Cluster analysis and its application in anomaly detection

IMT 4741 Intrusion detection and prevention
Gjøvik University College
Student 995995

Abstract

In this project report we will look at methods for anomaly based intrusion detection based on clustering with focus on defining what assumptions they are based on, some examples of proposed methods and their performance.

1. Introduction

The goal of an intrusion detection system (IDS) is to automate detection of attempts at compromising the security goals of a system of computing devices, often connected by networks. Some systems extend the IDS by also automating the response upon a detected compromise and are called intrusion prevention systems (IPS), but are in practice the same thing regarding detection. There are in principle two approaches for defining intrusions [Song et al., 2013]: Misuse based and anomaly based.

In the misuse approach we try to define what misuse such as intrusions look like in terms of observable evidence. Everything that does not match these criteria, often called signatures, are assumed benign. This is very much similar to the concept of black listing. As long as the definitions of malicious is precise enough, we would expect low rates of false alarms, but at the same time the system is not able to detect non-described attacks. In addition, if the attack is described too strict, it might miss attacks performed in a slightly different way. New attack vectors are discovered all the time and misuse based systems can be considered a way to automate the task of recognizing these.

In anomaly based systems we instead try to describe what benign activity looks like, and then alert on things that fall outside, hence the term anomaly. This is similar to white listing. If benign activity can be defined precisely, then all misuse can be alerted on. The obvious downfall of this approach is that failing to describe normal activity precisely will lead to a lot of false alarms and missed attacks.

IDS can also be classified in terms of placement [Beigh and Peer, 2012]. An intrusion detection system can be located at individual hosts (host based), and they can be located on the network connecting the devices (network based). A combination (hybrid) is also possible. The amount and type of evidence available will naturally depend on the placement of the sensor. For example, a host based sensor can have access to data that is otherwise encrypted on the network and the state of programs running on it. A network based IDS will have access to aggregated communication of several hosts and thus cover more ground, and might even be more reliable in case of a compromised host, but at the same time miss out on important state changes on the hosts it is protecting.

In this paper focus will be on how cluster analysis can be applied in anomaly detection. The goal of clustering algorithms is to group similar observations based on some criteria. Clustering analysis is a category of unsupervised learning and refers to the fact that the algorithm does not have any knowledge of the data. This in contrast to supervised learning where data is associated with an additional label such as normal or attack, or even finer level labels such as category of attack. Unsupervised algorithms are specially interesting because access to classified data to work on is costly to generate, and given the assumption that an attack is distinct from legitimate use, all we need to do is to figure out what properties to monitor and what algorithms to separate the cases.

In the following chapter we will first have a brief overview of the lecture material for this course, followed by some examples from different research papers, followed by some discussion regarding the performance of these methods.

2. Main content

For unsupervised anomaly detection algorithms [Leung and Leckie, 2005] mention (and cites) two assumptions. They are written in the context of network based IDS, but in general they say: We assume that in a given set of observations, only a very small number of them will actually be intrusions, and the other say that: Observations of intrusions will be outliers. If the traces of intrusive and benign activity are not inherently different, then it would be meaningless trying to separate them out. There are discussions regarding the size of overlap between the categories normal and misuse as mentioned in the courses slides[1].

Another implicit assumption is that these traces are observable [Petrovic, 2013]. This is actually very important to realize. To exemplify this let's say a miscreant gets access to an employees computer during normal working hours and has access to the correct credentials from a written note and now able to change a record in a database normally accessed by the employee. This would be defined as an intrusion because of the context, but to the system everything is normal. The evidence that the credentials were used by an unauthorized person is not available to the machine. The takeaway from this assumption is how important it is to select proper things to monitor and monitor them at the most efficient location.

