Nonparametric Semi-Supervised Learning for Network Intrusion Detection: Combining Performance Improvements with Realistic In-Situ Training

Christopher T. Symons
Computational Sciences and Engineering
Oak Ridge National Laboratory
Oak Ridge, TN 37831
symonsct@ornl.gov

Justin M. Beaver
Computational Sciences and Engineering
Oak Ridge National Laboratory
Oak Ridge, TN 37831
beaverjm@ornl.gov

ABSTRACT

A barrier to the widespread adoption of learning-based network intrusion detection tools is the in-situ training requirements for effective discrimination of malicious traffic. Supervised learning techniques necessitate a quantity of labeled examples that is often intractable, and at best cost-prohibitive. Recent advances in semi-supervised techniques have demonstrated the ability to generalize well based on a significantly smaller set of labeled samples. In network intrusion detection, placing reasonable requirements on the number of training examples provides realistic expectations that a learning-based system can be trained in the environment where it will be deployed. This in-situ training is necessary to ensure that the assumptions associated with the learning process hold, and thereby support a reasonable belief in the generalization ability of the resulting model. In this paper, we describe the application of a carefully selected nonparametric, semi-supervised learning algorithm to the network intrusion problem, and compare the performance to other model types using feature-based data derived from an operational network. We demonstrate dramatic performance improvements over supervised learning and anomaly detection in discriminating real, previously unseen, malicious network traffic while generating an order of magnitude fewer false alerts than any alternative, including a signature IDS tool deployed on the same network.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: General—*security and protection*; I.2 [**Artificial Intelligence**]: Applications and Expert Systems

Keywords

machine learning, network intrusion detection, nonparametric, semi-supervised

AISec'12, October 19, 2012, Raleigh, North Carolina, USA.

1. INTRODUCTION

Learning systems are becoming increasingly pervasive in network intrusion detection research and practice. These systems strive to *generalize* knowledge about network transactions the way a human would and apply it to previously unseen network transactions in order to identify malicious traffic. Although there are many examples of the potential promise of these methods, a variety of factors combine to make both research advances and practical deployment of machine learning systems difficult in the intrusion detection domain. Many of these factors are related to the lack of available labeled data on operational networks. Attacks captured in the wild are rarely made available, and even then, they cannot be directly leveraged to learn models that will be applied to different networks. The critical nature of the *i.i.d.* assumption (that all points used in training and to which the model will be applied are pulled independently and identically from the same distribution) underlying almost all machine learning methods, is often neglected due to the belief that training a model *in situ* (in this context we mean where it will be deployed) is too costly or impractical. Prior results on synthetic datasets have utilized hundreds of thousands or millions of training examples. Even previous semi-supervised learning experiments in this domain have utilized thousands of labeled examples. Obtaining examples in a new network is typically considered too costly to support models that require such large numbers of labeled events, particularly when they aren't guaranteed to dramatically outperform alternate methods.

Guarantees on the generalization performance of machine learning approaches are based on theoretical error bounds that do not apply if the assumptions of the method do not match the reality of its utilization. In other words, when deploying a learning system in an environment with its own idiosyncrasies, and keying off of network statistics and other variables that are intertwined with the noise peculiar to the network, the standard assumptions upon which the learning algorithm depends need to be recognized. In nearly all cases, the theoretical performance guarantees depend on the *i.i.d.* assumption. In practice, this means that effective learning systems would ideally always be trained *in situ*, or using examples from the network environment in which they are deployed. Therefore, discriminative learning based on known attack data is potentially limited by the cost inherent in identifying and/or generating real attacks in a new environment. In addition, there is some cost involved in ensuring

that normal traffic in an existing network is indeed innocuous. There are often model-specific assumptions that must be recognized as well, and many model types are not well suited to the intrusion detection domain.

Semi-supervised learning methods [10], which use unlabeled data to augment the learning process, have been shown in many domains to achieve better generalization performance with far fewer labeled examples than would otherwise be required.A practical question that arises is the following: Can we learn an effective intrusion-detection model using a small number of labeled examples? If so, the costs associated with training in situ become less prohibitive. Having a penetration testing team perform attacks on a network for a few hours as opposed to weeks or months, or manually verifying dozens of network transactions as opposed to tens of thousands, becomes a much more manageable requirement to place on an organization for proper instrumentation of a system.

In this work, we carefully select a nonparametric approach to semi-supervised machine learning that has several implicit assumptions that match the data generation process in the network intrusion detection domain. We justify the use of this model both theoretically and experimentally and use this algorithm to provide strong evidence on data derived from large-scale operational network data [29] that we can indeed build very effective models using a small number of labeled examples. In addition to comparing our model to multiple supervised and semi-supervised models, we compare our results to published results using sophisticated anomaly detection methods and to the output of a signature-based intrusion detection system (IDS) applied to the same data. An ability to generalize very effectively based on few observations is confirmed, demonstrating clear potential to augment current IDS tools with very simple features and very realistic training requirements in a way that can potentially provide strong alerting coverage against unknown attacks with trivial false positive rates.

The experimental analysis uses data from Kyoto University that was recently made available to the public (see [29]). While this is a carefully curated, valuable new resource, it has limitations, and therefore, we only claim to provide strong evidence that real tools with these characteristics are currently viable. Our experiments are carefully designed to demonstrate true generalization performance on *unknown* attacks using minimal training sets, and the results on the operational data show that we can catch nearly all previously unseen attacks with a false positive rate that is an order of magnitude lower than any of the alternatives (including the signature IDS, which cannot identify previously unseen attacks).

2. BACKGROUND

In this section, we cover some background on the use of machine learning for intrusion detection, and we describe semi-supervised learning for the uninitiated.

