Improved Phishing Detection Algorithms using Adversarial Autoencoder Synthesized Data

Hossein Shirazi*, Shashika R. Muramudalige†, Indrakshi Ray* and Anura P. Jayasumana†
*Department of Computer Science
Colorado State University
Fort Collins, CO 80523
Email:{shirazi, iray}@colostate.edu
† Electrical & Computer Engineering
Colorado State University
Fort Collins, CO 80523
Email: {shashika.muramudalige, anura.jayasumana}@colostate.edu;

***Abstract*—Malicious actors often use phishing attacks to compromise legitimate users' credentials. Machine learning is a promising approach for phishing detection. While the accuracy of machine learning algorithms is often dependent on the training data, very little attack data for training is available. We propose an approach for augmenting existing datasets that can be used by machine learning algorithms. We use an Adversarial Autoencoder (AAE) to generate samples that mimic the phishing websites and provide metrics to assess the quality of the generated samples. We test these samples against models trained with real-world data. Some of generated samples are able to evade existing detection model. We then use a portion of these samples in training. The new machine learning models are more robust and have higher accuracy. In other words, real-world phishing site data augmented with AAE synthesized data used for training the model is more effective for phishing detection.**

I. Introduction

Phishing uses social engineering and technology to steal user's identity [1] and sensitive information [2]. Commercial and government sectors have seen a proliferation of these attacks in recent years. Federal Bureau of Investigation (FBI) estimates $12.5 billion in financial losses world wide from 78,617 reported incidents between October 2013 to May 2018 [3], [4].

Khonji *et al.* [5] provide the most comprehensive definition that covers all types of phishing attacks. Attackers mimic legitimate electronic communications and websites and lure legitimate users in revealing their credentials. We focus on detecting phishing websites. Supervised machine learning [6] appears to be a promising technique for phishing detection [7]–[10]. However, this technique requires a priori labeled data in the training phase. This trained model is then used to classify real-world data [11] into genuine and phishing websites.

Supervised machine learning algorithms need a sufficiently large labeled dataset for training the model. However, obtaining such a dataset is not easy. Das *et al.* [12] studied more than 300 papers published during 2010-2018 and recognized four issues of *availability*, *diversity*, *recency*, and *quality* related to existing phishing datasets in the literature. In this study, we focus on the problem of low volume phishing datasets obtained from the real-world. Collecting phishing data and labeling them is hard and labor-intensive [13]. Gathering data in an adversarial context such as phishing poses unique challenges. In addition, researchers are often reluctant to share their datasets related to cyber-security problems due to concerns such as privacy and liability. Only 10% of researchers shared their dataset publicly for a similar security networking problem [14].

There are repositories that collect the link to phishing websites like `PhishTank.com` or `OpenPhish.com`. However, such websites only provide a list of links. Obtaining the links, extracting the features, and converting them into a labeled dataset are complex tasks requiring expertise.

Adversaries actively try to bypass the detection algorithm or corrupt it. For example, an attacker can obfuscate code to invalidate the feature extraction process. Moreover, the low volume of existing phishing datasets [12] may cause the learning classifier to not converge and the performance to be inconsistent. In short, the training model may be imperfect in the absence of adequate data. We focus on how to increase the size of datasets while preserving the characteristics of existing data but without doing actual data collection. Such an approach is essential when data is unavailable or when the data collection process is laborious and infeasible.

A. Proposed Approach

We model an attacker, define his goals, and discuss his capabilities. We develop an adversarial autoencoder (AAE) network to mimic websites that are in-tune with the capabilities and characteristics of actual attackers. We check the similarity between synthesized samples and phishing samples at the *feature* and *instance* level to make sure the synthesized samples following same characteristics as real samples. This is needed to prove the validity of our synthesized samples.

We use four publicly available datasets developed by other researchers. We begin by training the learning models for each of those datasets. Our results are close to the results that have

been reported by the authors of the datasets. This validates the ability of our learning model to accurately detect the relation between the features and to discriminate phishing samples from the legitimate ones. In a separate experiment, we show that inadequate number of samples can negatively affect the learning scores. We then test if synthesized samples can circumvent the trained model, and show that a significant number of new synthesized samples can bypass existing models. This shows that the learning models are prone to exploratory attacks [15].

Subsequently, we demonstrate that by injecting a small portion of newly synthesized samples into the training phase results in more robust learning algorithms that are more resistant to exploratory attacks while also providing better accuracy in that it can increase the classifier's performance with regards to detecting phishing as well as legitimate websites.