2.1 Summary of course material

Based on the lecture material [Petrovic, 2013], clustering are non-parametric methods, meaning no assumptions is made of the distribution of the observations used. Clustering was defined this way:

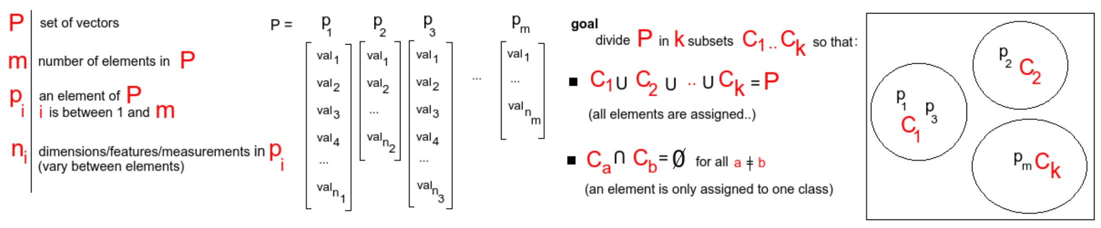

P set of vectors
m number of elements in P
p_i an element of P
i is between 1 and m
n_i dimensions/features/measurements in p_i (vary between elements)
P = p_1 p_2 p_3 ... p_m
val_1 val_2 val_3 val_4 ... val_{n_1}
goal
divide P in k subsets C_1..C_k so that:
C_1 ∪ C_2 ∪ .. ∪ C_k = P
(all elements are assigned..)
C_a ∩ C_b = Ø for all a ≠ b
(an element is only assigned to one class)
p_1 p_3 C_1
p_2 C_2
p_m C_k

Figure 1: *General definition of clustering*

In figure 1, **P** can be though of as a collection of records where each record consists of a number of observations. Examples of a record can be a network packet, network session, a firewall or application log entry. Each of these records consists of observations and are the things measured like IP-address, transport port number, payload, API calls, timing information etc. and are usually called features. A record can contain many of these features. The dimensionality of each record is simply the number of collected features and can by this definition vary between records. The task is to divide **P**, all the records, to one and only one cluster.

[1]Probably based on [Bace, 2000, page 84], found via Google books

In order to be able to cluster one must define the distance between two records. The requirement is a metric as illustrated in figure 2. If all the features are converted into decimal numbers, then Euclidean distance (or more generally Minkowski distance) can be used. It is then recommended to normalize the data first so that attributes with huge ranges won't dominate attributes with smaller ranges. Edit (Levenshtein) distance and hamming distance are other measures that works on strings.

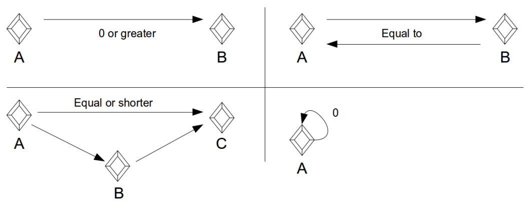


Figure 2: *Criteria for a metric*

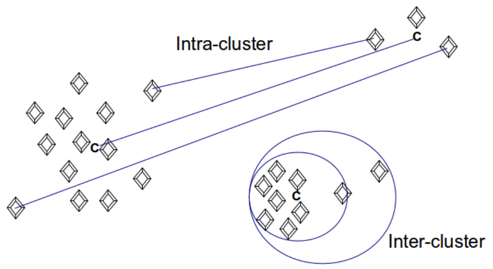


Figure 3: *Different ways to define inter- and intra-cluster distances*

A naive method for clustering is to calculate the distance between every record, a point in a high dimensional space, combine the two closest ones and then repeat until all records (or groups of records) are combined into one. This is called hierarchical clustering and has the downside that the complexity is O(n^2) meaning it is very inefficient.

Non-hierarchical, also called partitional clustering algorithms try to approximate building clusters, often based on some additional parameters and assumptions. The presented method for clustering [Petrovic, 2013] was an algorithm called k-means with the goal of making only two clusters (k=2) using Euclidean distance. One for normal and one for abnormal activities. In order to make timely predictions, records are grouped in intervals and processed in batches. This means a new (**P**) for each run.

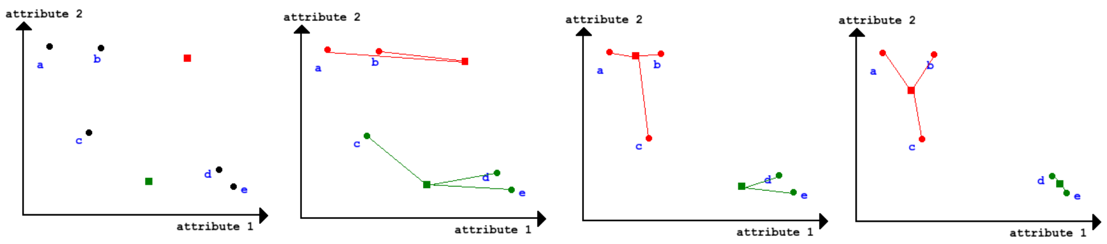


Figure 4: *How k-means work: The red and green dot represent the initial centroids. Each record grabs the closest centroid and pulls it towards it iteratively.*

