Black Box Attacks on Deep Anomaly Detectors

Aditya Kuppa
University College, Dublin
Symantec Corporation
aditya.kuppa@ucdconnect.ie

Slawomir Grzonkowski
Symantec Corporation
slawomir_grzonkowski@symantec.com

Muhammad Rizwan Asghar
The University of Auckland
r.asghar@auckland.ac.nz

Nhien-An Le-Khac
University College, Dublin
an.lekhac@ucd.ie

ABSTRACT

The process of identifying the true anomalies from a given set of data instances is known as anomaly detection. It has been applied to address a diverse set of problems in multiple application domains including cybersecurity. Deep learning has recently demonstrated state-of-the-art performance on key anomaly detection applications, such as intrusion detection, Denial of Service (DoS) attack detection, security log analysis, and malware detection. Despite the great successes achieved by neural network architectures, models with very low test error have been shown to be consistently vulnerable to small, adversarially chosen perturbations of the input. The existence of evasion attacks during the test phase of machine learning algorithms represents a significant challenge to both their deployment and understanding.

Recent approaches in the literature have focused on three different areas: (a) generating adversarial examples in supervised machine learning in multiple domains; (b) countering the attacks with various defenses; (c) theoretical guarantees on the robustness of machine learning models by understanding their security properties. However, they have not covered, from the perspective of the anomaly detection task in a black box setting. The exploration of black box attack strategies, which reduce the number of queries for finding adversarial examples with high probability, is an important problem.

In this paper, we study the security of black box deep anomaly detectors with a realistic threat model. We propose a novel black box attack in query constraint settings. First, we run manifold approximation on samples collected at attacker end for query reduction and understanding various thresholds set by underlying anomaly detector, and use spherical adversarial subspaces to generate attack samples. This method is well suited for attacking anomaly detectors where decision boundaries of nominal and abnormal classes are not very well defined and decision process is done with a set of thresholds on anomaly scores. We validate our attack on state-of-the-art deep anomaly detectors and show that the attacker goal is achieved under constraint settings. Our evaluation of the proposed approach shows promising results and demonstrates that our strategy can be successfully used against other anomaly detectors.

CCS CONCEPTS

• **Computing methodologies → Anomaly detection**; *Feature selection*; Unsupervised learning; Neural networks; Machine learning algorithms.

KEYWORDS

Black box attacks, Anomaly detection, Neural networks

ACM Reference Format:
Aditya Kuppa, Slawomir Grzonkowski, Muhammad Rizwan Asghar, and Nhien-An Le-Khac. 2019. Black Box Attacks on Deep Anomaly Detectors. In *Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES 2019) (ARES '19), August 26–29, 2019, Canterbury, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3339252.3339266

1 INTRODUCTION

Anomaly detection (a.k.a. outlier detection) [11, 18] aims to identify specific observations in a dataset, which are rare and significantly deviate from the remaining observations. In the context of cybersecurity, anomaly detection has been used in a number of applications including malware detection, Intrusion Detection System (IDS), log and sensor data analysis, user authentication, social behavior analysis, and discovering novel attacks.

One of the main tasks of an anomaly detector is to learn a hard decision boundary to separate anomalies from nominals. Unlike other domains, cybersecurity applications often face with adaptive adversaries, who actively adapt and modify their methods depending the detection method. This could cause the anomaly detector to make a mistake. It also makes learning the decision boundary for an anomaly detector even harder and may be prone to false positives.

Deep Neural Networks (DNNs) have been consistently successful in achieving state-of-the-art performance on a wide range of complex problems. Deep learning models capture the complex structure of training data and scale well to large datasets. The automatic feature learning capability helps to learn the unclear class boundaries between normal and anomalous behavior when compared to their shallow counterparts. In recent years, Deep learning-based Anomaly Detection (DAD) algorithms have increasingly become popular and have been widely applied for solving a diverse set of tasks in cybersecurity domain such as network and host intrusion detection, malware detection and log data anomaly detection, etc.

System call logs generated by Host-based Intrusion Detection Systems (HIDS) can be interpreted as a set of sequences and these sequences are used to train a Long Short Term Memory Network (LSTM) for discovering anomalies [8, 13, 34, 51]. The network traffic flow characteristics from Network Intrusion Detection Systems (NIDS) can be translated as time series data to train a DNN to find anomalous flows [22, 31, 41]. Log data, when inferred as a natural language sequence, has shown to be very effective in detecting outliers. Mapping the binary byte code as sequences and using these sequences for training a DNN, is one of common approach followed in literature for malware detection applications [15, 48]. We refer interested readers to dedicated surveys [10, 24] for a broad overview of the applications of DAD in the field of cybersecurity.

As a consequence, cybersecurity related applications based on machine learning/deep learning algorithms today have to take into account adversarial attacks to ensure their robustness. In literature, a large number of adversarial example based evasion attacks have been proposed for image classification, object detection, image segmentation, speech recognition, Intrusion Detection Systems (IDS) [27], malware detection systems [19, 42], malicious behavior detection [1] and digital forensics tools [2]. The body of work on attacks and defenses for machine learning systems is extensive and we refer to recent surveys [7, 29, 36] for a broader overview.

Many of the proposed attacks are classified as a white box, grey box and black box depending on the definition of the threat model considered. White box attacks may help system designers to understand the security limitations of models, but are unrealistic in a more practical setting. Typical proprietary learning systems, users including attackers have only access to input-output pairs of the underlying system, often through a rate-limiting and licensed binary or API. The query interface to models only provides users with the final decision, and sometimes not even output probabilities for the input. Attacking black box models in this scenario is the most challenging for adversaries [9].

Most of the proposed black box attacks are evaluated under the assumption that an attacker can perform a large number of queries to the black box model, i.e., query budget is not considered when designing the attack. These attacks ([35, 37]) exploit the property of *transferability* of adversarial examples. A surrogate model is trained at attacker end from the output of queries and can be used to attack the target model to which the adversary has no direct access. Recent attacks such as Chen et al. [14] have used finite difference methods in order to estimate gradients in the black box case, but are still expensive, requiring millions of queries to generate an adversarial example. The exploration of black box attack strategies, which reduce the number of queries for finding adversarial examples with high probability, is an important problem. Compared to image, text, and speech domains, where domain constraints are limited and models operate on raw pixels, in cybersecurity domain, an adversary has to ensure functionality of underlying data (i.e., valid file, network data) has to be preserved for attack success. This domain constraint makes the attacker job even harder.