2.1 Machine Learning in Intrusion Detection

Most operational network intrusion detections systems rely on very specific rules, or *signatures*, to identify potentially malicious traffic. Human experts generate the signatures after they have extensively analyzed an attack and determined the attack's indicative bit patterns and conditions. While signatures are effective at identifying a specific instance of an attack, developing them is a time-consuming and manually intensive process, during which the network remains vulnerable. Furthermore, simple variants of the attack on which the signature is based will often not trigger the signature pattern. As the frequency and diversity of attack attempts rise, organizations are finding it increasingly difficult to keep pace in developing the raw number of signatures required. A different process for attack analysis is necessary if computer network defense systems are to remain effective.

The intrusion detection research community has responded to the problem of signature development latency by exploring machine-learning methods capable of learning the discriminating characteristics of malicious traffic from exemplar network transaction data. The collective works cover a broad range of techniques and are applied in various architectures in order to propose an optimum approach to network traffic classification. See [14, 30] for reviews of the field. Despite this significant body of work, machine-learning approaches, and in particular supervised learning systems, are sporadically deployed compared with the less sophisticated signature-based approaches. We attribute this phenomenon to both a low confidence in the reported performance of machine-learning-based intrusion detectors, and a poor understanding of how to operationally field them.

2.1.1 Contrast with Anomaly Detection

Outside of the machine learning community, the phrase *anomaly detection* [9] is often used interchangeably with and regularly confused with machine learning. For people who understand all of the subfields that lie within these domains, this is not an issue, but a large portion of the community involved in cyber security is unaware of differences. One early contributing factor to the intermixing of these terms is due to a tendency to classify unsupervised learning, or clustering, as a form of machine learning. Another factor is the use of machine learning techniques to solve anomaly detection problems, e.g. where clusters are taken as ground truth for learning classification models. However, we submit that there is a fundamental distinction between the two areas of study based on theoretical foundations of machine learning that have generalization performance as a critical concept. Thus, while anomaly detection can be any mechanism that looks for unusual patterns, machine learning looks to generalize an *expert-defined* distinction. Therefore, on a fundamental level, it is perfectly natural in machine learning to build a model purposely designed to distinguish between malicious and benign network traffic and have optimal performance on previously unseen events. On the other hand, such a problem definition has very little connection with a general definition of anomaly detection, since attacks aren't necessarily anomalous and normal behaviors often are.

In light of the above, it is important to point out that this paper is written in a general context that differentiates learning from anomaly detection, in particular, based on the use of classification labels and the emphasis on generalization performance in machine learning. Thus, real ground-truth labels are a necessary component to the model building process that we hope to address. The downside to using labels is the cost of obtaining them, while the upside is the ability to steer a model in a desired direction. In addition, when we talk about using unlabeled data to augment label information via semi-supervised learning, we do so based on a notion of compatibility (see section 3.4) that uses concepts like reg-

ularization to achieve better generalization performance on a classification task defined by the labeled data.

In [28], the authors provide a good summary of the challenges inherent in the application of machine learning to intrusion detection, but the problem being addressed is still defined to be "outlier detection." In other words, one of the points being argued is that since the problem being solved is anomaly detection, machine learning techniques, which operate well on notions of similarity, are challenged. Our view is that normal traffic can often be completely different from anything previously seen on the network. We also contend that previously unseen attacks may not necessarily appear to be anomalous in the originally defined feature space, yet have distinguishing characteristics such that they are more similar to known attacks than normal traffic. If these assumptions are reasonably accurate, then outlier detection is not the problem we want to solve. Instead, we operate on the assumption that an expert-derived feature space can capture information that allows previously unseen attacks, whether anomalous or not, to be identified as sharing certain distinguishing characteristics with known attacks. The generalization performance we observe when detecting previously unknown attacks on operational data (see section 4) offers strong evidence that new attacks do indeed resemble known attacks in ways that allow them to be distinguished from normal traffic, even on data where anomaly-detection and signature-based systems struggle to reliably discriminate.

The problem becomes one of finding the right view through which the desired distinction can be seen. Therefore, our approach is to solve a classification problem where experts have provided a small number of ground truth labels on the target network. Our goal is to show the power of using a model whose assumptions very closely match the data generation process in this domain (see section 3.2). We use the availability of the labeled operational data in the Kyoto2006+ dataset [29] to help demonstrate that the label requirements for a machine learner can be made small enough, using current methods, to realistically deploy effective learning-based intrusion detectors.

2.1.2 Data Limitations

The lack of confidence that exists in academic evaluations of machine-learning network intrusion detectors can be traced to a shortage of publicly available data. Organizations typically keep their network intrusion data hidden to prevent publicizing any vulnerability. As a result, the majority of academic studies present results that explore a singular approach tailored to a specific environment, and are difficult to verify or validate more generally. A significant gap in the literature that applies machine-learning techniques to the network intrusion detection problem is the absence of a relevant network intrusion data set that can be used as a basis for comparison. While the 1999 KDD cup "classification task" data [21] provided an initial surge of interest in machine-learning-based intrusion detection, the background traffic was simulated and the data no longer accurately represents modern network traffic. The lack of other relevant, public labeled data sets has severely limited the exploration of machine learning methods in network intrusion detection. The release of the Kyoto2006+ dataset [29], which captures metric sets associated with real operational network flows, is therefore a very promising step toward more accessible research in this area. We summarize some of the most relevant characteristics of this dataset in section 4.