The inner workings of k-means is shown in figure 4. First step is to choose a number for k and place k centroids within the vector space. Selecting k random of the input vectors is a known strategy for their location. Then the distance such as Euclidean is measured between each vector and these initial centroids, and then assigned to the closest centroid. New centroids are calculated based on an average of the assigned vectors and then the process is repeated until convergence. Measuring the movement of the centroids is a way to determine convergence. The main difficulty of k-means is to decide on a good number of k, and there are some ways to find it by calculating compactness and separation on the clustered data. The result of the algorithm will also depend on initial centroids and thus might end up in a local optimum solutions.

With the assumption that the smallest cluster is the abnormal, the next step is to label the clusters and should be quite easy. Count the number of items and select the lowest number and report on it. Problem is that we know of cases where this assumption breaks. Examples are certain denial of service attacks. The proposed solution was to introduce a new assumption: Massive attacks generates records that are very similar, and as a result end up as very compact clusters. By using cluster quality measures one can make exceptions when one of the two clusters are very compact and by that deduce what cluster is normal versus intrusion. The Davies-Bouldin index was mentioned as a candidate, and it uses a relationship between the inter-cluster and intra-cluster distances. These can be defined in many ways, see figure 3.

2.2 Selected research papers

2.2.1 Intrusion-Detection Model

Looking through research papers in the field, one of the oldest and referred to as the beginning [Eskin et al., 2002] of anomaly detection for IDS is the paper "An Intrusion-Detection Model" by Dorothy E. Denning [Denning, 1987]. It dates back to 1987 written in the times when data was centralized on super computers mostly accessed by terminals, and the proposed hypothesis was that

"security violations could be detected from abnormal patterns of system usage."

Many examples were given, such as unusual number of login tries, login at unusual times or via unusual connection types, unusual amounts of browsing / copying of files or huge aggregated database queries and increased resource usage. The author is well aware that abnormal patterns will emerge from normal activities such as system updates and that new users and users changing working tasks are possible sources for false negatives. In the proposed model, profiles are created for individual users and for groups in order to compare abnormality at different aggregated levels. The conclusion has two very important arguments. First, detecting (abnormal) intrusions is all about finding practical things to monitor that will imply an intrusion. Monitoring low level activity, such as page faults and supervisor calls are not necessarily going to be successful (alone). Secondary, performing anomaly detection is not supposed to replace other security mechanisms such as access control and firewalls.

2.2.2 Geometric Framework

The paper "A geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data" [Eskin et al., 2002] was published in 2002, and explains the problem of getting access to enough training material for supervised learning methods, because it's very expensive to generate. This is where unsupervised methods can help. If the assumption holds that statistical outliers are intrusive, then by sorting out these as candidates for intrusive behavior we might have a method for faster generation of labeled training data.

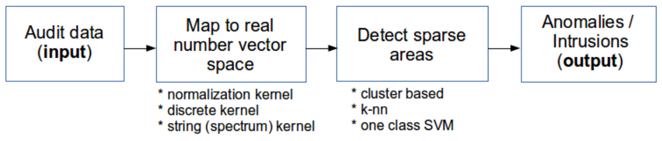


Figure 5: *Simplification of the geometric framework*

The framework as illustrated in figure 5 consists of mapping the input records of application dependent audit streams to a feature vector using real numbers so that the distance between records can be numerically calculated using dot products. Instead of actually transforming the data, distance can be computed directly using kernel functions, mostly for saving memory. Then different types of clustering algorithms are applied to separate out points that are far from the majority. 3 methods were presented: One based on cluster estimation, one based on k-nearest neighbors (k-NN) and one based on a modified support vector machine (SVM). Each record will now be a point in space as far as these algorithms concern.

The idea of the cluster estimation algorithm as illustrated in figure 6 is to set a threshold radius **w** and determine how many neighbors fall withing this radius and then select the ones with the lowest number determined by another threshold. Because this is an exponential problem, a method for approximation is explained: Select a point as a new center for a cluster. If the next point is within a previously defined cluster then add it to every one of them or else create a new cluster center. Continue for all points.