In this work, we focus on query optimized black box attacks on anomaly detectors. In a black box setting, attackers do not have full knowledge of the model but only interact via a query interface and each query result consists of binary output. Since queries to the model are costly, attackers are motivated to minimize the number of queries needed when interacting with the model. To evade a black box anomaly detector an attacker has to first discover the complex class boundaries between normal and anomalous data points and generate adversarial examples, which bypass various thresholds set for the decision-making process, respecting the query limits and domain constraints.

Contributions. We perform security evaluation of deep anomaly detectors in a black box setting with a realistic threat model:

- We propose a novel black box attack, which leverages Manifold Approximation Algorithm for query reduction and generates adversarial examples using spherical local subspaces.
- We evaluate the attack on 7 state-of-the-art anomaly detectors with different underlying architectures including Deep Autoencoding Gaussian Mixture Model (DAGMM), AutoEncoder (AE), AnoGAN, Adversarially Learned Anomaly Detection (ALAD), Deep Support Vector Data Description (DSVDD), One Class Support Vector Machines (OC-SVM), and Isolation Forests (IF).
- We present empirical evidence in support of our insights and algorithm on the latest IDS benchmark dataset.

The rest of the paper is organized as follows. We review related work in Section 3. and provide background information in Section 2. We describe the methodology and elaborate on our proposed attack in Section 4. Description of datasets, experiment setup and results of our experiments are covered in Section 5. Finally, Section 6 concludes our work and gives some recommendations for further research.

2 BACKGROUND

In this section, we describe the motivation behind studying black box attacks with query constraints and give a brief overview of the threat model for the proposed attack.

2.1 Motivation

We explain the motivation of our line of research with a running example. Let us consider Cloud Service Providers (CSP), which offer security services as an add-on to their cloud platform products. This often includes services such as IDS systems on pre-configured Virtual Appliances. The IDS may use a learning system to discover anomalous traffic. For a user and attacker alike, the underlying learning system of IDS is a black box. Users of the system route the network traffic via API provided by the CSP and receive a binary decision about a network flow record whether the flow is anomalous or not. The input to the system is a network capture file (i.e., pcap) or live traffic. The learning system has query limits in place to make sure it is not misused.

In a white box setting [5, 7, 20, 37], attackers are assumed to know everything about the target system. In this example, realistic white box attacks can be mounted by CSP to test the security robustness of learning system, as it has full access to model parameters, training data, and thresholds etc.

Gray box settings [7] assume the attacker knows the features and the architecture of the underlying learning system, but not

Table 1: Threat model of the proposed attack.

Threat Model Characteristic	Type	Attacker View
Attacker's Knowledge	Training data	✗
	Feature set	✓
	Feature extractor	✓
	Feature transformers	✓
	Learning algorithm	✗
	Objective function	✗
	Parameters and hyper-parameters	✗
	Inference API	✓
	Error/debug interface	✗
	Confidence intervals	✗
Attacker's Goal	Compromising integrity (Evasion)	✓
	Compromising availability	✗
	Compromising privacy	✗
Attacker's Capability	Manipulate training data	✗
	Manipulate test data	✓
	Manipulate model	✗
	Manipulate software/hardware	✗
Attacker's Strategy	Train a surrogate model for parameter extraction	✗
	Train a surrogate model for query reduction	✓
	Satisfy domain constraints	✓

the training data used or the learning system parameters. These attacks can be performed by malicious users of the system. In a typical attack, attackers can collect a surrogate dataset, in this case, network capture files, and query the system and get feedback from the IDS system. One can use decisions from the learning system to label the surrogate dataset and train a surrogate (local) model on the labeled surrogate dataset. This method exploits the property of *transferability* between learning algorithms to generate valid adversarial samples for an IDS learning system.

In a realistic black box setting, an attacker has very limited knowledge about the feature space, learning algorithm and the training data [7, 37]. These attacks can be performed by malicious users of the system who can query the system in a black box manner and get feedback on the provided labels or confidence scores. Attackers understand the purpose, raw input, output and domain constraints of the system. In our example, a malicious user knows that the raw input to the learning system is network traffic and the purpose of the system is to discover anomalous network flows, and a malformed/bad network packet may never reach the learning system for analysis.

In real-world settings, the efficacy of attacks on black box learning systems is highly limited by the presence of rate limiting and domain constraints. This motivated us to propose a novel attack, which achieves the attacker goals respecting all the black box properties and domain restrictions.

2.2 Threat Model

Threat models against machine learning system [5, 7, 20, 37] can be defined by understanding (a) Attacker's Knowledge, (b) Attacker's Goal, (c) Attacker's Capability, and (d) Attacker's Strategy.

Our attack aims to compromise the integrity of anomaly detectors by reducing the confidence and modifying the input (an anomalous sample) in order to output (nominal class) by the detector. The attacker can have different levels of knowledge of the targeted system, including but not limited to: (a) training data, (b) feature set, (c) feature extractors, (d) learning algorithm with the objective function and its parameters/hyper-parameters, (e) inference API, in case of Machine Learning as a Service (MLaSS), (f) error/debug interface. An attacker's capability depends on the influence she has on input data (i.e., training and testing or only testing) and domain-specific constraints [7].

The effectiveness of an attack is given by the objective function of the attack strategy. In our proposed attack, Attack Success Rate (ASR) is measured by the ratio of samples that meets the adversary's goal:

$$\min Q(\mathbf{x}) \qquad \text{s.t. } f(x_{\text{adv}}) \neq y_t$$

where $Q(\mathbf{x})$ denotes the total number of queries needed for finding an adversarial example for seed sample x, y_t represents the original label of instance x with threshold t, and x_{adv} refers to the adversarial instance. Table 1 summarizes the threat model for the proposed attack.

3 RELATED WORK

Prior works on learning systems in a black box setting are categorised into surrogate model attacks, approximation based attacks, and optimization based attacks.

Szegedy et al. [53] showed that adversarial examples are transferable between models, which enabled attacks on deployed models in black box settings. Transfer-based attacks use query responses obtained from the target model to train a substitute model [17, 33, 35, 37, 42] and then generate adversarial examples for the surrogate model, which are effective against target model. The main drawback of this method is transfer loss, not all adversarial examples can transfer from one model to another model. In order for attack to be effective, the number of training samples needed to train the substitute model may be very large [21].

