Computer Network Anomaly Detection Utilizing Restricted Boltzmann Machines And Discriminative Regression [1]

A.W.R. Gold

25 June 2019

Abstract

As internet connectivity increases at a dramatic pace around the world, the volume of traffic transmitted on computer networks is at the highest point in human history with no slowdown in sight. With new technologies such as the Internet of Things (IoT), cybersecurity tools become a necessary tool for individuals and organizations to thwart potential attackers from exploiting inherent weaknesses in computer networks. However, due to the voluminous nature of modern internet traffic and the ever-evolving internet landscape, machine learning approaches that can automatically detect anomalies with no human interaction merit research and investigation. This thesis explores the effectiveness of an energy-based machine learning model that utilizes the expressive capabilities of generative modelling, such that benign traffic patterns can be learned and "memorized" by the model to be used in discriminating between normal traffic and potential threats. The principal model is compared with a baseline regression model for comparison. Results of the experiments were largely inconclusive, as the baseline model outperformed the energy-based model in every experiment, though discrepancies between the two models' abilities at specific classification tasks were notable.

Keywords: Network Traffic Anomaly Detection Restricted Boltzmann Machine

1 Introduction

The general purpose of a computer network intrusion detection system (IDS) is the detection of anomalous network traffic behavior in the effort to identify and prevent intrusion attempts, malware infection, and disruption of network functionality. Intrusion detection systems that utilize heuristic methods to categorize traffic patterns are inherently weak at detecting previously unseen patterns, whether malicious or benign. Rule-based systems are fundamentally ill-equipped to adequately protect a network or machine from novel attacks that utilize previously unencountered attack vectors, which leads to the need for an IDS model capable of identifying such potential threats. An IDS that utilizes generative machine learning models such as the Restricted Boltzmann Machine (RBM) is an attractive area of research interest, due to the robust outlier detection capabilities of such models. Due to the ever-changing nature of cybersecurity threats and paradigms, generative models capable of detecting abnormal traffic patterns are of particular interest due to the inherent property that these models possess such that they need not be constantly adapted to newly encountered threats.

Generative models in this domain are trained on benign network traffic examples, and are capable of generating approximated models of such traffic that are then used to compare testing data with the benign model to see how well each testing instance fits the model's approximation. Restricted Boltzmann Machines are particularly suited to the task of anomaly detection, and when coupled into an ensemble model alongside a discriminative classifier, such as a logistic regressor or support vector machine, are potentially capable of not only identifying potentially malicious behavior as anomalies, but also the specific type of incoming attack [1, 3]. Previously unseen potential threats can be compared to the benign model to determine a likelihood that such an anomaly fits a benign traffic pattern, and can be further compared with previously encountered malicious patterns to determine a likelihood that the anomaly belongs to a specific threat category.

The challenge in such approaches lies particularly in the ability of the model to approximate a robust description of benign traffic, and is heavily dependent on the quality and quantity of data it is trained upon. Many cybersecurity threats are naturally disguised as benign activity in an attempt to avoid detection, which is why the model must be properly and extensively trained on all known types of benign traffic in order to reduce the surface area of potential attack vectors in the generated model. Cybersecurity training data sets have existed for many years [11] that consist of various benign and malicious traffic patterns to be used in machine learning applications, however these data sets are usually limited in scope and scale for various reasons. Many such data sets

[1] This thesis was prepared in partial fulfillment of the requirements for the Degree of Bachelor of Science in Data Science and Knowledge Engineering, Maastricht University. Supervisors: Kurt Driessens & Evgueni Smirnov.

artificially introduce malicious attacks into a controlled network, where every packet is captured and stored for later analysis. However, such artificial environments are difficult to model as "real-world" environments for reasons such as the reluctance of network administrators to allow malicious traffic to be transmitted on a network with real users, potentially putting sensitive personal data and devices at risk of infection. Therefore, these data sets must take explicit care in reproducing realistic conditions without putting users or vital network components at risk of damage or exploitation, which naturally inhibits the scale of such controlled environments. Furthermore, researchers must also take care to introduce a wide variety of known benign traffic patterns such that any models trained on these data sets are exposed to realistic benign behavior to avoid mis-classification.

In this thesis, a Gaussian-Bernoulli Restricted Boltzmann Machine is utilized as a generative model for describing the features of benign network traffic as a joint probability distribution between its inputs and hidden layer activations, and is coupled with discriminative logistic regression for classification of outliers that don't conform to the RBM's interpretation of the benign training data. The main purpose of this approach is to apply feature extraction and dimensionality reduction to better inform the classification model during testing and deployment. One of the advantages of feature extraction in this domain is that network traffic packets often adhere to similar patterns depending on the type of traffic, and should abnormal behavior be encountered, the extracted features of known benign traffic can be compared to determine the likelihood that the abnormal behavior is indeed anomalous. Furthermore, such a model is advantageous in its domain-agnostic applications such that it can aid in the detection of novel threats that have never before been encountered without the need to be trained to identify such threats. Therefore, the efficacy of the model's generative abilities and feature extraction, when utilized in a stacked model with a discriminative classifier, is compared to a baseline regression classifier over the same data. More specifically, the RBM's ability to discern between different classes of malicious traffic that it has never encountered is of particular interest, as the purpose of such a model is to be robust in the detection of examples that lie outside the norm of what it has previously encountered.

2 Background and Related Research

2.1 Anomaly Detection and Cybersecurity

Network security research has existed for decades with research in anomalous pattern detection dating back to 1980 [2], where early efforts were directed at creating "profiles" of behavior, and detecting any patterns that did not fit those profiles. During this time, access to the internet was largely limited to governments and academic institutions, where overall traffic was but a mere fraction of today's volume. However, the concept of identifying anomalies persisted outside the domain of machine learning until efficient algorithms for neural networks and deep learning would be developed, eventually leading to renewed interest in the domain around the turn of the 21st century.

Among the leading researchers contributing to the dramatic improvement in deep learning algorithms was Geoffrey Hinton, who is often called "the godfather of deep learning" [20, 21]. Improvements in learning speed for energy-based models such as the Boltzmann Machine, and its variant, the Restricted Boltzmann Machine, were largely directed at improvements in image classification via the use of stacked energy-based models known as Deep Belief Networks [6]. However, these energy-based models proved to be applicable in a wide variety of domains, and in 2012 Uro Fiore et al. described their implementation of a Discriminative RBM (DRBM) in the domain of network traffic anomaly detection [3]. Their work specifically addressed a semi-supervised learning approach where the RBM served as an unsupervised model trained only on benign traffic, such that any examples that could not be classified as benign was defined as anomalous via a supervised classification approach.