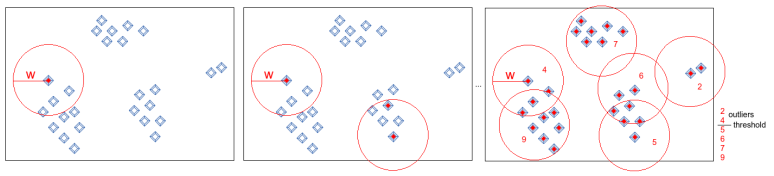


Figure 6: *Clustering estimation algorithm*

The k nearest neighbor (k-NN) algorithm illustrated in figure 7 also got exponential complexity, and the goal is for every point to determine the sum of the distances to the **k** nearest points. In sparse areas this sum will be high and in dense areas it will be low. In order to make the algorithm effective, an approach called canopy clustering, similar to the fixed width clustering is performed. Instead of allowing a point to belong to many clusters, we now only allow membership to one cluster. Finding the **k** nearest points problem is now constrained to comparing the current clusters members. Because a point just outside of the cluster can potentially be closer than any point within, additional checks are performed, and nearby clusters are added to the search space if needed. The point still being that only a small subset of other points are candidates for the nearest neighbors. This process is continues until the sum of **k** distances per point is found. The inner circle of figure 7 illustrates the safe area where it is guaranteed that no other point can be closer than those inside here.

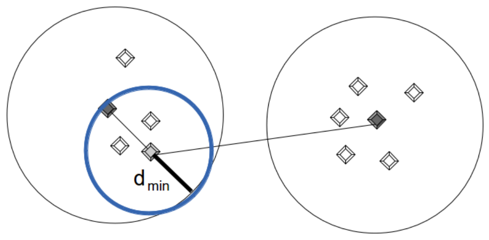


Figure 7: *k-NN algorithm to find sum of the k nearest neighbors*

The last one is a one-class support vector machine (SVM). It's a complicated method with roots in supervised learning where the goal is to find the maximum margin between two classes. As far as I understand, one-class SVM is modified so that it will try to cover as many points as possible while minimizing the area it covers, and thus functions as an anomaly detector.

For testing they used two datasets: The KDD Cup 1999 and a part of the MIT Lincoln Labs DARPA Intrusion Detection Evaluation (1999). KDD Cup is based on a simulated military network with several types of attack. The records consists of network sessions and meta data totaling 41 features both numerical values and discrete categories. The numerical attributes are normalized to the number of standard deviations they are from the mean. The way discrete values are converted is described in a very convoluted way. The main principle would be that different values of a discrete value for two records will result in an additional distance component that depend on how many possible values that discrete value can take. The authors filtered the KDD dataset to make it fit their assumption that there must be way more normal traces compared to intrusive.

The MIT Lincoln Labs DARPA Intrusion Detection Evaluation are system call traces for a Solaris machine, and a subset of these were selected for this test. Here the input records consisted of strings, and a method called spectrum kernel was used to map it into the feature space as required for the clustering algorithms used. It is based on dividing strings into smaller substrings.

For measuring the performance, they show detection rate and false positive rate. Detection rate is the probability of detecting an intrusion. The false positive rate is the probability of classifying a normal event as intrusive. Test data was divided in two parts, one for adjusting the algorithms parameters and the rest for benchmarking the optimized parameters. On the system call dataset, all methods separated all the intrusions, given that success was defined as triggering for at least one of the process traces for each attack, and no false negative were found. For the KDD we see from the ROC curve that the cluster based approximation performs quite well keeping the false negative rate low.

2.2.3 Y-means

The paper presenting Y-Means [Guan et al., 2003] from 2003 tries to extend using k-means and do something about the two weaknesses the algorithm suffers from: The difficulty of determining a value of k, and that the algorithms sometimes end up with empty clusters. They build on a modified version of k-means called H-means+ that solve the problem of empty clusters by moving clusters that end up without members to a new location. This location is determined by for each cluster with members find the distances between the cluster center (centroid) and each member. Then the point furthest away is taken as the new location.