Another approach, which uses an approximation [14] method for estimating the gradient of the black box. This method requires a large number of queries since each step requires gradient estimation which in turn need to query target model. Boundary attack based[52] uses a Bayesian optimization technique which starts with a large adversarial perturbation sample and reduces the norm of the perturbation in an iterative manner. This approach assumes the underlying data distribution is Gaussian and may not scale for high dimension data.

In the cybersecurity domain, multiple attacks have been proposed for learning systems. By injecting *chaff* traffic Rubinstein et al. [44] poisoned a PCA based anomaly detector trained on network traffic. Newsome et al. [33] devised an attack to mislead signature generation for malware detection. Nelson et al. [32] bypassed spam filters by learning spam emails containing good words during training. In their threat model, they assume the attacker can control the training data, which is not the case in our system. Zhou et al. [57] used a surrogate SVM model to generate malicious spam to bypass a spam detection model.

Functionality preserving attacks on malware detectors have been proposed by Grosse et al. [16] by applying forward derivative of the attacked neural networks to craft adversarial android malware. Hyrum et al. [4] use reinforcement learning approach for generating adversarial malware examples. Rosenberg et al. [42] proposed a generic framework based on API call sequences and static features for malware classifiers. Hu et al. [19] proposed a Generative Adversarial Network (GAN) framework to generate adversarial malware examples for the black box attacks. In another work, Hu et al. [19] used a surrogate model to bypass Recurrent Neural Networks (RNN) detection systems. Al-Dujaili et al. [3] proposed four methods to generate binary-encoded adversarial malware examples with the preserved malicious functionality and utilized the SLEIPNIR to train the robust detectors.

Most of the proposed attacks, which train a surrogate model to achieve attacker goals and query limited scenario is not explicitly considered in any of the attacks. The query optimised black box attack was first studied by Li et al. [25] who consider the spam email setting and set a bound on the total number of queries and feature modification cost. The attacker then applies a query strategy to find adversarial examples for linear classifiers.

Most black box attacks usually require a large number of queries to the remote model, while this is an important constraint in practice. Li et al. [39] introduced an active learning strategy to significantly reduce the number of queries for training a surrogate classifier. Ilyas et al. [21] applied natural evolution strategies and only used two to three orders of magnitude fewer queries than previous methods. Aditya et al. [2] used pixel influence to reduce the number of queries. Bhagoji et al. [6] proposed random feature grouping and principal component analysis, which can achieve both high query efficiency and high success rate. However, these works do not explicitly consider minimizing the total number of queries and do not take into account the realistic threat model where satisfying domain constraints is important and has practical implications.

Our problem setting closely resembles query synthesis methodology in the field of active learning [12], which is similar to an adversary exploiting query interface to learn more about the proprietary model or perform an evasion attack to misclassify the test samples.

In this paper, our proposed method works on both linear and non-linear models including neural networks. To the best of our knowledge, this work is the one to study security robustness of deep anomaly detectors with realistic threat model in the cybersecurity domain.

4 OUR METHODOLOGY

Before describing our attack methodology we highlight some interesting properties of adversarial subspaces (the data manifold in which adversarial examples reside) found in the literature [30, 43, 49, 54, 55]. Adversarial subspaces are: (a) compared to data manifold, they are found in low probability; (b) class distributions generally vary from their closest data sub-manifold; (c) adversarial samples are found closer to nearest neighbour proximity to the unperturbed sample than to any other neighbour in the training or test set; and (d) for any two points x and y on a sphere,

$$d_\infty(x, y) \le d_2(x, y) \le d_g(x, y),$$

where $d_\infty(x, y)$, $d_2(x, y)$, and $d_g(x, y)$ denote the l_∞, Euclidean, and geodesic distances, respectively. Any ϵ-adversarial example in the geodesic metric would also be adversarial in l_∞ and euclidean metrics.

In our threat model, the attacker has limited number of queries to evade the anomaly detector, i.e., she has to work with a small subset of test samples n and select the samples in an efficient and optimized manner. One can train a surrogate classifier with collected test samples but data n may be limited and also the attacker has no knowledge of distributions of original training data of which the model was trained. Also, typical anomaly detection systems use thresholds to classify a sample as benign or abnormal, which makes the attacker job even harder.

Motivated by attacker constraints and properties of adversarial subspaces, an efficient attack should have the following properties:

- Capture multiple data distributions with limited test data for reducing query count.
- Generalize hyper-planes but preserve the adversarial subspaces.
- Exploit nearest neighbours properties of adversarial examples.
- Easy addition of new samples without retraining, i.e., when an attack is unsuccessful, the attacker can collect more samples and project them to already known subspace to avoid duplicate queries.

As noted in [26] geometric properties of sphere, provide a good generalization of hyper-planes by approximating the local curvature.

Let $X \in \mathbb{R}^{n \times p}$ with rows $x_i \in \mathbb{R}^p$, then the sphere centered at c with radius r is called the spherelet of X, denoted by $s(X) = S(V, c, r)$ where V determines an affine subspace the sphere lies in.

The spherical error [26] of X is defined by

$$\epsilon(X) = \frac{1}{n}\sum_{i=1}^{n}(\|x_i - c\| - r)^2$$

. If x_i lie on the sphere then $\epsilon(X) = 0$. Similarity between two datasets X and Y can be measured in terms of their geometric and topological properties [26]. The distance between $s(X)$ and $s(Y)$ is called spherelet distance[26], which is defined as:

$$d_S = d_R(X, Y) + d_E(X, Y) > \lambda \qquad (1)$$

where d_E is the Euclidean distance between two sets of points, d_R is Riemannian divergence (geometric) of dataset of X and Y, and $\lambda > 0$ is a tuning parameter.

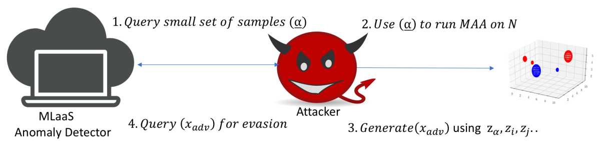
1. Query small set of samples (α)
2. Use (α) to run MAA on N
MLaaS
Anomaly Detector
4. Query (x_{adv}) for evasion
Attacker
3. Generate(x_{adv}) using z_α, z_i, z_j..