2.2 Semi-supervised learning

Semi-supervised learning [10] is generally defined as any learning method that uses both labeled and unlabeled data during the model discovery process. Methods for incorporating the unlabeled information can vary. Some methods include the use of data-dependent priors and low-density separation, but the most common approach is graph-based [10, 16]. Graph-based methods are typically designed based on the assumption that the data naturally occur on an underlying manifold, such that the true degrees of freedom of the problem can be discovered using unlabeled data. In essence, the intent is to find structure in the ambient space that can be exploited to constrain the search for a good model.

A central construct in many methods is the graph Laplacian [12]. A graph is constructed to represent a manifold (or densely populated region of interest in the ambient space), and the graph Laplacian facilitates the discovery of a low-dimensional space that is smooth with respect to this graph. In semi-supervised learning the goal is to augment learning through the use of unlabeled samples, so the unlabeled data is often used to find a low-dimensional space on which learning via the labels can be more effective.

There are many other approaches to semi-supervised learning. One that is particularly noteworthy for its ability to use the labels in a robust manner is the predictive structure framework of [2]. Their approach is based on multi-task learning and attempts to find structure that overlaps many prediction problems that are formulated in such a way that they always have labels. One drawback to the approach is that the framework, as described, requires a different kind of domain expertise in terms of the construction of the formulated tasks.

2.3 Semi-Supervised Intrusion Detection

Semi-supervised learning has begun to be explored in intrusion detection. In [11], the authors explore the use of transductive spectral methods and Gaussian random fields on the 1999 KDD Cup dataset [21]. The transductive approach achieves the best results in that study, but unfortunately the use of transductive methods is not practical in real-time systems. Although it is probably safe to assume that an out-of-sample extension would not suffer a major drop in performance, the results only support incremental improvement over supervised methods, and the authors lament that they do not reach a level of performance that would be valuable in practice. For example, although the test setup is different, the anomaly detection techniques used in [1] appear to achieve significantly better results on the same dataset.

In [18], semi-supervised concepts are explored, but in a very untraditional manner, involving partially observable markov decision processes (POMDPs), an area of reinforcement learning that suffers from scalability issues [27]. The method is intended to be a proposed framework in which to place intrusion detection. As such, it has merit, but it does not make significant strides toward practical usage. Mao et al. [22] take an interesting approach based on multi-view, semi-supervised learning and active learning, which requires an interactive process with the user, and apply it to the

KDD Cup data. They show improvement over their baseline, which is single view learning without active learning, but the amount of labeled data used is still very high and the number of false positive alerts remains in an unusable range.

3. LEARNING METHODS

Parametric methods in machine learning constrain models to a certain functional form defined by the parameter space. In real-world problems, it is often the case that an appropriate functional form is not known. Nonparametric techniques, on the other hand, offer much more flexibility to represent complicated models. While nonparametric methods almost always include some parameters for the sake of tractability, the core philosophy underlying these methods is that the complexity of the model is allowed to increase with the size of the data [23]. Because the use of some parameters is often inevitable, it is common to refer to some of these methods as *semi-parametric*. The semi-supervised methods that we apply in this paper are non-parametric, which allows greater model flexibility while trying to avoid imposing unrealistic constraints on the form of the model.

For experimental comparison with supervised methods, we also use a linear Support Vector Machine (SVM) and a maximum entropy classifier from the Minorthird machine learning software library [13].

3.1 Graph-based classifier

In this section we focus on the use of Laplacian Eigenmaps for semi-supervised learning [6, 7]. Our main focus will be on the Laplacian Regularized Least Squares algorithm in section 3.2 We use the Laplacian Eigenmaps as an alternate nonparametric semi-supervised learner in our experiments. In particular, comparison with this method is useful because both semi-supervised methods use the graph Laplacian, which is described below.

In our use of the Laplacian Eigenmap approach, we first construct a nearest neighbor graph using the six nearest neighbors based on cosine similarity. Unlike some nonlinear dimensionality reduction methods, the use of Laplacian Eigenmaps does not automatically suggest the size of the new space. Therefore, we retain a basis size that is twenty percent of the number of labeled data points used to reflect the suggestions from [7]. Note that in addition to the nonparametric nature of the nearest neighbor graph, the number of dimensions is a reflection of the labeled data size. Thus, in this case, complexity of the model can grow with the size of the unlabeled data through the graph Laplacian, and it can also grow with the size of the labeled data.

The normalized graph Laplacian is a matrix defined as follows:

$$\mathcal{L}(u,v) = \begin{cases} 1, & \text{if } u = v \text{ and } d_v \neq 0 \\ \frac{-1}{\sqrt{d_u d_v}}, & \text{if } u \text{ and } v \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where u and v are vertices in the graph, d is the degree (number of incident edges) of a vertex, and adjacency refers to a neighboring connection in the graph. We use the unnormalized form given below:

$$L(u,v) = \begin{cases} d_v, & \text{if } u = v \\ -1, & \text{if } u \text{ and } v \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Using the eigenvalues and associated eigenvectors of these positive, semi-definite, symmetric matrices provides a method for discovering dimensions that are smooth with respect to the graph that defines it. If the graph varies smoothly with respect to the target problem (i.e. examples from different classes or clusters are rarely linked, similar examples from the perspective of the target problem are linked, etc.), then it can be used to represent a manifold. The Laplacian of the graph can then be used to find a space that roughly represents that manifold. The eigenvector associated with the smallest non-zero eigenvalue is smoothest with respect to the graph, such that points connected in the graph will be close together in the dimension defined by said eigenvector. This smoothness with respect to eigenvector-defined dimensions decreases as you progress to the larger eigenvalues.

Another useful property of the eigensystem is the fact that the number of zero-value eigenvalues is equal to the number of connected components in the graph. In addition, an eigenvector will not involve more than one component of the graph. Thus, after counting the number of connected components, which is an $O(n)$ operation, you need to retain at least that many dimensions in a new space in order to distinguish between all points after they are mapped.