We show our proposed sample generation approach is useful on multiple counts. First, it obviates the need for data collection. Data may be unavailable or the data collection process may be infeasible for real-world attacks. Second, adding synthesized samples leads to a more precise classifier for phishing samples. Third, injecting these synthesized samples to training data generates more robust classifiers which are resistant to data poisoning attacks.

B. Key Contributions Approach

Our key contributions are as follows:

- As we demonstrate that the existing datasets lack a sufficient volume of samples for training phishing detection algorithms, we present an AAE (Adversarial Autoencoder) model to synthesize phishing data that mimic real phishing samples to enhance the training dataset.
- The proposed data generation method further widens opportunities in other research contexts that suffers from the lack of data for training models or rigorous analysis.
- We show that the actual samples and synthesized samples via AAE are similar enough in terms of two levels, i.e., feature and instance level. At the feature level, we use marginal distribution combined by calculating Euclidean distance. At the instance level, we clustered phishing samples and tested synthesized samples with that model.
- We demonstrate that supplementing such datasets with adversarial synthetic data can significantly enhance the effectiveness of detection algorithms. We show injecting the synthesized samples into the training set will increase the detection rate of phishing algorithms. In addition, we show the vulnerability of the existing model in that they cannot label many of these new synthesized samples correctly.

The rest of the paper is organized as follows. In Section II, we describe the various existing phishing detection algorithms that use machine learning. We also provide some background on generative networks. In Section III, we model the attacker and describe the construction of synthesized samples using generative networks. In Section IV, we create our experimental configuration and explain the results of our experiments. In Section V, we conclude the paper and discuss some future work.

II. Related work

Machine learning algorithms are well suited for phishing detection as they can assimilate common attack patterns such as hidden fields, keywords, and page layouts across multiple phishing data instances and create learning models that can detect whether a given website is genuine or phishing. In the prior machine learning approaches [7]–[10], [16], [17], researchers engineered novel sets of features from diverse perspectives using public datasets or their own generated datasets. The models were trained on phishing and legitimate datasets. These models were then used to predict whether unknown datasets are genuine or phishing.

Niakanlahiji *et al.* [7] introduced PhishMon, a scalable feature-rich framework with a series of new and existing features derived from HTTP responses, SSL certificates, HTML documents, and JavaScript files. It does not rely on third-party services to extract features and is language agnostic and detects phishing instances in real-time. The authors reported accuracy of 95% on their datasets.

Zhou *et al.* [18] extracted 154 features based on the content of a webpage merging with four time-based, two search-based, and 11 heuristic features to create a labeled dataset. They created a balanced dataset with 8180 instances. Zhou *et al.*compared Random Tree as the best performing classifier among other classifiers and achieved precision of 99.4% and 0.1% false positive rate.

Mao *et al.* [9] studied visual similarity of phishing and legitimate websites by comparing Cascading Style Sheets (CSS) using an automated process. They proposed a learning-based aggregation analysis mechanism that can distinguish phishing websites from legitimate ones.

Detecting phishing instances by analyzing the URL of phishing websites have been widely studied in the literature. Sahinguz *et al.* [8] proposed a set of natural language processing based features on URL of the websites and ran seven different classification algorithms to detect phishing websites. This study is language independent and can detect phishing websites in real-time without needing third-party services. They achieve a 97.98% accuracy rate for detecting phishing URLs.

Recently, Hong *et al.* [16] focused on solving the phishing problem by only considering the URL of the website and collecting a handful of lexical features by surveying the literature and combining them with the blacklisted domain. The results show the F-1 scores of 0.84.

Jain *et al.* [10] extracted 19 different features from the URL and source code of websites to distinguish phishing websites from legitimate ones. The features are extracted from the client-side and do not rely on third-party services and achieved a 99.39% true positive rate and the overall accuracy was 99.09%.

Patil *et al.* [17] surveyed three approaches for phishing detection, including analyzing URL features, checking to host

the website, and visual appearance-based analysis for checking the genuineness of the website.

While machine learning approaches have demonstrated excellent results for detecting phishing websites, the evaluated dataset's quality is an important factor.