Thereafter, Aldwairi et al. published their evaluation of the performance of a Restricted Boltzmann Machine in computer network anomaly and intrusion detection [1], utilizing primarily the more recent cybersecurity dataset from the Information Security Center of Excellence (ISCX) [17]. Fiore and colleagues discovered that synthetic training environments led to significant model discrepancies depending on the testing environment. Aldwairi et al. attempted to rectify this by utilizing a more meticulously compiled benchmark dataset, focusing largely on the preprocessing of training data alongside rigorous tuning of the model hyperparameters. Among their principal conclusions however was that such extensive manipulation of training data significantly hampered their methodology in real-time and online learning scenarios.

2.2 Boltzmann Machines

Named after the Boltzmann Distribution from statistical mechanics, Boltzmann Machines are a stochastic recurrent neural network capable of feature detection and representation. Proposed in 1985 by Geoffrey Hinton and Terry Sejnowski in [9], they are among the first neural networks capable of learning internal representations of their inputs, and led to further developments including the Restricted Boltzmann Machine. The Boltzmann Machine is comprised of two interconnected visible and hidden layers of neurons (see Fig. 1). The visible layer

$v = (v_1, ..., v_m)$ is comprised of stochastic binary neurons, and each of these neurons is bidirectionally connected to a stochastic non-observable hidden layer of binary units, $h = (h_1, ..., h_n)$. Each of these connections are influenced by a matrix of weights $W_{(i,j)}$. The activations of the hidden units act as descriptors of the state of the system over time, which represents the probability of being in a given state at a given point in time. The probability of being in state $P(v, h)$ is dependent on the energy $E(v, h)$ of that state, belonging to a Boltzmann distribution function [1]:

$$P(v, h) = \frac{e^{-E(v,h)}}{Z} \tag{1}$$

where Z represents the normalization (or partition) function such that the probability of all states must sum to 1:

$$Z = \sum_{(v,h)}^{M} e^{-E(v,h)} \tag{2}$$

where M represents the total number of possible states (v, h) accessible by the machine, resulting in the joint probability of every possible pair of visible and hidden units [8]:

$$P(v, h) = \frac{e^{-E(v,h)}}{\sum_{(v,h)}^{M} e^{-E(v,h)}} \tag{3}$$

The goal is to approximate the distribution of the inputs through iterative reconstruction, and the difference between the "real" input $P^+(V)$ and the machine's reconstruction $P^-(V)$ is measured via the Kullback-Leibler Divergence [12], G:

$$G = \sum_{v} P^+(v) \ln \left(\frac{P^+(v)}{P^-(v)} \right) \tag{4}$$

where G is a function of the weights W, which in turn determine the energy E of the system [9]. W is optimized, usually via gradient descent, such that the energy of the system eventually settles into "thermal equilibrium," known as convergence. However, Boltzmann Machines are decidedly inefficient when approximating distributions beyond a trivial scale, which led to the development of the Restricted Boltzmann Machine.

2.3 Restricted Boltzmann Machines

The Restricted Boltzmann Machine was first proposed in 1986 under the name *Harmonium* by Paul Smolensky [19] and further developed by Geoffrey Hinton and several colleagues in the early 2000's, culminating in several papers by Hinton, et al. [6, 7, 10], which are widely regarded as monumental contributions to the field of machine learning that reinvigorated the scientific community's interest in deep learning. While the Boltzmann Machine is a fully-connected graphical model, the RBM

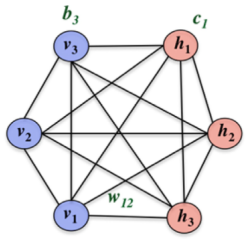

Figure 1: Graphical representation of a fully-connected Boltzmann Machine with 3 visible nodes v_i and 3 hidden nodes h_j, from [1].

differs in that there exist no intra-layer connections between nodes, forming a bipartite graph. Such a structure means that each of the hidden nodes are independent of one another given the input of the visible layer, and inversely the visible nodes are mutually independent given the activations of the hidden layer.

Traditionally, the visible and hidden layers are fixed to binary states in what is often called a Bernoulli Restricted Boltzmann Machine [8], though it is also possible to input continuous values into the visible layer while retaining the binary activations of the hidden layer. Such a model is referred to as a Gaussian-Bernoulli Restricted Boltzmann Machine [8], and is the principal model used in this thesis.

A Restricted Boltzmann Machine is trained in a similar fashion to Boltzmann Machines, aside from a few key differences. Most notably, the energy function E differs from a traditional Boltzmann Machine such that:

$$E(v, h) = -h^T W v - b_v^T v - b_h^T h \tag{5}$$

where h^T is the matrix transposition of h, and b_v and b_h are the biases of the visible and hidden units, respectively [3, 8]. This allows for the expression of the conditional probabilities of the visible units given the hidden activations:

$$P(v|h) = \prod_{i=1}^{m} P(v_i|h) \tag{6}$$

and the conditional probability of the hidden activations, given the state of the visible units:

$$P(h|v) = \prod_{j=1}^{n} P(h_j|v) \tag{7}$$

However, it is significantly more difficult to obtain an unbiased sample of the entire model's expectation. As Hinton describes in his guide to training Restricted Boltzmann Machines [8], obtaining such a sample "...can be done by starting at any random state of the visible units and performing alternating Gibbs sampling for a very long time."

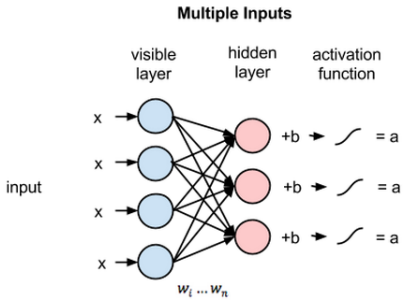

Figure 2: Graphical representation of an RBM with multiple inputs x, weight matrix W, and the logistic sigmoid activation function σ, from [18].

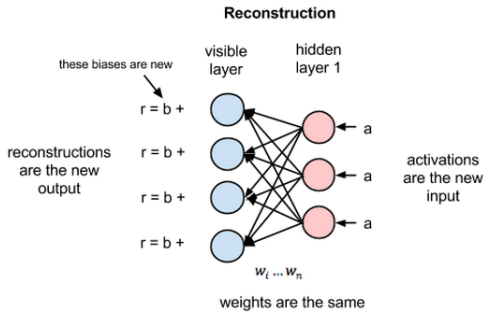

Figure 3: Graphical representation of the reconstruction phase of training an RBM, from [18].