For one of the semi-supervised test models, we rely upon dimensionality reduction as described above. Once the dimensionality reduction is achieved, an initial transductive model is constructed. First, we construct a simple classifier in the new space in the same manner as the approach in [7], in which the coefficients for the new dimensions are set by minimizing the sum of squared error on the labeled data. In other words, the weights of our new dimensions are given by the vector $\mathbf{a}$ in the following:

$$a = (EE^T)^{-1}Ec \quad (3)$$

where c is a vector representing the class labels, λ_k, v_k are the k-th eigenvalue and eigenvector, respectively, the entries of E are $\lambda_k v_{i,k}$, i is the index of the labeled point in the matrix, and k is the index in the new low-dimensional space; i.e. the k-th eigenvalue and eigenvector provide the mapping into the new space for labeled point i. The number of connected components in the graph is determined in order to eliminate the zero-valued eigenvalues, and then the mapping starts with the next eigenfunction.

Since the Laplacian Eigenmap approach is inherently transductive, it only creates a mapping for an unlabeled example if it was part of the set used for graph construction. This means that applying a method transductively would involve solving the eigenvalue problem all over again for any new point or set of points, which would be impractical for most purposes, and in particular, for intrusion detection in real time. For a nonparametric out-of-sample extension that allows efficient application to new points, we utilize the Nystrom Formula as described in [24]. The method has been shown (and for the most part verified via our own experiments) to provide inductive classification results with no significant difference in accuracy from the transductive application. It simply uses the Laplacian matrix as a data-dependent Kernel function K_D in the following formula in

order to map a new point into each dimension k of the new decision space:

$$f_k(x) = \frac{\sqrt{n}}{\lambda_k} \sum_{i=1}^{n} v_{ik} K_D(x, x_i) \quad (4)$$

where n is the size of the original dataset, and λ_k, v_k are the k-th eigenvalue and eigenvector, respectively.

3.2 Laplacian RLS/Bayesian Kernel Model

The main semi-supervised model that we focus on in this paper is interesting from multiple perspectives. In fact, it is possible to arrive at the same functional form for this model based on two completely different derivations. In other words, this model represents both the Laplacian Regularized Least Squares (Laplacian RLS) model in [8] and the Bayesian Kernel Model in [19, 20, 25] with a Dirichlet process prior. This is relevant to our discussion because we hope to use models whose assumptions more realistically match the realities of the data. And we can argue that both of these derivations are in tune with how we hope to shape (or avoid shaping) the model.

In the case of the Laplacian RLS [8], we are using unlabeled data as a graph-based regularization term, which essentially means that we can use as much unlabeled data as we want to penalize models that would assign points among the unlabeled data that are extremely close together in our expert inspired feature space as belonging to different classes. This makes sense as long as the features are relatively important to the problem domain. We believe this to be true a priori due to the fact that they were derived by experts.

In the case of the Bayesian Kernel Model [19, 20, 25], a model is estimated by selecting from among functions in the reproducing kernel Hilbert spaces (RKHS) induced by the chosen kernel. We would like to assume that we have a smooth function that we want to represent. In this case, we can look at our kernel as data that falls on a smooth manifold; i.e. that points in the original space actually vary along a dense manifold that cuts through that space. However, we also believe there are other unknown random processes generating the points that we see on the smooth manifold. Because each event may be generated based on its own random process, we don't want to restrict the form of each of these processes. In addition, we don't want to restrict the possible number of processes that could be generating the events we observe. This is a typical case in which a Dirichlet process prior might be used. It also makes sense because while we hope that the target function we are trying to learn lies in a dense region that cuts through the original feature space, it also allows us to represent each event as being generated by its own random process.

We will see that these assumptions can also lead us to the same functional form as the Laplacian RLS. The derivation can be found in [19, 20, 25]. The relevant Bayesian kernel derivation is based on integral operators. The form that is used in [19] is the following:

$$f(x) = \int k(x, u) d\gamma(u) = \int k(x, u) w(u) dF(u) \quad (5)$$

F is the unknown distribution of the kernel knots, u, where a knot is a data point on the manifold. In essence this means that F can be set to correspond to the marginal distribution of the data, X.

Given a fixed sample from an uncertain distribution F in the Dirichlet process (DP) model, the posterior is the following Dirichlet process [19]:

$$F|X_n \sim DP(\alpha + n, F_n), F_n = (\alpha F_0 + \sum_{i=1}^{n} \delta_{x_i})/(\alpha + n) \quad (6)$$

In [19], we see that if we want to predict the value of a new point, x, based on our sample from F, i.e. based on our labeled and unlabeled training data, then we want the following:

$$E[f|X_n] = a_n \int k(x, u) w(u) dF_0(u) + n^{-1}(1 - a_n) \sum_{i=1}^{n} w(x_i) k(x, x_i) \quad (7)$$

where $a_n = \alpha/(\alpha + n)$.

Then, assuming an uninformative prior, they take the limit $\alpha \to 0$ to get the following representer form:

$$\hat{f}_n(x) = \sum_{i=1}^{n} w_i k(x, x_i) \quad (8)$$

Thus, according to [19, 20, 25], when the uncertainty about the probability distribution function for the data, X, is expressed using a Dirichlet process prior, then the function f can be approximated by the following formula over labeled and unlabeled examples.

$$\hat{f}(x) = \sum_{i=1}^{n} w_{n,i} K(x, x_i) + \sum_{i=1}^{n_m} w_{n+n_m, n+i} K(x, x_i^m) \quad (9)$$

This results in the exact same functional form as that derived in [8]. And in fact, the graph Laplacian over the observed data can then approximate the Laplacian on the manifold by solving the following.

$$\hat{f}(x) = \text{argmin}_{f \in \mathcal{H}_K} \left[\frac{1}{n} \sum_{i=1}^{n} V(f(x_i), y_i) + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(n + n_m)^2} f^T L f \right] \quad (10)$$

where L is the Laplacian derived from the data and $f = \{f(x_1), ..., f(x_n), f(x_1^m), ..., f(x_{n_m}^m)\}$. γ_A and γ_I are parameters that control the amount of regularization in the ambient space and intrinsic space, respectively.