Shirazi *et al.* [19] observed datasets used in the literature are inadvertently biased with respect to the features based on the website URL or content. Moreover, some of the features may become obsolete with time or as new attacks emerge. Sometimes the authors extracted features for the first page of legitimate websites, not the other pages. A machine learning algorithm will be useful if trained on enough data samples, but there is not a simple way to estimate the needed dataset size. The right size is related to the complexity of the problem and the complexity of the learning algorithm. This could be seen as a type of sample size determination (SSD) that evaluates the needed sample size in a specific problem.

For example, Figueroa *et al.* described a sample size prediction algorithm that conducted weighted fitting of learning curves in an active learning algorithm [20]. Active learning systems attempt to minimize the number of required labeled data and maximize the performance of the model by asking queries in the form of unlabeled instances to be marked by another agent such as the domain expert [21].

Small datasets often times create inaccurate learning models. Thus, the right size data set is critically important. The data gathering and labeling are challenging and often times expensive operations. Getting enough data may not be possible in many cases, especially in the cyber-security context. To address this challenge, many data augmentation algorithms have been proposed and used in the literature [22]–[25].

Shirazi *et al.* [22] used an adversarial algorithm to generate new synthesized samples. It increased the size of the dataset and demonstrated how these samples can evade the classifier. Our approach improves this study in two ways. Shirazi *et al.* used a heuristic algorithm to generate samples by feature manipulation. Our current AAE network is capable of generating more sophisticated samples with a well-studied algorithm which ensures that the sample matches real-world phishing data. Shirazi *et al.* do not solve the problem of exploratory attacks. In our current work, we demonstrate how to train the model to make it resilient to exploratory attacks.

Other domains also need a good volume of high quality data. Scenarios involving social analytics [26]–[28], privacy [29], and health informatics are some areas that face the issue of limited data availability and data incompleteness. Data collection and maintenance are challenging because of data privacy and confidentiality issues. Behavioral and social network data are inherently sparse and incomplete because sometimes the behavioral indicators are not shown or recorded [30]. Muramudalige *et al.* [31] proposed an adversarial data generation technique using sparse, incomplete, and small training samples. The method was validated via a domestic radicalization dataset which was a small and incomplete dataset.

Goodfellow *et al.* [32] proposed Generative Adversarial Networks (GAN) use for data generation without requiring extensive problem-specific theoretical foundation or empirical verification [33]. The original GAN architecture [32] is capable of capturing the exact distribution of continuous and complete data but cannot be used for learning the distribution of discrete variables [34]. The critical need to capture data distribution with discrete features in diverse application domains such as phishing, medical, crime data, etc. was fulfilled by Goodfellow. Makhzani *et al.* [35] proposed the Adversarial Autoencoder (AAE), which is a probabilistic autoencoder that uses the GAN framework as a variational inference algorithm for both discrete and continuous latent variables. Choi *et al.* [34] also focused on learning the distribution of discrete features, such as diagnosis or medication codes, using a combination of an autoencoder and the adversarial framework.

III. Our Approach

We now explain how we used an AAE to generate synthesized samples of phishing instances. We begin by modeling an attacker and then discuss how to get data that mimics the one that may be produced by him taking into account his capabilities.

A. Threat Model

Shirazi *et al.* [22] modeled the attacker based on attackers' goal, knowledge, and influence in the context of phishing detection by a machine learning algorithm.

Attacker's Goal In the context of the phishing problem, we assume an attacker will attack the integrity of the system by forcing the system to label a phishing instance as legitimate.

Attacker's Knowledge We assume an attacker only knows about features of the phishing instances but not the learning model parameters. This could be considered a realistic assumption as an attacker may have access to the definition of existing datasets but not the specific implementation of a classifier. The adversary that has been modeled in [22] does not have any information about other system parameters like the algorithms that have been used, dataset instances, or learning parameters. The purpose of [22] was to show the vulnerabilities of existing learning models against adversarial sampling attacks and a feature manipulation approach was used. However, in our current approach, we focus on synthesizing new phishing samples to address existing limitations of dataset generation.

Attacker's Influence Ling *et al.* [36] defined two types of attacks: (a) *Causative Attacks* and (b) *Exploratory Attacks*.

In *Causative Attacks*, the attacker influences the training data and can poison the training set with mislabeled samples to affect the training phase. The attacker can poison a portion of data or whole training set depending on what he can access.

In *Exploratory Attacks*, the attacker targets the integrity of the system in which the attempts are toward circumventing the learning mechanism to exploit blind spots in the learning model. In this attack, the attacker crafts intrusions so to evade the classifier without direct influence. Our goal is to design a system that is resilient against exploratory attacks.