2.3.1 Gibbs Sampling and Contrastive Divergence

Unlike a traditional Boltzmann Machine, the lack of interconnection between the hidden units means it is easy to obtain an unbiased sample of the hidden layer [8]. Given an arbitrary input v, the binary activations h_j for each hidden unit j is set to 1 with probability:

$$P(h_j = 1|v) = \sigma(b_h^j + \sum_i v_i w_{ij}) \tag{8}$$

where $\sigma(x)$ is the logistic sigmoid function $\frac{1}{1+e^{-x}}$. For the same reasons stated above, it is just as easy to obtain an unbiased sample of the visible layer. Given the hidden activations h, equation 9 (below) is used to predict the new input values v:

$$P(v_i = 1|h) = \sigma(b_v^i + \sum_j h_j w_{ij}) \tag{9}$$

This process, known as *Gibbs sampling*, is repeated a finite number of times, with each iteration updating another input v recreated from the original input value, resulting the fully estimated probability distribution function $\tilde{v}$ for the model.

2.3.2 Contrastive Divergence:

Training an RBM is performed by minimizing the energy of the machine, with the most popular method being minimization of the log-likelihood gradient of the training data [1, 3, 8]. The exact computation of the gradient is intractable as it requires summing over all possible visible vectors, but can be estimated via an iterative process of updating the weights and biases of the machine, known as *Contrastive Divergence* (CD), proposed in [7]. This process begins by initializing the states of all visible units to a training vector v_0 and then simultaneously computing the hidden layer activations using equation 8 (see Fig. 2). With this result, a "reconstruction" of the input is computed by setting each visible unit v_i with the probability determined using equation 9 (see Fig. 3).

The weight matrix W is then updated by calculating the difference of the outer product between the input vector v_0 and its hidden layer activation probabilities (known as the *positive gradient*, $P^+(v)$), and the reconstructed vector v_k and its hidden layer activation probabilities (known as the *negative gradient*, $P^-(v)$), times some learning rate ϵ, over k Gibbs sampling steps. Both of the vectors v_0 and v_k are used to calculate the activation probabilities for the hidden values. The difference between the outer products of those probabilities with input vectors results in the updated matrix W:

$$\Delta W = \epsilon(v_0 \otimes P(h_0|v_0) - v_k \otimes P(h_k|v_k)) \tag{10}$$

Using the updated matrix, the new weights can be calculated with gradient descent:

$$\Delta W_{new} = W_{old} + \Delta W \tag{11}$$

and the biases are updated in similar fashion:

$$\Delta b_v = \epsilon(v_0 - v_k), \Delta b_h = \epsilon(h_0 - h_k) \tag{12}$$

2.3.3 Persistent Contrastive Divergence:

One of the drawbacks of contrastive divergence is that with a low k value (such as $k = 1$), calculating the gradient of the sample is efficient, however doesn't accurately reflect the global probability distribution of the input due to the low mixing rate of the Markov chains between training epochs. A solution was proposed by Tim Tieleman in [22], such that at the beginning of a new training epoch, instead of initializing the visible units to a new training vector v_0, *Persistent Contrastive Divergence* (PCD) initializes the visible units to the state in which the previous iteration "finished," essentially leading to a much more accurate estimation of the true gradient of a model over a finite number of training epochs. This estimation is still inexact, however given an infinitesimally small learning rate ϵ, it becomes an exact calculation [22]. In this thesis, Persistent Contrastive Divergence is utilized in place of Contrastive Divergence.

2.4 State of the Art Generative Models

Recently, however, the Restricted Boltzmann Machine has become generally regarded by the machine learning community as outdated in many regards [5], particularly due to its relative instability and the intractable nature of the partition function when estimating the probability distribution of the inputs. Researchers tend to favor various types of autoencoders for similar tasks, including variational autoencoders (VAE) and denoising autoencoders, as Goodfellow describes in his book *Deep Learning* [5]. He states, "the VAE framework is straightforward to extend to a wide range of model architectures. This is a key advantage over Boltzmann machines, which require extremely careful model design to maintain tractability." Goodfellow explains further, "the variational autoencoder approach is elegant, theoretically pleasing, and simple to implement. It also obtains excellent results and is among the state-of-the-art approaches to generative modeling."

Nonetheless, the Restricted Boltzmann Machine is explored herein as a facet of a semi-supervised model for feature extraction and dimensionality reduction, before the learned features are coupled with supervised logistic regression for anomaly detection and classification. The results of such a semi-supervised approach are compared to rudimentary supervised learning classification methods. As Goodfellow stated above, the capricious nature of the RBM requires meticulous engineering; as such, not only is the performance and efficacy of the RBM evaluated in the cybersecurity domain, but also the practicality of the model considering the availability of more state-of-the-art techniques. The semi-supervised nature of the model is of particular interest, as other more modern techniques may be applied in similar fashion such that they may be able to complement or replace entirely the RBM's functionality in both the cybersecurity domain, as well as more generalized anomaly detection applications.

3 Anomaly Detection and Classification

3.1 Energy-Based Models

Energy-based models such as the RBM estimate the probability distribution of the inputs to the model, and can utilize the activation probabilities of the hidden layer to train another model in a stacked ensemble method. These may include other RBMs, which when stacked together in multiple layers are known as Deep RBMs (not to be confused with DRBMs mentioned earlier), which are a specific type of Deep Belief Network [6], or may otherwise be other classifiers such as a logistic regressor. Pairing an RBM with a classifier

In a semi-supervised model such as the DRBM, the RBM is extensively trained on benign network traffic in an unsupervised fashion, allowing the model to express its interpretation of the benign traffic as the joint probability distribution of the inputs and the activations of its hidden layer. Previously unseen patterns of network traffic when introduced to the trained model are given a pseudo-likelihood based on the trained model's parameters, and the difference between these two functions are utilized during discriminative classification. Several methods exist for comparison between the model's interpretation of training and testing data via computing a reconstruction error, and during the classification phase a classifier can utilize the activation probabilities of the hidden layer within the RBM, or by logistically regressing over the estimated energy level of the machine, given testing data alongside target labels for training.