3.3 Laplacian RLS Model Implementation

So, once again, we can use a method of semi-supervised learning using the graph Laplacian. In our experiments, we use the unnormalized form of the graph Laplacian here as well (Equation 2).

The output function that is learned is the following:

$$f(x) = \sum_{i=1}^{l+u} \alpha_i K(x_i, x), \quad (11)$$

where K is the $(l+u) \times (l+u)$ Gram matrix over labeled and unlabeled points, and α is the following learned coefficient vector:

$$\alpha = (JK + \gamma_A lI + \frac{\gamma_I l}{(l+u)^2} LK)^{-1} Y, \quad (12)$$

with L being the Laplacian matrix described above, I being the $(l+u) \times (l+u)$ identity matrix, J being the $(l+u) \times (l+u)$ diagonal matrix with the first l diagonal entries equal to 1 and the rest of the entries equal to 0, and Y being the $(l+u)$ label vector, $Y = [y_1, ..., y_l, 0, ..., 0]$. See [8] for details.

This method does have two parameters that control the amount of regularization. For all of our experiments, we use the following parameters, as suggested for manifold regularization in [8]: $\gamma_A l = 0.005$, $\frac{\gamma_I l}{(l+u)^2} = 0.045$.

3.4 Additional Theoretical Context

Although a more thorough theoretical analysis is beyond the scope of this paper, we point the interested reader to existing theoretical work related to error bounds on the generalization performance of these methods. In [5, 4] semi-supervised learning is analyzed as a notion of compatibility, χ. The notion of compatibility is based on finding a model that has a low *unlabeled error rate*. In the case of a graph regularization approach like the Laplacian RLS model, this can indicate that the function being learned *agrees with the graph* and would seldom label two nodes sharing an edge with different class labels. Thus, the unlabeled data help guide the model choice through the graph by penalizing models that do not agree with it.

4. EXPERIMENTAL RESULTS

We use the Kyoto2006+ dataset [29] for all of the experiments in this section. The dataset covers nearly three years of network traffic through the end of 2008 over a collection of both honeypots and regular servers that are operationally deployed at Kyoto University. The data is provided in the form of observations and statistical features that characterize terminated connections. We only use the first 14 features since any system would have access to the information required to construct these features, whereas the additional features are unlikely to be available. The fact that the features are pre-calculated allows for more accurate comparison of different model types, but it unfortunately restricts the possible features to those provided.

Before using the data, we convert categorical features to binary, and normalize all numeric data using a Softmax scaling approach (with $r = 1$), which is purported to retain the most information [26].

Unfortunately, the dataset does not provide information on specific attack types. Therefore, we are unable to take advantage of a cost-sensitive learning scheme, and we are unable to determine how well we are doing with regard to differentiating attack types or prioritizing alerts. Moreover, there is a good deal of suspected labeling error. Even though the number of errors is likely tiny compared to the size of the dataset, this is an important point (see [29] for a detailed description of the dataset).

The dataset essentially represents a two-class classification problem, where the classes represent malicious traffic and non-malicious traffic in a network. There is a distinction made between *known* and *unknown* attack types, which we leverage in some of our experiments to test the ability to generalize knowledge to previously unseen attacks. *Unknown* attacks are defined as those that were not flagged by the signature IDS, but for which the Ashula tool detected shellcodes. The only packet information available to our models is the number of bytes sent by the source and destination.

4.1 Comparative Analysis

All tests in this section are performed across the test data used in [17], which comprises 12 days of traffic pulled from the last six months of 2008. Table 1 shows the initial results of training two supervised learners from the Minorthird library [13], a linear Support Vector Machine (SVM) and a maximum entropy learner, using a full day's labeled data from January 1, 2008. For comparison, we also display the alerting results from the intrusion detection system (IDS) that are included in the dataset, and we list the results from [17], which employed an anomaly detection approach using multiple classifiers trained over 10 million training examples.

Next, we compare the semi-supervised learners to the supervised learners using very small labeled datasets. The semi-supervised learners are the Laplacian Eigenmap (LEM) and the Laplacian Regularized Least Squares (RLS) algorithms described above. The results are shown in Table 2. Subsets of 100 labeled examples and approximately 3000 unlabeled examples from Jan. 1, 2008 are used for training, and testing is performed across the same test data as above. There were 111,589 examples (terminated connections) on January 1, 2008. The classification results are averaged over 10 random selection of the labeled data. We first randomly select 100 examples as our labeled training set and retain the rest as unlabeled examples for use by the semi-supervised learners. However, we also remove redundancy through an approximate similarity measure by hashing the examples based on label value, binary feature values, and 10% ranges of the normalized numeric feature values. This leaves an average of 56.6 labeled examples per experiment, with a high of 69 and a low of 19. It also preserves approximately 3000 unlabeled examples per experiment. We report the average recall, false positive rate, and area under the ROC curve, which is a plot of the tradeoff between false positive rate and recall as the decision threshold of the binary classifier is varied (i.e. the AUC score, see [15] for an interesting discussion of this measure). Keep in mind that we purposely restricted the number of labeled examples to an extreme in order to demonstrate the viability of training such models in their deployment environments.