Figure 1: Anomaly detection model is trained on data not known to the attacker and hosted by the server with a query interface. Step 1: The attacker queries a small subset of samples α for bootstrapping the attack. Step 2: She uses α to run MAA on N test samples that are collected offline. Step 3: Sample the distributions to solve equation 2 for generating adversarial sample. Step 4: Use the generated x_{adv} to bypass evasion.

With this background, let us explain how our attack works. There are two steps involved in our attack: (a) Query Reduction Step (b) Attack Step. Figure 1 describes the various steps in our proposed attack.

Query Reduction Step First, the attacker queries the remote model with a small subset $\alpha \subset n$ samples, which classify the data into benign and abnormal labelled as 0 and 1. We run Manifold Approximation Algorithm (MAA) [26] on α data points to find the best piecewise spherical manifold or subspace $\hat{M}$ and the projection map $p : \alpha^p \longrightarrow \hat{M}$ such that the mean square error $\sum_{i=1}^{n} \|x_i - p(x_i)\|^2$ is minimized, where $x_i \in \mathbb{R}^\alpha$. In other words, we find various data distributions in the labelled data. MAA has 5 key steps: (a) Normalise the data $x_i \in [-1, 1]^p$; (b) Split data into clusters with λ and ϵ tuning parameters, where λ controls the nearest neighbour distance between points and ϵ thresholds the spherical error before assigning a point to a cluster. This will result in data fitted in each cluster by a cluster-specific sphere; (c) Merge near clusters based on spherelet distance (equation 1); (d) Calculate spherelet properties for each labelled cluster. Figure 2 shows a toy example of spherelets in the test data. Next, for the remaining data points x_i in n - α, we find which local spherical subspace x_i belongs by calculating the projection of x_i onto $\widehat{M}$.

Attack Step At this point, from a small set of query samples α, we partitioned our larger test set n into multiple spherical local subspaces $S_p^i(V_i, c_i, r_i)$ with different data distributions $\boldsymbol{z_1}, \boldsymbol{z_2}..\boldsymbol{z_\alpha}$ and preserve the property of nearest neighbors by λ and ϵ. In order for our attack to be successful, we need to induce minimal distortions on the input distribution $\boldsymbol{z_a}$ to move the decision boundary to $\boldsymbol{z_i}$. To achieve this, we use the data distributions $\boldsymbol{z_1}, \boldsymbol{z_2}..\boldsymbol{z_\alpha}$ to solve Equation 2.

$$\begin{aligned} \min_{\boldsymbol{d}} \quad & \Delta(\boldsymbol{z_a}, \boldsymbol{z_i}) + C\|\boldsymbol{d}\| \\ \text{s.t.} \quad & S_p^a(V_a, c_a, r_a) \le \boldsymbol{x} + \boldsymbol{d} \le S_p^i(V_i, c_i, r_i) \end{aligned} \qquad (2)$$

where C is the regularisation constant that balances approaching the target and limiting the input distortion and KL-divergence as Δ. Algorithm 1 provides the pseudo code for our attack algorithm.

Algorithm 1: Attack Algorithm with Query Reducing Optimization.

```
input  : $\alpha = \{x_i, y_i\}_{i=1}^{a}$; $n = \{x_i\}_{i=1}^{n}$; parameters $\epsilon$, $\lambda$
output : $x_{\text{adv}}$
1  Let $X$= $\alpha$ ;
2  normalise $X$;
3  for $x$ = in : $X$ do
4      assign $x$ to the nearest cluster based on $d_\lambda$;
5      merge($x$,$\epsilon$,$\lambda$,);
6  Let $X$= $n$ - $\alpha$ , $\widehat{M}$ ;
7  for $x$ = in : $X$ do
8      project $x$ to $\widehat{M}$ to find cluster $c_i$;
9      assign $x$ to cluster with the smallest spherical distance.
10 Calculate $S_p^i(V_i, c_i, r_i)$ and $\boldsymbol{z_1}, \boldsymbol{z_2}..\boldsymbol{z_\alpha}$ for each cluster;
11 Solve equation 2 for $x_{adv}$.;
12 Apply domain constraints on $x_{adv}$ ;
```

5 EXPERIMENTAL SETUP

Dataset. One of the major applications of anomaly detection in the field of cybersecurity is IDS [50]. Unfortunately, most of the publicly available datasets lack diversity of the traffic, volumes and real world attacks [50]. We choose the latest IDS2018 benchmark dataset, which addresses the challenges mentioned above. The dataset [1] covers seven different attack scenarios: Brute-force, Heartbleed, Botnet, Denial of Service (DoS), Distributed DoS (DDoS), web attacks, and infiltration of the network from inside. The infrastructure used to generate attack and benign traffic reflects real world setting with 420 victim machines, 30 servers and 50 attack machines. We use *CICFlowMeter-V3* to extract 80 features from the captured traffic. Table 2 describes the size of different attacks in CICFlowMeter-V3 extracted data.

Features are extracted from bidirectional flows. Statistical time-related features are calculated separately for both directions. TCP flows are terminated by FIN packet and UDP flows are terminated by a flow timeout, which is set to 600 seconds. There are 8 group of features that are extracted from raw pcaps: (a) Forward Inter