The supervised phase of a semi-supervised model is comprised of a discriminative logistic regressor, which is trained on the activation probabilities of the hidden layer of the RBM alongside target labels, and then tested using unseen testing data. This allows the RBM to interpret the features of the input data (benign traffic) and then measure the probability that new, unseen input belongs to the Benign category. It should be considered that the process of training an RBM on every possible type of benign traffic is an unrealistic task, and as such even though the model is trained on exclusively benign examples, the introduction of new benign examples that deviate in any way from the model's interpretation may lead to the erroneous classification of an anomaly. Therefore it is of utmost importance with such models that the training phase be as thorough and extensive as possible to reduce this possibility.

3.2 Precision and Recall in Cybersecurity Applications

With respect to the domain of this thesis, it is important to note that any approach in classifying network traffic must emphasize completeness in positively classifying all potentially malicious instances, such that the true positive rate (TPR) of classification is maximized towards 1. While undesirable, false positives (FP), or the mis-classification of benign traffic as malicious, are far more acceptable than false negatives (FN), or the mis-classifying malicious traffic as benign. Other accuracy metrics worth observing are the true negative rate (TNR), false positive rate (FPR), false negative rate (FNR), and positive predictive value (PPV) - otherwise known as precision.

The balance between precision (also known as relevance) and recall (also known as sensitivity) in an intrusion detection system is worth investigation depending on the scenario, as it has been shown that the majority of classification errors in semi-supervised anomaly detection models are false positives [3], or benign examples

mis-classified as malicious. While such false positives can be inconvenient for the user, when security of a network is of paramount importance it becomes necessary for additional analysis of all positives, leading to the consumption of time and resources by security administrators. Therefore, the purpose of such energy-based anomaly detection models is to build as robust a model capable of expressing the nuances of benign traffic such that security administrators can assuredly classify any detected anomalies as potential threats.

4 Experiments

4.1 Dataset

The dataset used in this analysis is the CICIDS2017[1] cybersecurity dataset [16], a research-quality dataset created by the Canadian Institute for Cybersecurity (CIC) and generated over the course of 5 days in a controlled network environment where 14 specific types of modern cybersecurity threats were injected into the network in controlled intervals. Packet capture software was used to collect all packets transmitted on the network over the course of the experiment, and each attack was labeled and time-stamped for efficient observation and training. Background benign traffic was synthesized based on an abstract model of benign human behavior proposed in [4, 13, 17].

The raw dataset is originally composed of 3119345 packet capture instances containing 83 attributes[2] and 15 class labels - 1 benign and 14 malicious labels (see Table 1) - however some attributes were combined or dropped for a few reasons. The "Source" and "Destination IP" address attributes were dropped due in part to the synthetic nature of the various attack origins and targets, as the environment in which the dataset was generated was a closed computer network such that multiple different malicious events originated from the same IP address. Indeed, this and other flaws were also addressed by Panigrahi and Borah in [14], where they describe in detail several relevant preprocessing methods for the CICIDS2017 dataset. In the CICIDS2017 dataset there exists 288602 instances with missing class labels, which comprises nearly 10% of the entire dataset. Additionally, there exists 203 instances with missing or infinite attribute values. By removing these unwanted instances from the dataset, 2830540 unique instances remain.

Another potential shortcoming of the dataset is the high class imbalance between the benign traffic and the malicious attacks, as approximately 83.34% of the data belongs to the "Benign" class (see Fig. 4). Furthermore, between the malicious classes there exist some extreme imbalances, for example the "Heartbleed" attack class contains just 11 instances, representing fewer than 0.001% of the malicious examples and fewer than 0.0004% overall, while the "DoS Hulk" class contains 230124 instances, representing over 41% of all malicious examples and 7.3% of the overall instances. Such imbalance can lead to strong bias towards the majority class (Benign), and during multi-class classification can lead to overfitting of the over-represented malicious classes, such as the DoS attacks which make up roughly 45% of all malicious examples. Steps were taken to address this, described later in this section.

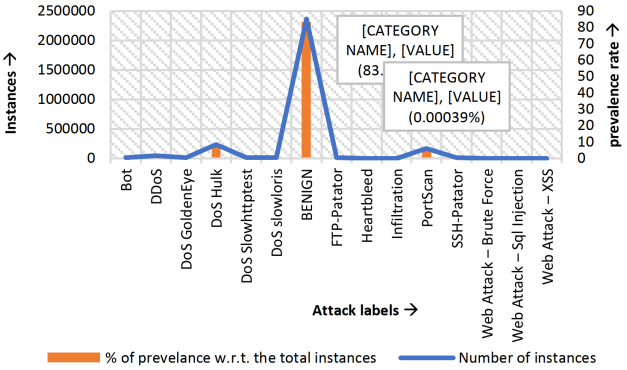

Figure 4: Graphical distribution of class categories in the CICIDS2017 dataset, from [14].

4.1.1 Preprocessing

After removing the undesirable instances mentioned above, a number of steps were taken to prepare the data for training. First, in anticipation of training an RBM on exclusively benign data, 50% of all labeled benign traffic in the dataset was set aside and processed separately. This was performed to prevent any benign examples from leaking into any of the models during the experimentation phase, such that the RBM trained on exclusively benign traffic would never witness the same benign traffic packets when transforming[3] training and testing sets for classification. Such a leak would lead to erroneous results, as the model is more likely to correctly classify benign data examples it has already witnessed during training. The training and testing datasets used in both the baseline classification model (to which the RBM model performance is compared) as well as the principal model, are comprised of the remaining benign traffic and 100% of the malicious traffic examples. All attribute values (excluding the class label vector) were scaled and normalized[4] as continuous values between 0 and 1. The class label vector was numerically encoded in several different encodings with integer values, with the purpose of separating the target labels into a few different

[1] https://www.unb.ca/cic/datasets/ids-2017.html

[2] Information regarding the attributes can be found at: http://www.netflowmeter.ca/netflowmeter.html

[3] Calculate the hidden layer activation probabilities.

[4] Utilizing the MinMaxScaler in the Python machine learning library Scikit-Learn [15]

experiment categories. For binary classification between benign and malicious instances, the target labels were encoded into a separate target vector as 0 (Benign) and 1 (Malicious). For multi-class classification in a separate training instance, the target labels were encoded into a target vector with integer values ranging from 0 (Benign) to between 1-14, with one label for each specific class of malicious example.