The Y-means algorithm first performs an initial clustering based on an initial choice of k. Empty clusters are replaced as described and the algorithm keeps clustering until no empty clusters exist. They explain a method for splitting and joining clusters based on a confidence threshold of 2.32 standard deviations from the mean of the distances between points sharing a centroid and the centroids location. The idea is that when a cluster is evaluated, and there are points having a distance greater than this 2.32 standard deviations from the mean, then the furthest point will be reassigned as a new centroid. The opposite event occurs when two clusters have points within the confidence threshold. The algorithm also appear in 2010 in [Ghorbani and Onut, 2010] by the same author

The evaluation is performed using the KDD Cup 99 dataset, and the authors manage to define the False Alarm Rate to be a constant.

> "FAR equals the number of normal instances divided by the number of normal instances in the data set."

which is obviously a mistake. The authors show that the Y-means method is able to in average get the optimal number of clusters independent of number of starting clusters. They brag of best performance being 89.89% detection rate and 1.00% false alarm rate, and in the next sentence they get 82.32% / 2.5% after more learning on randomly selected records of the dataset.

2.2.4 pfMAFIA

In 2005 [Leung and Leckie, 2005] explains that a lot of methods have been proposed (Like using principle component analysis (PCA) and self organizing maps (SOM)) in this field, but the problem is we need faster algorithms that scales to very high dimensional data. They propose a clustering system called pfMAFIA. The main idea is to use density and grid based methods for estimating dense regions. The method will not cover all data but is said to guarantee more than 95% coverage. pfMAFIA builds on existing methods, and does some modifications to them. For all records containing attributes, each attribute is scanned for range and then sorted into bins in order to build a data structure similar to a histogram.

This is performed in steps based on some stopping criteria, meaning these bins will be merged in steps. If the distribution for this attribute (or dimension) is constant, then the expected outcome would be only one "bin".

Binary and discrete attributes are mapped to integers and treated as continuous values. The number of initial bins is also adjusted depending on the range of values an attribute can take for performance. The important part of this method is the usage of the monotonicity principle:

"if a k-dimensional unit is dense, then so are its projections in (k - 1) dimensional space"

So, each attributes value is replaced by the corresponding bin it fall under, the the task of determining dense regions is translated into a data mining problem trying to figure of what combinations of bins are most frequent. This is solved by a modified "frequent pattern growth" algorithm.

The testing of this method is interesting as their results are visibly much worse than methods compared to based on the ROC curve (the detection rate vs. false alarm rate). They then present the area under ROC curve as a way to compare results, and that makes their result look better in comparison.

2.2.5 Song et al.

A fairly recent paper [Song et al., 2013] from 2011 also looks at getting rid of the parameters of a previous method they proposed. They state it is very important to separate the different types of normal data (network traffic) before outliers are looked for. They therefore first clusters the records in some number of clusters and then performs one class SVM on all of the clusters to locate the outliers.

The authors cite Mukkamala et al. for determining redundant features in the KDD Cup 99, and because of this weakness they instead use captured data from honeypots deployed inside the Kyoto university and they assume all traffic of these are intrusive. A mail server were set up and even though it might have been attacked, it was still claimed to be a very small amount and thus classified as benign for testing purposes. They also report on having higher detection rates and lower false alarm rates than other existing methods.

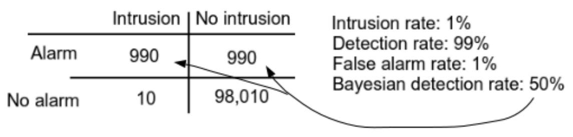
2.3 Performance

From reading papers in the field, it is easy to verify that access to good training sets are difficult to get by, and this was actually one of the reasons for promoting unsupervised methods such as clustering in the first place. Still we got the problem that in order to establish a measurement of the performance of these methods we still rely on labeled data. The KDD Cup 99 is very popular, and even though it allows for comparison between methods, it's simply not reliable to base the performance of a real world evolving problem on one dataset that is now also getting old.