4.2 Training on Known to Catch Unknown

Of particular interest is the ability to catch previously unobserved and unknown attacks after training on a small or reasonable number of known attack types. Because the Kyoto2006+ dataset [29] differentiates between known and unknown attacks, we can test this ability directly. In Table 3, we examine the ability of the Laplacian RLS learner to catch unknown attacks after being trained on normal traffic and known attacks only. The setup is the same as before using data from Jan. 1, 2008, such that the results are averaged over 10 random selections of the labeled data. Each set has 100 labeled data points total to begin with, thus after eliminating redundancy, we observe a combined total of under 70

Table 1: Reported IDS results, multi-classifier anomaly detection results, and results of using all (111,589) labeled examples from Jan. 1, 2008 for supervised learning. Testing is performed across the same test data as in [17], which comprises 12 days of traffic pulled from the last six months of 2008. *The signature IDS alerts are recorded in the dataset.

Classifier	Recall	False Positive Rate	AUC Score
Signature IDS*	0.09004	0.01619	N/A
Anomaly Detection [17]	0.8093	0.0590	N/A
Maximum Entropy	0.77292	0.02059	0.72044
Linear SVM	0.98952	0.03528	0.96295

Table 2: Classifier comparison using small training sets of fewer than 100 labeled examples and approx. 3000 unlabeled examples from Jan. 1, 2008. Testing is performed across the same test data as in [29], which comprises 12 days of traffic pulled from the last six months of 2008. Results are averaged over 10 random selections of labeled examples.

Classifier	Recall	False Positive Rate	AUC Score
Maximum Entropy	0.77292	0.02059	0.72044
Linear SVM	0.96354	0.03029	0.94802
Laplacian Eigenmap	0.64112	0.08715	0.75926
Laplacian RLS	0.89144	0.02667	**0.98651**

labeled examples (combined number of normal and known-attack terminated connections) for each classifier, with as few as 19 labeled examples. Once again, there are approximately 3000 unlabeled examples per experiment. We also count how often the IDS results recorded in the Kyoto2006+ dataset alerted on the data with normal and unknown attacks only. There are a total of 398 unknown attacks that occur during the 12 days in the test set.

If we look more closely at the individual results, the real promise of the Laplacian RLS, and potentially other semi-supervised methods whose assumptions match the domain, shines through. In Table 4 we provide the results of each of the 10 runs in order to demonstrate how low the number of false positives can be bounded. The first run has the lowest AUC score of 0.99968, but has the lowest false positive rate of 0.00022 (out of 808,108 normal events). It is also the only classifier to have a recall of less than 100%, but it still catches 99.75% of the unknown attacks. The binary Laplacian RLS model uses a threshold, so the AUC score indicates how much tradeoff needs to occur between precision and recall. Therefore, since the model that catches 397 unknown attacks, while missing only one, only has 178 false positive alerts and yet has the lowest AUC score, all of the other models should be tunable to allow them to miss a single attack while keeping their false positive number at 178 or lower, as well, since they require less of a tradeoff than the first model.

Given the AUC scores in Table 4, it makes sense to add an automatic threshold selection routine to the training step in order to obtain better performance. Table 5 and Table 6 show the results of the Laplacian RLS classifiers when the thresholds are tweaked during training (on training data) to eliminate false positives. In this case, we used a method whereby we rank all labeled training data by the score assigned by the model, and then we attempt to find a threshold that will guarantee a maximum false positive rate of 0.00000001 on the training data with the hope that this will transfer to the test data. We find the distance between this discovered threshold and the maximum score of 1, multiple it by 0.75, and add it to the old threshold to obtain a new one. Unfortunately, our choice of 0.75 is rather arbitrary, so despite the fact that the threshold is set on the training data, it is likely that such a method would need to be tweaked manually in practice based on the number of false positives that a user could tolerate. However, it is clear that these models are very powerful methods of finding unknown attacks, and it is equally clear that if the intention is to find previously unseen attacks, then these methods hold great promise for the defense of large networks. As mentioned above, the optimal threshold for each of these learners should guarantee fewer than 178 false positives for any of the the classifiers. Thus, the improvements shown in Table 6 can be improved upon as well. Therefore, future work will include better methods of automatic threshold generation, which is a particular challenge when the size of the training data is limited to realistic numbers as in this paper.

4.3 Additional Insight

In addition to the experimental results, it is interesting to observe the effect that the unlabeled data has on the dimensionality reduction used by the Laplacian Eigenmap approach. Therefore, as an example of a real nonlinear transformation using Laplacian Eigenmaps, Figure 1 shows the values of the first two non-zero eigenvectors for the labeled points during training of a Laplacian Eigenmap classifier on the smallest subset (19 labeled examples) of the Kyoto University data from January 1, 2008. The addition of the unlabeled data clearly improves the separability between classes along the first two dimensions (which are always the smoothest ones with respect to the graph). Since the method preserves closeness in the original graph, the addition of the unlabeled data can be seen as increasing the density so that it is possible to build a graph that does indeed represent a smooth manifold along which the attacks and normals vary. In terms of the Laplacian RLS, this means that the graph should have a small unlabeled error rate (see section 3.4), which appears to be confirmed by the experimental results.

Table 3: Alerting on unknown attacks. The Laplacian RLS classifiers were trained on subsets of fewer than 70 labeled data comprising only known attacks and known normals. *The signature IDS alerts are recorded in the dataset [29].

Classifier	Recall	| False Positive Rate|	AUC Score
Signature IDS*	0.00000	0.01619	N/A
Laplacian RLS	0.99975	0.02538	0.99987

Table 4: Performance of the individual classifiers (randomly selected training sets). All classifiers require less tradeoff between precision and recall than classifier 1. Therefore, they can all conceivably be tuned to achieve the same results: 178 or fewer false positives, while alerting on 397 out of 398 unknown attacks. *The signature IDS alerts are recorded in the dataset [29].