Additionally, due to the high class imbalance mentioned above, several individual malicious classes were combined into more generalized meta-classes in a third training instance (see Table 1). The classes with significantly fewer instances were combined and labeled from 0 (Benign) to integer values between 1-6, namely:

- SSH Brute Force and FTP Brute Force
- DoS-slowloris, DoS-Slowhttptest, DoS-Hulk, DoS-GoldenEye, and Distributed DoS (DDoS)
- Heartbleed and Dropbox Infiltration
- Web-based Brute Force, XSS, and SQL-injection

However, the above meta-class dataset is not fully balanced as it just reduces certain imbalances in the target classes with very few examples. Therefore, a fully-balanced binary classification dataset with 50% benign and 50% malicious data is constructed, which contains 100% of the malicious examples and an identical number of randomly sampled benign traffic examples.

4.1.2 Training, Testing, and Validation Sets

Aside from the benign training set which was set aside and processed separately at the beginning of the preprocessing stage, stratified training, validation, and testing sets were crafted for the experiments. The stratified training sets were crafted with 80% of the data, and the testing set with the remaining 20%. The validation sets were constructed by removing 20% of the aforementioned training set before training the model, and used to tune the hyperparameters of the model via 5-fold grid search cross validation across a number of hyperparameter options. After the hyperparameters were tuned, the validation set was recombined with the training set for training and testing of the model in full. This procedure was done to ensure that during the experiments, the model would be trained on as many examples as possible while still minimizing data leakage during the hyperparameter-tuning process.

The results of hyperparameter tuning revealed several distinctive traits of both the principal model and the baseline model. First, a relatively small batch size of 1000 for the RBM was sufficient in reaching equilibrium across all training and validation sets. Secondly, a relatively low learning rate ε for the RBM was sufficient in training the RBM such that a state of relative equilibrium was reached in a relatively small number of epochs, even for such a large training set. This may be partially attributed to the dramatic improvement in gradient estimation that Persistent Contrastive Divergence provides over traditional Contrastive Divergence. Finally, strong regularization of the training data during classification led to significantly improved results in the validation set, which would be logical considering the high dimensionality of the data.

During hyperparameter tuning, it is important to note that with the Scikit-Learn toolkit used for these experiments, it was impossible to optimize the hyperparameters of a purely benign-trained RBM in tandem with the hyperparameters of the classification model, as the Scikit-Learn logistic regressor cannot be trained on a single-class instance. As such, the hyperparameters of both the RBM and classifier were tuned in two different methods. The first cross-validation approach utilized training and validation sets that contained both benign and malicious examples, and the hyperparameters for the RBM were chosen based on the highest average recall accuracy across all malicious classes (as the true positive rate is considered of utmost importance in this thesis). Using these chosen hyperparameters the RBM was then trained on exclusively benign traffic, and the classifier's hyperparameters were tuned by training the logistic regressor utilizing the RBM's estimation of the training set's hidden activation probabilities. These results were validated using the validation set, and the best parameters were chosen and fixed for the following experiments.

4.2 Methodology

The principal model utilized in this investigation can be described in two parts: the unsupervised learning phase, where the Gaussian-Bernoulli Restricted Boltzmann Machine is trained exclusively on benign network traffic examples in the form of packet captures with 77 numerical attributes, input as normalized continuous numerical values between (0,1) with no target labels. Secondly, the supervised learning phase, where the logistic classifier is fit using the trained RBM's estimated hidden layer activation probabilities of traffic examples containing both benign and malicious data packets in the form of the training sets described above. Finally, the trained logistic classifier tests the hidden layer activation probabilities of each of the testing datasets, and is evaluated based on several metrics - namely precision, recall, accuracy, and F1 score. The F1-score is an accuracy measure that balances the true positive rate with the predictive value of the model on a scale from 0.0 to 1.0, with a perfect score being 1.0. A high F1 score is indicative of both low false positives and low false negatives in the prediction model. These metrics are calculated as such:

- **Recall/True Positive Rate (TPR):**

$$TPR = \frac{TP}{P} \tag{13}$$

- **Precision/Positive Predictive Value (PPV):**

$$PPV = \frac{TP}{TP + FP} \tag{14}$$

- **Accuracy:**

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{15}$$

- **F1 score (F1)**

$$F1 = 2 \times \frac{TPR \times PPV}{TPR + PPV} \tag{16}$$

No evaluation metric perfectly summarizes the capabilities of a model in a single score value, however when coupled together can provide insight to the model's performance and shortcomings. The principal model's performance will be compared across all testing datasets against each other and a logistic regression model with tuned hyperparameters acting as a baseline model[5].

5 Results and Discussion

Training times for the baseline classification model fluctuated moderately between approximately 61 seconds for the balanced binary classification training set to 685 seconds for multi-class training set, with a mean/median training time of 305.75 and 238.5 seconds, respectively, across all training sets. The baseline model's prediction of testing sets (comprised of 338444 examples each) fluctuated more dramatically between 27 milliseconds for the binary testing set, to 1530 milliseconds for the multi-class testing set, with a mean/median prediction wall time of 702.75 and 627.0 milliseconds, respectively. The baseline classifier model was able to converge before reaching its maximum number of allowed iterations on the binary and balanced binary datasets, however did not converge in time for the meta-class and multi-class datasets.

For the principal model training times were higher, but not prohibitively so. Training the RBM on 1135660 purely benign data examples across 15 training epochs took 231 seconds, with which the trained model was used to transform separate datasets for training and testing the discriminative classifier. Training the classifier with the binary training set took 165 seconds, and did not converge before reaching its maximum alloted iterations. Prediction of the testing set on the same model took 721 milliseconds. The principal model took 107 seconds to fit to the transformed balanced binary training data, 392 milliseconds to predict the testing data, and was also unable to converge. The meta-class training set on the same model took 388 seconds to train, and prediction for the testing set took 632 milliseconds, also failing to converge. The multi-class training set took the longest to train at a wall time of 688 seconds, reaching the maximum iterations without convergence. Predicting the multi-class testing set took 659 milliseconds.

To determine the speed with which both the trained baseline and principal model could predict unseen examples, a single example was passed to each fitted classifier model after transformation by the trained RBM. For all models and datasets, the prediction time for a single sample was under 1 millisecond, indicating the ability to quickly predict new incoming packets should the model be deployed in a live testing environment.

[5]Please see the appendix for detailed results for both the baseline and principal model across all training and testing sets.

Binary Classification

The baseline model's performance on the binary classification set was impressive considering the lack of any feature extraction in the model. The baseline correctly classified both the benign and malicious examples with 93% recall, with 96% and 86% precision on benign and malicious examples, respectively (see Fig. 5), for a global weighted F1 score of 93% and 92.7% global accuracy.