One of the assumptions for using clusters is that the number of intrusions compared to normal event must be greatly in favor normal activity. Adjusting datasets to conform with this assumption is common. Because of this it is also important to discuss the "Base-rate fallacy" [Axelsson, 1999]. This paper explains that for such skewed distribution, what we need to focus on is not the detection rate, but rather the false alarm rate. The Bayesian detection rate is the probability that a triggered alarm is actually an intrusion and expressed in figure 8 and exemplified in figure 9.

$$P(\text{intrusion} \mid \text{alarm}) = \frac{P(\text{intrusion}) \cdot \overset{\text{detection rate}}{P(\text{alarm} \mid \text{intrusion})}}{\underset{\text{very small}}{\underset{\uparrow}{P(\text{intrusion})}} \cdot \underset{\text{detection rate}}{P(\text{alarm} \mid \text{intrusion})} + \underset{\text{very large}}{\underset{\uparrow}{P(\text{not intrusion})}} \cdot \underset{\text{false alarm rate}}{P(\text{alarm} \mid \text{not intrusion})}}$$

Figure 8: *The Bayesian detection rate: The base-rate fallacy*



	Intrusion	No intrusion
Alarm	990	990
No alarm	10	98,010

Intrusion rate: 1%
Detection rate: 99%
False alarm rate: 1%
Bayesian detection rate: 50%

Figure 9: *The Bayesian detection rate: Confusion matrix*

Because of the very large factor, the false alarm rate will totally dominate the outcome of this formula, unless the false alarm rate is also very small, in the order of 0.001% according to the assumptions made in that paper. Most of the results summarized earlier range from 1% up to 10+% which is actually extremely poor.

3. Conclusion

In this report we have looked at some clustering methods and how they can be applied to problems regarding detecting intrusion in computer systems based on the assumption that intrusive behavior result in outliers in terms of observable evidence. Important steps in the process are:

- What features (properties) to monitor
- Normalization, encoding to numerical and defining distance between two observations
- Choice of method for detecting / estimating outliers
- Deciding what clusters, if any, are intrusive

It is easy to find papers describing clustering methods, and to some degree ideas of how observations are encoded. Many papers simply use KDD Cup 99 and does not go much into details of what would be beneficial to monitor. The quality of the algorithms very much depend on their input, so I find this a bit strange. With limited data sets available, it looks like new methods to a high degree get good benchmarks as a result of overfitting their algorithms for the problem.

Other important takeaways is the importance of getting the basic protection mechanisms to work like firewalls and access policies before one consider intrusion detection automation. Lower the noise so that abnormal things actually stick out. In misuse rule based IDS you know what the attack is because of what was triggered. In anomaly you only know that something is not right and requires additional work by experts.

References

[Axelsson, 1999] Axelsson, S. (1999). The base-rate fallacy and its implications for the difficulty of intrusion detection. In *In Proceedings of the 6th ACM Conference on Computer and Communications Security*, pages 1–7.

[Bace, 2000] Bace, R. G. (2000). *Intrusion Detection*. Macmillan Technical Publishing, 201 West 103rd Street Indianapolis, IN 46290 USA.

[Beigh and Peer, 2012] Beigh, B. M. and Peer, M. A. (2012). Intrusion detection and prevention system: Classification and quick review. *ARPN Journal of Science and Technology*, 2(7):4 – 14.

[Denning, 1987] Denning, D. (1987). An intrusion-detection model. *Software Engineering, IEEE Transactions on*, SE-13(2):222–232.

[Eskin et al., 2002] Eskin, E., Arnold, A., Prerau, M., Portnoy, L., and Stolfo, S. (2002). A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In *Applications of Data Mining in Computer Security*. Kluwer.

[Ghorbani and Onut, 2010] Ghorbani, A. and Onut, I.-V. (2010). Y-means: An autonomous clustering algorithm. In Graña Romay, M., Corchado, E., and Garcia Sebastian, M., editors, *Hybrid Artificial Intelligence Systems*, volume 6076 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin Heidelberg.

[Guan et al., 2003] Guan, Y., Ghorbani, A., and Belacel, N. (2003). Y-means: a clustering method for intrusion detection. In *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on*, volume 2, pages 1083–1086 vol.2.

[Leung and Leckie, 2005] Leung, K. and Leckie, C. (2005). Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science - Volume 38*, ACSC '05, pages 333–342, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

[Petrovic, 2013] Petrovic, S. (2013). Anomaly detection systems. Lecture slides in IMT4741.

[Song et al., 2013] Song, J., Takakura, H., Okabe, Y., and Nakao, K. (2013). Toward a more practical unsupervised anomaly detection system. *Information Sciences*, 231(0):4 – 14. Data Mining for Information Security.