Classifier	|# Training Data|	Recall	|# False Negatives|	|# False Positives|	AUC Score
Signature IDS*	N/A	0.00000	398	13,074	N/A
Laplacian RLS 1	**19**	**0.99749**	**1**	**178**	**0.99968**
Laplacian RLS 2	57	1.0	0	14,753	**0.99993**
Laplacian RLS 3	58	1.0	0	28,498	**0.99992**
Laplacian RLS 4	60	1.0	0	28,498	**0.99970**
Laplacian RLS 5	64	1.0	0	25,621	**0.99993**
Laplacian RLS 6	65	1.0	0	17,456	**0.99993**
Laplacian RLS 7	59	1.0	0	18,278	**0.99986**
Laplacian RLS 8	69	1.0	0	28,498	**0.99986**
Laplacian RLS 9	57	1.0	0	28,498	**0.99995**
Laplacian RLS 10	58	1.0	0	14,707	**0.99995**

Table 5: Alerting on unknown attacks. The Laplacian RLS classifiers were trained on subsets of fewer than 70 labeled data comprising only known attacks and known normals, and they were built using automatic threshold-finding functions intended to reduce false positive alerts.. *The signature IDS alerts are recorded in the dataset [29].

Classifier	Recall	| False Positive Rate|	AUC Score
Signature IDS*	0.00000	0.01619	N/A
Laplacian RLS	0.99749	0.00166	0.99987

Table 6: Performance of the individual classifiers (randomly selected training sets) when using an automatic threshold-finding function during training. This function is intended to raise the threshold to avoid false positive alerts, but it only uses training-data to find the threshold. *The signature IDS alerts are recorded in the dataset [29].

Classifier	|# Training Data|	Recall	|# False Negatives|	|# False Positives|	AUC Score
Signature IDS*	N/A	0.00000	398	13,074	N/A
Laplacian RLS 1	19	**0.99749**	**1**	**164**	**0.99968**
Laplacian RLS 2	57	**0.99749**	**1**	**173**	**0.99993**
Laplacian RLS 3	58	**0.99749**	**1**	**676**	**0.99992**
Laplacian RLS 4	60	**0.99749**	**1**	**9807**	**0.99970**
Laplacian RLS 5	64	**0.99749**	**1**	**166**	**0.99993**
Laplacian RLS 6	65	**0.99749**	**1**	**167**	**0.99993**
Laplacian RLS 7	59	**0.99749**	**1**	**1779**	**0.99986**
Laplacian RLS 8	69	**0.99749**	**1**	**166**	**0.99986**
Laplacian RLS 9	57	**0.99749**	**1**	**203**	**0.99995**
Laplacian RLS 10	58	**0.99749**	**1**	**151**	**0.99995**

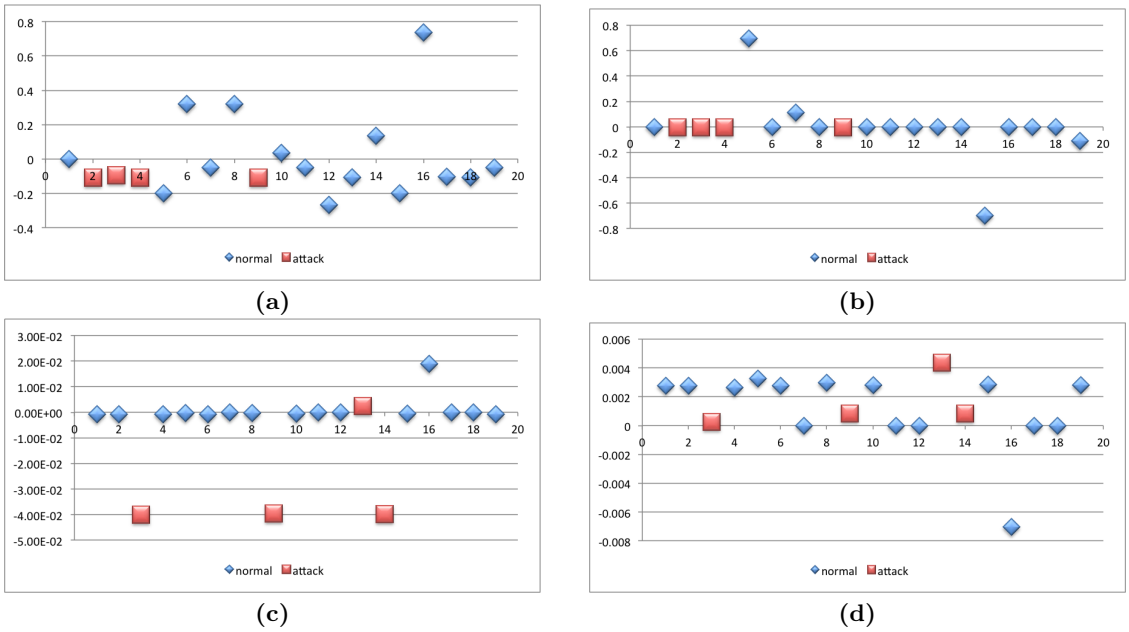

Figure 1: Eigenvector values of the labeled data from the graph Laplacian based on nearest neighbors. 1a and 1b are the values in the first and second dimensions, respectively, using a graph based on labeled points only. 1c and 1d are the values in the first and second dimensions, respectively, when using a graph of the labeled and unlabeled data. In this case, the addition of the unlabeled data provides a transformation that allows linear separation of the two classes along the first two dimensions.