[1]https://www.unb.ca/cic/datasets/ids-2018.html

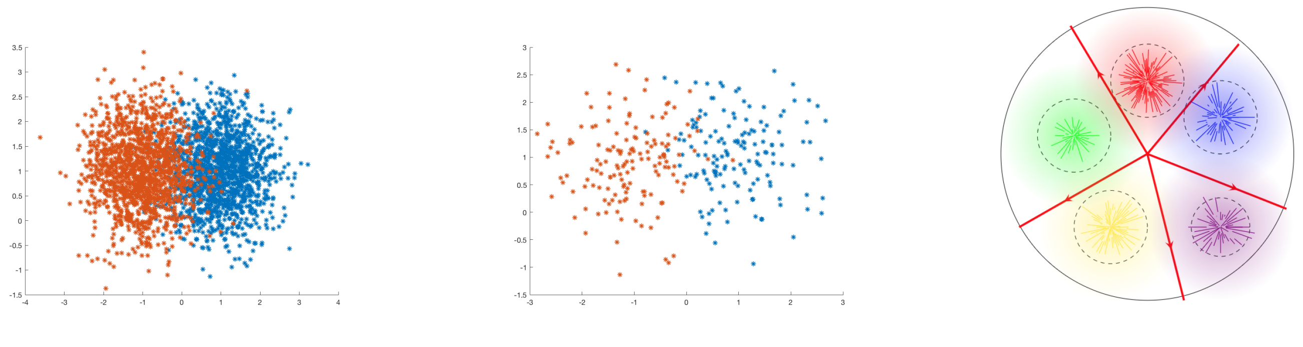

Figure 2: An example of spherelets in the test data: (a) Scatter plot of 6000 data points (x in our methodology) generated with overlapping Gaussian distributions. (b) Scatter plot of randomly selected 300 data points (α in our methodology). (c) All points in x transformed into piece wise local spherical subspaces by applying Manifold Approximation Algorithm (MAA). Circle surrounding all points is unit sphere and smaller circles (in different colors) enclose neighbourhood points with similar spherical distance. Red lines divide the manifold space into multiple data distributions, which are used in the attack generation step.

Table 2: Statistics of the datasets.

Dataset	Total Instances	Percentage
Bot	286191	1.76303%
DoS attacks-Hulk	461912	2.84552%
DoS attacks-SlowHTTPTest	139890	0.861766%
Brute Force - Web	611	0.00376%
Brute Force - XSS	230	0.00142%
SQL Injection	87	0.00054%
FTP-BruteForce	193360	1.19116%
SSH-Bruteforce	187589	1.15560%
DDoS attack-HOIC	686012	4.22605%
DDoS attack-LOIC-UDP	1730	0.01066%
Infilteration	161934	0.99756%
DoS attacks - GoldenEye	41508	0.255702%
DoS attacks - Slowloris	10990	0.067702%
DDoS attacks-LOIC-HTTP	576191	3.54952%
Benign	13484708	83.07001%
All Attacks	2748235	0.16930%
Total Instances	16232943	100%

Arrival Time, the time between two packets sent forward direction (mean, min,max, std); (b) Backward Inter Arrival Time, the time between two packets sent backwards (mean, min, max, std); (c) Flow Inter Arrival Time, the time between two packets sent in either direction (mean, min, max, std); (d) Active-Idle Time, amount of time flow was idle before becoming active (mean, min, max, std) and amount of time flow was active before becoming active (mean, min, max, std); (e) Flags based features – Number of times the URG, PSH flags are set both forward and backward direction; (f) Flow characteristics – bytes per second , packets per second, length of flow (mean,min,max,std) and download and upload ratio of bytes; (g) Packet count with flags – FIN, SYN, RST, PUSH, ACK, URG, CWE and ECE; (h) Average number of bytes and packets sent in forward and backward direction in the initial window, bulk rate, and sub flows.

In case of cybersecurity domain, adversarial samples have to respect domain constrains, i.e., the original functionality of sample has to remain intact when adversarial noise is added. The two way mapping of raw data to the feature space and adversarial noise induced sample in the feature space to raw data has to be preserved. The distortion added should not create malformed packets when features are transformed back to *pcap*. To solve this constraint we selectively add noise to a small set of features instead of perturbing any of 80 features and we use Scapy[2] to generate valid packets for changed values keeping other properties of the flow intact.

We extract time based features from Forward Inter Arrival Time, Backward Inter Arrival Time, Active-Idle Time and the average number of bytes and packets sent in forward and backward directions in initial window and sub flows to induce distortion in equation 2. Once we have feature change value we modify the original *pcap* using Scapy to create new *pcap* file, which has new values for the modified feature. This process helps us to validate our attack, which is operating in the domain constraints and not generating any malformed packets that could be discarded before reaching the learning system.

5.1 Architecture and Training

AutoEncoders (AEs) are widely used for unsupervised learning tasks such as learning deep representations or dimensionality reduction. Typically, a traditional deep autoencoder consists of two components, the encoder and the decoder. Let us denote the encoder's function as $f_\theta : \mathcal{X} \rightarrow \mathcal{H}$, and denote the decoder's function as $g_\omega : \mathcal{H} \rightarrow \mathcal{X}$, where θ, ω are parameter sets for each function, $\mathcal{X}$ represents the data space and $\mathcal{H}$ the feature (latent) space. The reconstruction loss, which is

$$L(\theta, \omega) = \frac{1}{N} \|X - g_\omega(f_\theta(X))\|^2 \tag{3}$$

[2]https://scapy.readthedocs.io/en/latest

where $L(\theta, \omega)$ represents the loss function for the reconstruction.

In our experiments, we train a plain autoencoder, which consists of 5 fully-connected layers with (79, 38, 18, 9, 1) units each. The ReLU activation function is used in all layers, with the exception of the final encoder layer (using linear activation), and the last decoder layer (using sigmoid activation). The latent space is restricted to 1 dimension. This architecture is used in all models that has a auto-encoder component.

Deep Autoencoding Gaussian Mixture Model (DAGMM) [59] uses two different networks, a deep autoencoder and a Gaussian Mixture Model (GMM) based estimator network to determine a sample is anomalous or not. For the experiments, compression network with 3 dimensional input to the estimation network, where one is the reduced dimension and the other two are from the reconstruction error, the compression network uses same architecture as autoencoder described above, and the estimation network performs with fully-connected layer of f(4, 10, tanh)-Drop(0.5)-fully-connected(10, 4, softmax).

AnoGAN [46] is GAN-based method for anomaly detection. The method involves training a DCGAN [40], and at inference time using it to recover a latent representation for each test data sample. The anomaly score is a combination of reconstruction and discrimination components. The original paper [46] compares two approaches for the anomaly score and we adapted the GAN by using fully connected layers. We use DCGAN architecture and hyper-parameters [40] in our experiments.

Adversarially Learned Anomaly Detection (ALAD) [56] is based on bi-directional Generative Adversarial Networks (GANs) anomaly detector, which uses reconstruction errors from adversarially learned features to determine if a data sample is anomalous. It employs spectral normalization and additional discriminators to improve the encoder and stabilize GAN training.

Deep Support Vector Data Description (DSVDD) DSVDD [45], jointly trains a deep neural network while optimizing a data-enclosing hyper-sphere in output space. In our experiments, we used the weights from the trained AE encoder for initialization, thus establishing a pre-training procedure. We use the similar training procedure as described in the original paper[45] in our experiments.