The principal model performed only slightly worse than the baseline on the task of binary classification, correctly identifying 93% of malicious examples, with a lower rate of relevance at 79%, meaning more false positives were produced. Overall, the principal model achieved a weighted F1 score of 90% and an accuracy of 89.5%.

Balanced Binary Classification

The baseline classifier performed nearly as well on the balanced binary set, with 95% precision and 90% recall on benign examples and 91% precision and 95% recall on malicious examples, settling overall with a weighted F1 score of 93% and 92.8% global accuracy (see Fig. 6).

The principal model again was outperformed by the baseline model, although in this case the principal results were arguably the closest to the baseline among all datasets. Identical to the baseline, 94% recall was achieved on malicious traffic, but with a notably worse 84% precision rate. The baseline however far outperformed the principal in classification of benign traffic, with the principal achieving 82% recall and 93% precision, achieving both a global weighted F1 score and global accuracy of 88%. It is worth noting that the balanced binary dataset contained significantly fewer examples, as achieving a 1:1 ratio between benign and malicious traffic led to nearly 35% fewer examples to learn from, as only a total of 556556 malicious examples existed in the overall dataset. However, the baseline model was still trained on the same dataset and achieved a higher overall accuracy and F1 score.

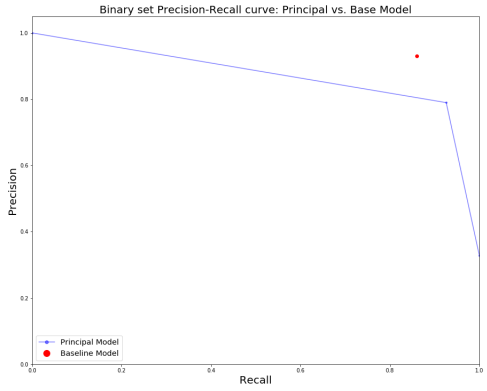

Figure 5: Precision-Recall comparison between the principal model (blue curve) and the base model (red point) for the binary classification task.

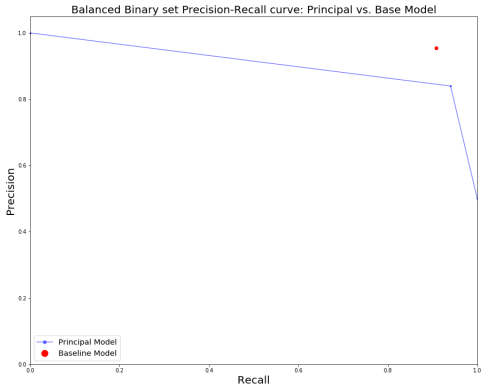

Figure 6: Precision-Recall comparison for the balanced binary classification task.

Meta-Class Classification

The baseline model's performance began to slip once introduced to a non-binary classification problem. The meta-class dataset contained 6 malicious target categories alongside a benign category, and although it performed exceedingly well on correctly classifying benign data with a class-based F1 score of 98%, there were wide discrepancies between malicious classification scores. Both the Denial of Service (DoS) and the PortScan attacks were classified with 99% recall (note that recall is considered the paramount metric for these experiments), the web attack meta-category was completely undetected with 0% classification accuracy, recall, and precision. This may be due to the web-based nature of the attacks, such that packet payloads (which were not included in the dataset) may have revealed more information regarding the true nature of the attacks. With just 9 examples to learn from, only 33% of the infiltration attacks were correctly identified, although with 100% precision, meaning that there were no false positives in this category.

The principal model also suffered once extended beyond binary classification, though fared far worse than the baseline model, having incorrectly classified the entirety of not just the Web Attack category, but also the Infiltration and Botnet examples. Among its best performing classes were the PortScan and DoS attacks at 99% and 95% recall, respectively, though producing more false positives in both categories than the baseline model.

Multi-Class Classification

Finally, the baseline model's limitations were much more apparent in the multi-class problem, where 14 classes of malicious traffic were interspersed with benign traffic. Again, the model was able to achieve an F1 score of 98% on the benign data, though performance varied enormously across malicious target classes. Among the most easily detected malicious examples were the FTP-Patator brute force, Hulk DoS, DDoS, Heartbleed, and PortScan attacks, all of which achieved higher than 90% accuracy and recall. Among the more surprising results were the 100% recall scores for the Heartbleed attack, as the Heartbleed attack represented less than 0.001% of the training set examples.

The principal model suffered even more in this case, incorrectly classifying the entirety of 6 malicious classes, compared to the baseline model's 4. Among these 6 classes, the Heartbleed attack went entirely undetected, whereas the baseline model was able to correctly identify 100% of such examples. Furthermore, the principal model obtained lower precision, recall, and accuracy across all categories benign and malicious, leading to the objective conclusion that the baseline model outperforms the RBM-based model in every category of multi-class classification.

5.1 Discussion

The purpose of an energy-based anomaly detection model in this domain is principally to develop a tool that can reliably detect anomalies that it has never encountered before, primarily to protect a network against novel or zero-day attacks[6]. Although the principal model was unable to outperform the baseline classification model in every single testing instance, the results of this thesis do not conclusively indicate that such models are ineffective in this domain. Several considerations need to be made, notably the synthetic nature of the generation of the dataset. As stated previously, cybersecurity datasets are often generated in sterile conditions due to the implications of initiating various cyberattacks on a live net-

[6]Defined as a weakness or gap in security exploited before developers or security administrators become aware that such a weakness even exists.

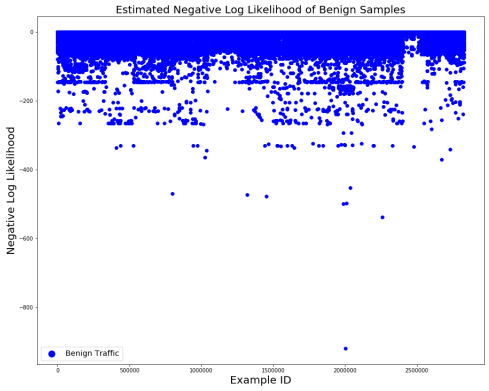

Figure 7: Negative log likelihood of benign traffic.

