D., and Pazzani, M. J. 1999. Detecting change in categorical data: Mining contrast sets. In *Knowledge Discovery and Data Mining*, 302–306.

Benjamini, Y., and Hochberg, Y. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B.* 57:289–300.

Bishop, C. M. 1994. Novelty detection and neural network validation. *IEEE Proceedings - Vision, Image and Signal Processing* 141(4):217–222.

Bonferroni, C. E. 1936. Teoria statistica delle classi e calcolo delle probabilità. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze* 8:3–62.

Brin, S.; Motwani, R.; Ullman, J. D.; and Tsur, S. 1997. Dynamic itemset counting and implication rules for market basket data. In Peckham, J., ed., *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, 255–264. ACM Press.

Eskin, E. 2000. Anomaly detection over noisy data using learned probability distributions. In *Proceedings of the 2000 International Conference on Machine Learning (ICML-2000)*.

Goldenberg, A. 2001. Framework for using grocery data for early detection of bio-terrorism attacks. Master's thesis, Carnegie Mellon University.

Good, P. 2000. *Permutation Tests - A Practical Guide to Resampling Methods for Testing Hypotheses*. New York: Springer-Verlag, 2nd edition.

Hamerly, G., and Elkan, C. 2001. Bayesian approaches to failure prediction for disk drives. In *Proceedings of the eighteenth international conference on machine learning*, 202–209. Morgan Kaufmann, San Francisco, CA.

Lane, T., and Brodley, C. E. 1999. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security* 2:295–331.

Maron, O., and Moore, A. W. 1997. The racing algorithm: Model selection for lazy learners. *Artificial Intelligence Review* 11(1-5):193–225.

Maxion, R. A., and Tan, K. M. C. 2001. Anomaly detection in embedded systems. Technical Report CMU-CS-01-157, Carnegie Mellon University.

Miller, C. J.; Genovese, C.; Nichol, R. C.; Wasserman, L.; Connolly, A.; Reichart, D.; Hopkins, A.; Schneider, J.; and Moore, A. 2001. Controlling the false discovery rate in astrophysical data analysis. Technical report, Carnegie Mellon University.

Wagner, M. M.; Tsui, F. C.; Espino, J. U.; Dato, V. M.; Sittig, D. F.; Caruana, R. A.; McGinnis, L. F.; Deerfield, D. W.; Druzdzel, M. J.; and Fridsma, D. B. 2001. The emerging science of very early detection of disease outbreaks. *Journal of Public Health Management Practice* 7(6):51–59.