5. DISCUSSION

This paper provides a demonstration of true generalization performance on real operational network data using a small, randomly selected, set of known attacks and normal connections for training. The resulting classifiers trained on data from Jan 1, 2008 were tested on twelve days of data from the same environment from the latter half of 2008 (the same test set used in [17]). Each of the models alerted on 397 out of 398 *unknown* attacks during the 12-day window, while generating false positive alerts at a rate an order of magnitude lower than that of a signature tool incapable of catching previously unseen attacks. In fact, an alert from a typical model would have a greater than two-thirds chance of being an actual previously unseen attack as opposed to a false alert. Given the difficulties that machine-learning-based intrusion detection has faced in reaching operational status, it is important to continue to improve the performance of such classifiers, but it is equally important to do so on recent operational data. Our hope is that dramatic results such as those presented here on non-synthetic data pulled from a real operational network can help this field move forward more rapidly into real systems. We are extremely grateful to the curators of the Kyoto University data [29] for the release of their data and hope that others will follow suit.

It should not be surprising that performance on unknown attacks has the potential to be very high. While some known attacks, such as probes and denial of service attacks, may be nearly indistinguishable from normal traffic due to their nature, more insidious attacks are harder to blend into the environment while still accomplishing their goals. Although this paper hopes to emphasize the critical nature of in-situ training, we also demonstrate that such training can be made to be extremely cost effective when using the latest semi-supervised models, if the employed models have been carefully selected such that their implicit assumptions match the data domain.

The base-rate fallacy [3] is a real issue in intrusion detection, but demonstrations like the one in this paper of real classifiers with tiny false positive numbers, such that they are fewer than one-third of real attacks detected, go a long way to eliminating the problem. In particular, this is significant when the training requirements for such models can be limited to extremely reasonable numbers. In fact, the in-situ training requirements for the models in this paper are so small that they are dwarfed by the costs of dealing with missed attacks or investigating false positives from other tools. Our hope is that if we can begin to gather strong evidence of useable performance on real datasets, such as the results in this paper, then the community can move more aggressively to ubiquitously deploy tools that are trainable in place and that generalize effectively to catch previously unseen attack types. Improvements beyond these approaches may be necessary in the long run to deal with strong adversarial learning scenarios, but we believe it is currently possible for large network systems to augment the use of signature tools with more dynamic, intelligent systems with little cost in terms of training or the investigation of false alerts.

6. ACKNOWLEDGMENTS

Research sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U.S. Department of Energy under contract no. DE-AC05-00OR22725. This manuscript has been authored by UT-Battelle, LLC,

under contract DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes..

7. REFERENCES

[1] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[2] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[3] S. Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(3):186–205, Aug. 2000.

[4] M.-F. Balcan. *New Theoretical Frameworks for Machine Learning*. Phd thesis, 2008.

[5] M.-F. Balcan and A. Blum. *An Augmented PAC Model for Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[6] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[7] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56:209–239, 2004.

[8] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[9] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection - a survey. *ACM Computing Surveys*, 41(3), 2009.

[10] O. Chapelle, B. Scholkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[11] C. Chen, Y. Gong, and Y. Tian. Semi-supervised learning methods for network intrusion detection. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, pages 2603 –2608, 2008.

[12] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, Providence, RI, 1997.

[13] W. W. Cohen. Minorthird: Methods for identifying names and ontological relations in text using heuristics for inducing regularities from data, 2004.

[14] S. Dua and X. Du. *Data Mining and Machine Learning in Cyber Security*. CRC Press, 2011.

[15] P. Flach, J. Hernandez-Orallo, and C. Ferri. A coherent interpretation of auc as a measure of aggregated classification performance. In *the 28th International Conference on Machine Learning*, 2011.

[16] R. Johnson and T. Zhang. Graph-based semi-supervised learning and spectral kernel design. *IEEE Transactions on Information Theory*, 54(1):275–288, 2008.

[17] K. Kishimoto, H. Yamaki, and H. Takakura. Improving performance of anomaly-based ids by combining multiple classifiers. In *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, pages 366–371, 2011.

[18] T. Lane. *A Decision-Theoretic, Semi-Supervised Model for Intrusion Detection*. Springer London, 2006.

[19] F. Liang, K. Mao, M. Liao, S. Mukherjee, and M. West. Nonparametric bayesian kernel models. Technical report, Duke University, 2007.

[20] F. Liang, S. Mukherjee, and M. West. The use of unlabeled data in predictive modeling. *Statistical Science*, 22(2):189–205, 2007.

[21] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In *the 2000 DARPA Information Survivability Conference and Exposition*, 2000.

[22] C.-H. Mao, H.-M. Lee, D. Parikh, T. Chen, and S.-Y. Huang. Semi-supervised co-training and active learning based approach for multi-view intrusion detection. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 2042–2048, New York, NY, USA, 2009. ACM.

[23] P. Orbanz and Y. W. Teh. *Bayesian Nonparametric Models*. Springer, 2010.

[24] M. Ouimet and Y. Bengio. Greedy spectral embedding. In *the 10th Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2005.

[25] N. S. Pillai, Q. Wu, F. Liang, S. Mukherjee, and R. L. Wolpert. Characterizing the function space for bayesian kernel models. *Journal of Machine Learning Research*, 8:1769–1797, 2007.

[26] D. Pyle. *Data Preparation for Data Mining, Volume 1*. Morgan Kaufmann, 1999.

[27] N. Roy, G. Gordon, and S. Thrun. Finding approximate pomdp solutions through belief compression. *Journal of Artificial Intelligence Research*, 23:1–40, 2005.

[28] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of IEEE Symposium on Security and Privacy*, 2010.

[29] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao. Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation. In *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns*, 2011.

[30] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994 – 12000, 2009.