Table 3: Training performance of models on 'Benign vs AllAttacks' dataset.

Dataset	Model	Precision	Recall	F1 Score
Benign vs AllAttacks	OC-SVM	0.8457	0.8523	0.7954
	IF	0.9116	0.9273	0.9194
	DSVDD	0.8921	0.7272	0.7928
	AE	0.8719	0.6246	0.7599
	DAGMM	0.9397	0.9232	0.9269
	AnoGAN	0.9086	0.9197	0.9065
	ALAD	0.9227	0.9377	0.9201

One Class Support Vector Machines (OC-SVM). [47] are kernel based method that learns a decision boundary around normal examples. We use the radial basis function kernel in our experiments. The ν parameter is set to the expected anomaly proportion in the dataset, which is assumed to be known, whereas the γ parameter is set to $1/m$ where m (80) is the number of input features.

Isolation Forests (IF). [28] are a partition-based method which isolates anomalies by building trees using randomly selected split values. We use default parameters provided by the scikit-learn [38] package in our experiments.

Setting of Parameters For the reconstruction anomaly score that is used as detection criterion, the 98% percentile:

$$T_{\text{rec}} = p_{0.98}(L(\mathbf{x}, \mathbf{x}^*)|\mathbf{x} \in \mathbf{X})$$

of reconstruction errors, and the 2% percentile of likelihoods $T_{\text{prior}} = p_{0.02}(f(\mathbf{x})|\mathbf{x} \in \mathbf{X})$ are used. In case of 'Benign vs All Attacks' dataset we set 80% percentile for calculating threshold as this value reflects the true distribution of training set. We use same threshold criteria for all networks in our experiments. For ALAD based models, we use the outputs of the underlined layer in the discriminator for calculating anomaly score as described original paper [56]. Table 3 gives the performance of trained models on 'Benign vs AllAttacks' dataset.

We tune parameters λ and ϵ for attack algorithm via grid search. To solve equation 2 we use SciPy [23] L-BFGS-B [58] optimiser, with initial disturbance sampled from a uniform distribution $\mathcal{U}(10^{-8}, 10^{8})$, 50 corrections on the memory matrix, and termination test tolerance of 10. We set n to 10000 in all our experiments and randomly sampled 2% of each attack type dataset and 8% of benign data from original dataset for n. We discarded n from the training data to reflect the threat model, i.e., attacker has no access to training data. For generating α dataset, i.e., the original query dataset for which attacker has labels we randomly sample 2% of each attack type and 8% of data from n.

5.2 Results and Discussion

Table 4 summarizes attack success rate on models with different α values. The attack success rate increases with α, i.e., a larger number of samples helps to capture the underlying distribution of training data. This observation indicates that our attack strategy of first allocating some small numbers of queries to approximate the manifold of collected test samples and then using the data distributions to generate the adversarial examples is reasonable and effective. Figure 4 shows how a query reduction strategy helps an attacker to run a successful attack independent of underlying thresholds of the detector. Figure 3 compares the proposed method with surrogate model based attacks and optimization based attacks. For the surrogate model, we first query the remote model to train a local classifier and use the local model to generate adversarial samples using standard white-box attack techniques. Our proposed attack performs better when compared with other state of art query based black box attacks.

GAN based models have performed well against the attack when compared with other counterparts. We think one of the reason

Table 4: Results for the proposed attack on state of art deep anomaly detectors on an independent test set with $\alpha = \{1\%, 5\%, 10\%\}$ and n = 10000 samples. Each model was trained and tested with normal (benign) and anomaly (attack class). n was sampled randomly from all dataset and removed from the training set to reflect the real attacker.

Attacked Model	Benign vs AllAttacks	Benign vs Infilteration	Benign vs BruteForce	Benign vs DoS Attack	Benign vs WebAttack	Benign vs Botnet	Benign vs DDoS and PortScan
Attack Success Rate with α= 1 % of n(10000), Number of Queries 100							
AE	0.71	0.609	0.653	0.632	0.679	0.63	0.662
OC-SVM	0.616	0.618	0.697	0.66	0.687	0.699	0.617
IF	0.628	0.65	0.609	0.61	0.628	0.682	0.614
DSVDD	0.687	0.642	0.688	0.697	0.638	0.635	0.621
DAGMM	0.654	0.643	0.671	0.699	0.626	0.641	0.648
AnoGAN	0.593	0.567	0.583	0.602	0.616	0.608	0.608
ALAD	0.571	0.544	0.567	0.552	0.565	0.563	0.569
Attack Success Rate with α= 5 % of n(10000), Number of Queries 500							
AE	0.849	0.873	0.818	0.849	0.817	0.823	0.856
OC-SVM	0.749	0.767	0.804	0.793	0.772	0.84	0.806
IF	0.77	0.725	0.751	0.808	0.757	0.797	0.766
DSVDD	0.829	0.841	0.779	0.784	0.84	0.783	0.827
DAGMM	0.813	0.816	0.836	0.812	0.794	0.762	0.847
AnoGAN	0.609	0.682	0.67	0.689	0.675	0.661	0.661
ALAD	0.674	0.68	0.688	0.679	0.678	0.682	0.680
Attack Success Rate with α =10 % of n(10000), Number of Queries 1000							
AE	0.931	0.917	0.92	0.928	0.912	0.928	0.908
OC-SVM	0.773	0.792	0.753	0.761	0.742	0.782	0.784
IF	0.868	0.855	0.878	0.814	0.848	0.839	0.832
DSVDD	0.854	0.878	0.826	0.848	0.86	0.822	0.878
DAGMM	0.868	0.893	0.888	0.833	0.838	0.803	0.861
AnoGAN	0.756	0.735	0.758	0.743	0.764	0.784	0.792
ALAD	0.737	0.726	0.742	0.752	0.731	0.787	0.748

is the way GAN models score the anomaly data points, they not only consider the reconstruction error from AE but also the feature distances in the scoring logic.