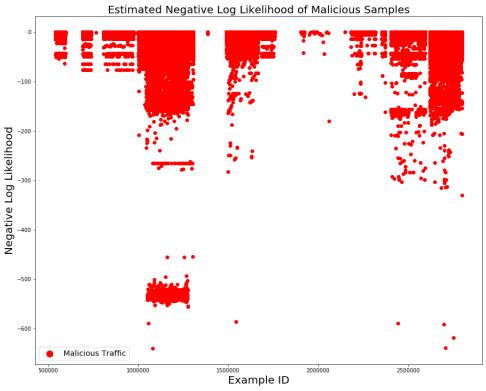

Figure 8: Negative log likelihood of malicious traffic.

work with real users. The benign traffic in the dataset used was synthetically generated based on known patterns, but was hardly exhaustive.

Furthermore, due to the closed nature of the environment in which the dataset was generated, a limited number of traffic sources and destinations existed, such that there was great overlap in the directionality of packet flow for both benign and malicious traffic [14]. Such overlaps are unrealistic, as a single victim's address was the primary target of multiple attacks, and furthermore the source of multiple attacks originated from a single attacker address. Additionally, the packet payload contents were omitted from the dataset, which may have revealed useful information regarding the nature of the traffic to the learning model. However, it is not uncommon for packet contents to be encrypted in a live setting, so the contents of the packets may only provide marginal benefit to the model.

Future research may be conducted with more balanced datasets with respect to malicious examples, as class imbalance may have played a significant part in the significant decline in accuracy for the meta and multi-class classification tasks. However, care must be taken as balanced datasets may not accurately represent real-world conditions. Such real-world conditions may prove to be significantly more fruitful for models such as the principal model explored herein, due to significantly higher volumes of traffic and less synthetically generated traffic. In addition, the exploration of more modern machine learning models, such as variational autoencoders and de-noising autoencoders may prove more capable; de-noising autoencoders especially, due to the highly "noisy" nature of internet traffic. Indeed, figures 7 and 8 show the contrast in negative log likelihood of benign and malicious traffic, as malicious traffic adheres to more discernible patterns. Machine learning models that are better suited to dimensionality reduction with more stable results than the RBM may be able to better discern dimensional boundaries between malicious categories and benign traffic. Deep learning models may also be worth exploration, as a single layer RBM was implemented for this thesis. Additional layers in the learning process may be able to better reduce dimensionality and abstractify patterns that a single-layer model may not be capable of.

6 Conclusion

In this thesis, an energy-based network anomaly detection model was implemented with a Restricted Boltzmann Machine coupled with a discriminative classifier for the classification of various types of network traffic. From a high-quality cybersecurity dataset, several training and testing sets were constructed to compare the efficacy of the principal model's energy-based approach with a more "traditional" logistic regression model. Explicit care was taken to tune the hyperparameters of both models to represent a well-engineered model that might be deployed on a live network. Sterile experimental conditions were constructed to emphasize the minutiae of the model's capabilities with respect to various classification tasks of high-dimensional data.

Experiment results were relatively inconclusive, with the principal model performing nearly identically to the baseline model on binary classification tasks, with performance of both models deteriorating with classification tasks of higher dimension. The principal model never outperformed the baseline model in any experiment, however, and warrants further investigation with respect to hyperparameter tuning, data engineering, and training procedures.

Future research however may be directed at more modern generative models such as autoencoders and generative adversarial networks, as the instability of the Restricted Boltzmann Machine model may prove to be an inherent handicap for such sensitive cybersecurity applications.

References

[1] T. Aldwairi et al. "An Evaluation of the Performance of Restricted Boltzmann Machines as a Model for Anomaly Network Intrusion Detection." In: *Computer Networks* 144 (2018), pp. 111–119.

[2] J.P. Anderson. "Computer Security Threat Monitoring and Surveillance, Technical Report." In: *Computer Security Division of the Information Technology Laboratory* (1980).

[3] Ugo Fiore et al. "Network Anomaly Detection with the Restricted Boltzmann Machine." In: *Neurocomputing* 122 (2013), pp. 13–23. DOI: 10.1016/j.neucom.2012.11.050.

[4] A. Gharib et al. "An evaluation framework for intrusion detection dataset." In: *2016 International Conference on Information Science and Security (ICISS)* (2016), pp. 1–6.

[5] Ian Goodfellow et al. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[6] G.E. Hinton. "A Fast Learning Algorithm for Deep Belief Nets." In: *Neural Computation* 18 (2006), pp. 1527–1554. DOI: 10.1162/neco.2006.18.7.1527.

[7] G.E. Hinton. "Training Product of Experts by Minimizing Contrastive Divergence". In: *Neural Computation* 14.8 (2002), pp. 1771–1800.

[8] Geoffrey E. Hinton. "A Practical Guide to Training Restricted Boltzmann Machines." In: *Lecture Notes in Computer Science Neural Networks: Tricks of the Trade* (2010), pp. 599–619.

[9] G.E. Hinton et al. "A learning algorithm for Boltzmann machines." In: *Cognitive Science* 9.1 (1985), pp. 147–169. DOI: 10.1207/s15516709cog0901_7.

[10] G.E. Hinton et al. "Reducing the Dimensionality of Data with Neural Networks." In: *Science* 313 (5786) (2006), pp. 504–507. DOI: 10.1126/science.1127647.

[11] *KDD Cup 1999 Data*. URL: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. (accessed: 17.06.2019).

[12] S. Kullback et al. "On Information and Sufficiency." In: *Annals of Mathematical Statistics* 22 (1951), pp. 79–86. DOI: 10.1214/aoms/1177729694.

[13] J.-S.R. Lee et al. "Self-Similar Properties of Spam." In: *Proceedings of the Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, IMIS, IEEE* (2011), pp. 347–352.

[14] R. Panigrahi et al. "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems". In: 7 (Jan. 2018), pp. 479–482.

[15] Scikit-Learn. *Documentation: Preprocessing - MinMaxScaler*. URL: https://www.scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. (accessed: 13.02.2019).

[16] Iman Sharafaldin et al. "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization." In: *4th International Conference on Information Systems Security and Privacy (ICISSP)* (2018).

[17] A. Shiravi et al. "Toward developing a systematic approach to generate benchmark datasets for intrusion detection." In: *Computer Security* 31 (2018), pp. 357–374.

[18] Skymind. *A Beginner's Guide to Restricted Boltzmann Machines*. URL: https://www.skymind.ai/wiki/restricted-boltzmann-machine. (accessed: 21.05.2019).

[19] Paul Smolensky. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Chapter 6: Information Processing in Dynamical Systems: Foundations of Harmony Theory*. MIT Press, 1986, pp. 194–281. ISBN: 0-262-68053-X.