Our proposed attack is effective towards class of anomaly detection algorithms that employ: (a) Fidelity of reconstruction to determine an example is anomalous; (b) variants of Principal Component Analysis (PCA)- AE and GAN; and (c) Methods that learn classification boundary around the normal data (OC-SVM, DSVDD and IF). The attack on a class of algorithms, which use distances to nearest neighbors or clusters in the data to assess if data is anomalous is left for future work.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a new approach of black box attack on deep anomaly detectors with a realistic threat model taking into account the domain constraints. First, we run manifold approximation on samples collected at the attacker end for query reduction and understanding various thresholds set by underlying anomaly detector, then we use spherical adversarial subspaces to generate attack samples. This method is well suited for attacking anomaly detectors where decision boundaries of nominal and abnormal classes are not very well defined and decision process is done with a set of thresholds on anomaly scores.

We validate our attack on popular anomaly detectors and show that the attacker's goal is achieved under constraint settings. Our proposed approach shows promise in our experiments, and it is

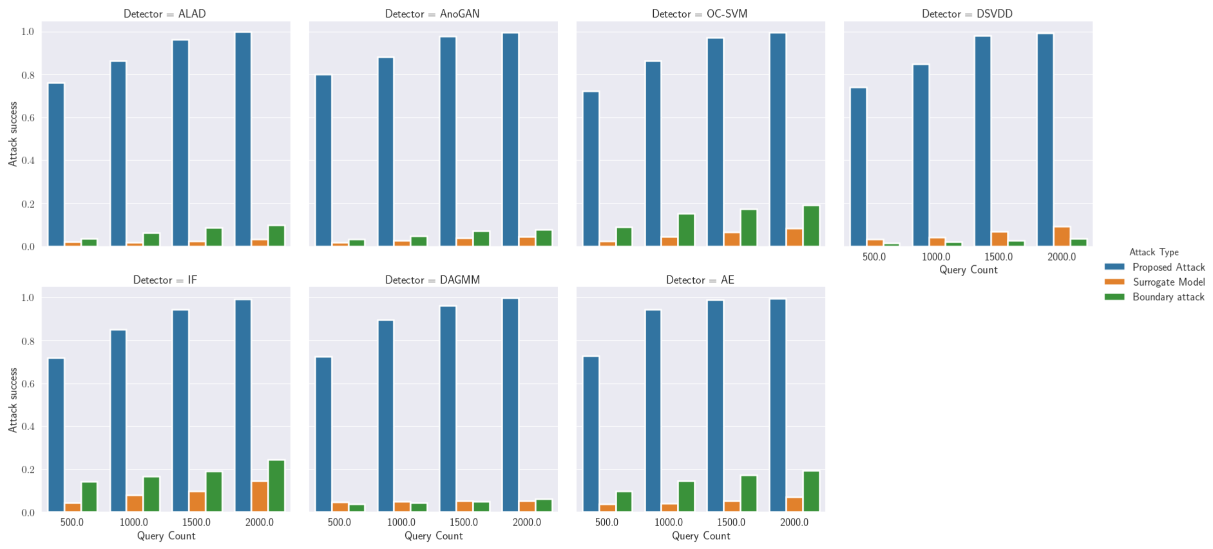

Figure 3: Performance of query based black box attack methods with our proposed method on each model. We discard n = 50000 samples from the training set and use these samples for generating the adversarial sample. For the proposed attack, α was set to 20%.

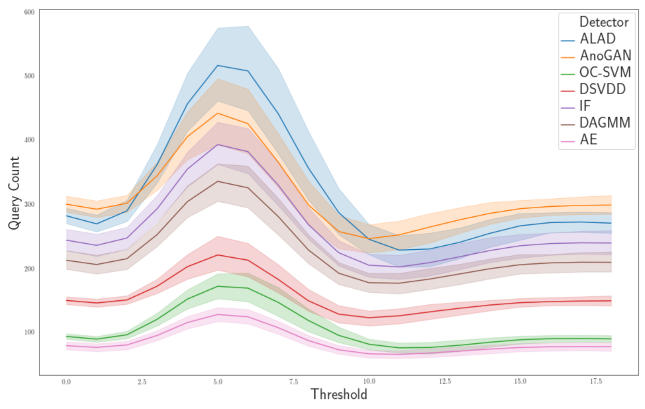

Figure 4: Effect of different threshold values on the number of queries for each anomaly detector. X-axis is threshold set by underlying detector for 'Benign vs AllAttacks' dataset and α=20 % . Y-axis: count of queries to achieve attacker goal. Our attack strategy of first allocating some small numbers of queries (1000) approximate the manifold, which resulted in reduction in query count irrespective of change in threshold values.

a general strategy that can be used against any anomaly detector. There are many scenarios mainly in cybersecurity, where attackers will be constrained by the number of interactions they have with target model, so attacks which reduce the number of queries for finding adversarial examples with high probability is an important problem.

Our future work will focus on three areas: (a) Study the defense mechanisms against the various attacks for anomaly detectors; (b) Extend the proposed attack to unsupervised tasks for, e.g., clustering; and (c) Examine the security robustness of anomaly detectors with different neural network architectures.

REFERENCES

[1] K Aditya, Slawomir Grzonkowski, Muhammad Rizwan Asghar, and Nhien-An Le-Khac. 2019. Finding Rats in Cats: Detecting Stealthy Attacks using Group Anomaly Detection. In *18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom 2019), Roturua, New Zealand*. IEEE, (to appear).
[2] K Aditya, Slawomir Grzonkowski, and Nhien-An Le-Khac. 2018. Enabling Trust in Deep Learning Models: A Digital Forensics Case Study. In *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom 2018), New York, USA*. IEEE, 1250–1255. https://doi.org/10.1109/TrustCom/BigDataSE.2018.00172
[3] Abdullah Al-Dujaili, Alex Huang, Erik Hemberg, and Una-May O'Reilly. 2018. Adversarial deep learning for robust detection of binary encoded malware. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 76–82.
[4] Hyrum S Anderson, Anant Kharkar, Bobby Filar, and Phil Roth. 2017. Evading machine learning malware detection. *Black Hat* (2017).
[5] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D Joseph, and J Doug Tygar. 2006. Can machine learning be secure?. In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*. ACM, 16–25.
[6] Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. 2018. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *European Conference on Computer Vision*. Springer, 158–174.
[7] Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition* 84 (2018), 317–331.
[8] Loic Bontemps, Van Loi Cao, James McDermott, and Nhien-An Le-Khac. 2016. Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks. *Lecture Notes in Computer Science* 10018 (2016), 141–152.
[9] Wieland Brendel, Jonas Rauber, and Matthias Bethge. 2017. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248* (2017).
[10] Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep Learning for Anomaly Detection: A Survey. *arXiv preprint arXiv:1901.03407* (2019).
[11] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 15.
[12] Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. 2018. Model Extraction and Active Learning. *arXiv preprint arXiv:1811.02054* (2018).

[13] Ashima Chawla, Brian Lee, Sheila Fallon, and Paul Jacob. 2018. Host based intrusion detection system with combined cnn/rnn model. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 149–158.

[14] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 15–26.

[15] Alessandra De Paola, Salvatore Favaloro, Salvatore Gaglio, Giuseppe Lo Re, and Marco Morana. 2018. Malware Detection through Low-level Features and Stacked Denoising Autoencoders.. In *ITASEC*.

[16] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. 2016. Adversarial perturbations against deep neural networks for malware classification. *arXiv preprint arXiv:1606.04435* (2016).

[17] Jamie Hayes and George Danezis. 2018. Learning Universal Adversarial Perturbations with Generative Models. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 43–49.

[18] Victoria Hodge and Jim Austin. 2004. A survey of outlier detection methodologies. *Artificial intelligence review* 22, 2 (2004), 85–126.

[19] Weiwei Hu and Ying Tan. 2017. Generating adversarial malware examples for black-box attacks based on GAN. *arXiv preprint arXiv:1702.05983* (2017).

[20] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. 2011. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*. ACM, 43–58.

[21] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. 2017. Query-Efficient Black-box Adversarial Examples (superceded). *arXiv preprint arXiv:1712.07113* (2017).

[22] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. 2016. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*. ICST (Institute for Computer Sciences, Social-Informatics and …, 21–26.

[23] Eric Jones, Travis Oliphant, and Pearu Peterson. 2014. {SciPy}: Open source scientific tools for {Python}. (2014).

[24] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. 2017. A survey of deep learning-based network anomaly detection. *Cluster Computing* (2017), 1–13.

[25] Bo Li and Yevgeniy Vorobeychik. 2014. Feature cross-substitution in adversarial classification. In *Advances in neural information processing systems*. 2087–2095.

[26] Didong Li and David B Dunson. 2017. Efficient manifold and subspace approximations with spherelets. *arXiv preprint arXiv:1706.08263* (2017).

[27] Zilong Lin, Yong Shi, and Zhi Xue. 2018. Idsgan: Generative adversarial networks for attack generation against intrusion detection. *arXiv preprint arXiv:1809.02077* (2018).

[28] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*. IEEE Computer Society, Washington, DC, USA, 413–422. https://doi.org/10.1109/ICDM.2008.17

[29] Qiang Liu, Pan Li, Wentao Zhao, Wei Cai, Shui Yu, and Victor CM Leung. 2018. A survey on security threats and defensive techniques of machine learning: A data driven view. *IEEE access* 6 (2018), 12103–12117.

[30] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. 2018. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613* (2018).

[31] Erxue Min, Jun Long, Qiang Liu, Jianjing Cui, Zhiping Cai, and Junbo Ma. 2018. Su-ids: A semi-supervised and unsupervised framework for network intrusion detection. In *International Conference on Cloud Computing and Security*. Springer, 322–334.

[32] Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D Joseph, Benjamin IP Rubinstein, Udam Saini, Charles A Sutton, J Doug Tygar, and Kai Xia. 2008. Exploiting Machine Learning to Subvert Your Spam Filter. *LEET* 8 (2008), 1–9.

[33] James Newsome, Brad Karp, and Dawn Song. 2006. Paragraph: Thwarting signature learning by training maliciously. In *International Workshop on Recent Advances in Intrusion Detection*. Springer, 81–105.

[34] Nga Nguyen Thi, VL Cao, and Nhien-An Le-Khac. 2017. One-Class Collective Anomaly Detection Based on LSTM-RNNs. *Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXVI, Lecture Notes in Computer Science* 10720 (2017), 73–85.

[35] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* (2016).

[36] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. 2016. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814* (2016).

[37] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 582–597.

[38] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[39] Li Pengcheng, Jinfeng Yi, and Lijun Zhang. 2018. Query-Efficient Black-Box Attack by Active Learning. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1200–1205.

[40] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[41] Jorge Rivero, Bernardete Ribeiro, Ning Chen, and Fátima Silva Leite. 2017. A grassmannian approach to zero-shot learning for network intrusion detection. In *International Conference on Neural Information Processing*. Springer, 565–575.

[42] Ishai Rosenberg, Asaf Shabtai, Lior Rokach, and Yuval Elovici. 2018. Generic black-box end-to-end attack against state of the art API call based malware classifiers. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 490–510.

[43] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. 2019. The Odds are Odd: A Statistical Test for Detecting Adversarial Examples. *arXiv preprint arXiv:1902.04818* (2019).

[44] Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J Doug Tygar. 2009. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*. ACM, 1–14.

[45] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep One-Class Classification. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, StockholmsmÃ€ssan, Stockholm Sweden, 4393–4402. http://proceedings.mlr.press/v80/ruff18a.html

[46] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. 2017. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*. Springer, 146–157.

[47] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. 2000. Support vector method for novelty detection. In *Advances in neural information processing systems*. 582–588.

[48] Mohit Sewak, Sanjay K Sahay, and Hemant Rathore. 2018. An investigation of a deep learning based malware detection system. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*. ACM, 26.

[49] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. 2018. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104* (2018).

[50] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. 2018. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization.. In *ICISSP*. 108–116.

[51] Soroush M Sohi, Fatemeh Ganji, and Jean-Pierre Seifert. 2018. Recurrent Neural Networks for Enhancement of Signature-based Network Intrusion Detection Systems. *arXiv preprint arXiv:1807.03212* (2018).

[52] Fnu Suya, Yuan Tian, David Evans, and Paolo Papotti. 2017. Query-limited black-box attacks to classifiers. *arXiv preprint arXiv:1712.08713* (2017).

[53] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[54] Thomas Tanay and Lewis Griffin. 2016. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690* (2016).

[55] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453* (2017).

[56] Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. 2018. Adversarially Learned Anomaly Detection. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 727–736.

[57] Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Bowei Xi. 2012. Adversarial support vector machine learning. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1059–1067.

[58] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. 1997. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)* 23, 4 (1997), 550–560.

[59] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. (2018).