[20] James Somers. *rogress in AI Seems like It's Accelerating, but Here's Why It Could Be Plateauing*. URL: www.technologyreview.com/s/608911/is-ai-riding-a-one-trick-pony/. (accessed: 12.06.2019).

[21] Chris Sorensen. *How U of T's 'Godfather' of Deep Learning Is Reimagining AI*. URL: www.utoronto.ca/news/how-u-t-s-godfather-deep-learning-reimagining-ai. (accessed: 11.06.2019).

[22] T. Tieleman. "Training Restricted Boltzmann Machines Using Approximations to the Likelihood Gradient." In: *Proceedings of the 25th International Conference on Machine Learning - ICML 08, 2008* (2008). DOI: 10.1145/1390156.1390290.

Appendix

Table 1: CICIDS2017: including the number of examples for each class and the encoded class labels in parentheses.

Num. Examples	Class Label	Meta-class	Class Description
2271320	Benign (0)	Benign (0)	Benign background traffic of 25 simulated users based on the HTTP, HTTPS, FTP, SSH, and email protocols [13, 16, 17]
7935	SSH-Patator (1)	Brute Force (1)	Brute-force login attempt via Secure Shell (SSH)
5897	FTP-Patator (2)	Brute Force (1)	Brute-force login attempt via File Transfer Protocol (FTP)
5796	DoS-slowloris (3)	DoS (2)	Denial of Service (DoS) via slowloris (partial syn-ack request to server left open indefinitely)
5499	DoS-Slowhttptest (4)	DoS (2)	DoS via slow http request (leaving connection to server open for long periods of time)
230124	DoS-Hulk (5)	DoS (2)	DoS via high-volume artificial traffic directed at server
10293	DoS-GoldenEye (6)	DoS (2)	DoS via attempting to fill every available socket in server
128025	DDoS (7)	DoS (2)	DoS via utilization of distributed connections from various locations, such as a botnet
11	Heartbleed (8)	Infiltration (3)	CPU cache exploitation via sending erroneously formed requests
36	Dropbox Infiltration (9)	Infiltration (3)	Phishing attempt via dropbox download link of malicious file
1507	Brute Force (10)	Web Attack (4)	Brute-force password cracking attempt via http
652	XSS (11)	Web Attack (4)	Cross-site scripting for script injection
21	SQL-injection (12)	Web Attack (4)	Database language request injection to return database contents without permission
1956	Botnet ARES (13)	Botnet (5)	Network of devices operated by attacker for various malicious purposes
158804	PortScan (14)	PortScan (6)	Scanning available ports to discern server/device information, possibly for later attack

Table 2: Binary classification results of baseline and principal model.

Label	Precision		Recall		Accuracy		F1-Score		Examples
	Baseline	Principal	Baseline	Principal	Baseline	Principal	Baseline	Principal	
Benign (0)	0.96	0.96	0.93	0.88	0.93	0.88	0.94	0.92	227133
Malicious (1)	0.86	0.79	0.93	0.93	0.93	0.93	0.89	0.85	111311
Weighted Avg.	**0.93**	0.90	**0.93**	0.89	**0.93**	0.90	**0.93**	0.90	338444

Table 3: Balanced binary classification results of baseline and principal model.

Label	Precision		Recall		Accuracy		F1-Score		Examples
	Baseline	Principal	Baseline	Principal	Baseline	Principal	Baseline	Principal	
Benign (0)	0.95	0.93	0.90	0.82	0.90	0.82	0.93	0.87	111311
Malicious (1)	0.91	0.84	0.95	0.94	0.95	0.94	0.93	0.89	111312
Weighted Avg.	**0.93**	0.89	**0.93**	0.88	**0.93**	0.88	**0.93**	0.88	222623

Table 4: Meta-class classification results of baseline and principal model.

Label	Precision		Recall		Accuracy		F1-Score		Examples
	Baseline	Principal	Baseline	Principal	Baseline	Principal	Baseline	Principal	
Benign (0)	0.99	0.97	0.96	0.92	0.95	0.92	0.98	0.94	227133
Brute Force (1)	0.90	0.97	0.78	0.32	0.29	0.32	0.84	0.48	2766
DoS (2)	0.94	0.87	0.99	0.94	0.96	0.94	0.96	0.90	75948
Infiltration (3)	1.00	0.00	0.33	0.00	0.00	0.00	0.50	0.00	9
Web Attack (4)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	436
Botnet (5)	0.80	0.00	0.02	0.00	0.003	0.00	0.04	0.00	391
PortScan (6)	0.89	0.80	0.99	0.99	0.99	0.99	0.94	0.89	31761
Weighted Avg.	**0.97**	0.93	**0.97**	0.93	**0.95**	0.93	**0.97**	0.92	338444

Table 5: Multi-class classification results of baseline and principal model.

Label	**Precision**		**Recall**		**Accuracy**		**F1-Score**		**Examples**
	Baseline	Principal	Baseline	Principal	Baseline	Principal	Baseline	Principal	
Benign (0)	0.99	0.94	0.97	0.95	0.97	0.95	0.98	0.95	227133
SSH-Patator (1)	0.95	0.00	0.48	0.00	0.48	0.00	0.64	0.00	1180
FTP-Patator (2)	0.89	0.94	1.00	0.16	0.99	0.15	0.94	0.27	1587
slowloris (3)	0.89	0.91	0.54	0.51	0.54	0.51	0.67	0.65	1159
Slowhttptest (4)	0.88	0.72	0.89	0.19	0.89	0.20	0.88	0.31	1100
Hulk (5)	0.97	0.91	0.99	0.80	0.99	0.80	0.98	0.85	46025
GoldenEye (6)	0.95	0.80	0.87	0.65	0.87	0.65	0.91	0.71	2059
DDoS (7)	0.97	0.84	1.00	0.85	0.99	0.85	0.99	0.85	25605
Heartbleed (8)	0.50	0.00	1.00	0.00	1.00	0.00	0.67	0.00	2
Dropbox (9)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7
Brute Force (10)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	301
XSS (11)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	130
SQL-injection (12)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4
Botnet ARES (13)	0.77	0.00	0.03	0.00	0.03	0.00	0.05	0.00	391
PortScan (14)	0.88	0.80	1.00	0.99	0.99	0.99	0.94	0.89	31761
Weighted Avg.	**0.97**	0.91	**0.97**	0.91	**0.97**	0.91	**0.97**	0.91	338444