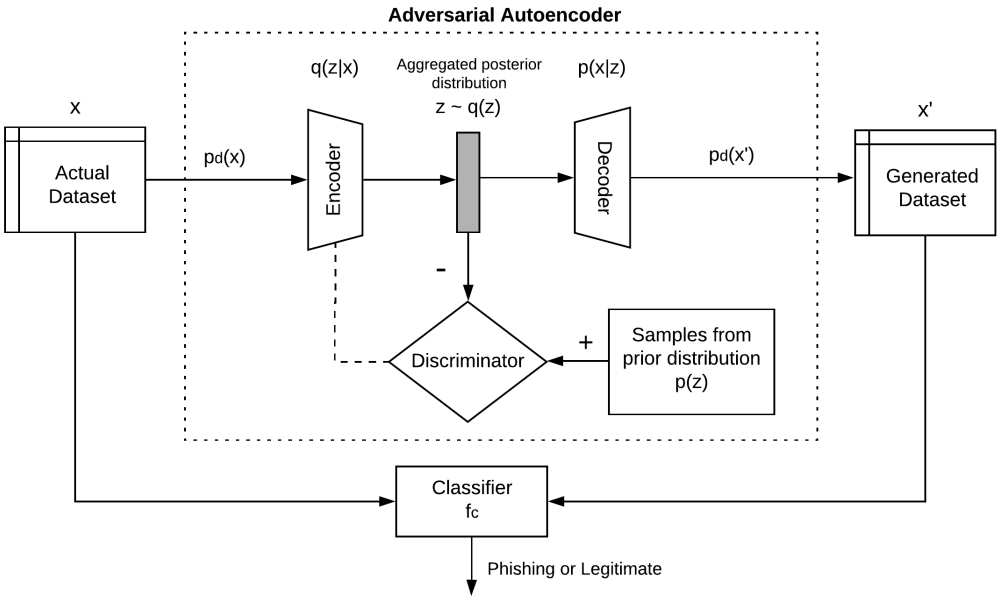


Fig. 1: The architecture of our proposed approach. It consists of an adversarial autoencoder that generates synthesized data using phishing data. The top row depicts the ordinary autoencoder that reconstructs the data from the latent code z. The next row depicts the discriminative network that predicts whether the samples emerge from the hidden code of the autoencoder $q(z)$ or the user-defined prior distribution $p(z)$ [35]. $p_d(x)$ denotes the data distribution. $q(z|x)$ and $p(x|z)$ denote the encoding and decoding distributions respectively. After the data generation, a machine learning classifier (f_c) described in Subsection III-C is used to determine whether the synthesized samples belong to legitimate or phishing sites.

B. Adversarial Autoencoder (AAE) for Synthesized Data Generation

We use the adversarial autoencoder for synthesizing samples that mimic the phishing websites. The adversarial autoencoder is capable of generating both continuous and discrete data distributions. Therefore, AAE is a perfect fit for generating discrete feature sets in phishing samples. The architecture of the adversarial autoencoder is shown in Figure 1. The autoencoder derives a compressed knowledge representation of the original input, which reconstructs the same data distribution.

$$q(z) = \int_x q(z|x)p_d(x)dx \tag{1}$$

An aggregated posterior distribution of $q(z)$ on the latent code is defined with the encoding function $q(z|x)$ and the data distribution $p_d(x)$ as shown in Eq. 1 where x denotes real phishing dataset.

The AAE's operating principle is that the autoencoder seeks to minimize the reconstruction error while the adversarial network attempts to minimize the adversarial cost. *Reconstruction phase* and *regularization phase* are two simultaneous phases that arise during training. In the reconstruction phase, the autoencoder's data reconstruction error is minimized, often referred to as the loss. The regularization phase relates to the adversarial component of the network. It minimizes the adversarial cost to fool the discriminator by maximally regularizing an aggregated posterior distribution $q(z)$ to the prior $p(z)$ distribution.

The simultaneous training process allows the discriminative adversarial network into thinking that the samples from hidden code $q(z)$ come from the prior distribution $p(z)$ [35]. In this work, a normal distribution is exploited as the arbitrary previous $p(z)$. After the training process, the adversarial network synthesizes samples similar to the phishing samples through the prior distribution $p(z)$.

We train generative models for each dataset because each has different sets of distinct features. The feature values are varied in many value ranges. Thus, the values are normalized between -1 and 1 before feeding to the encoder and are denormalized after data generation from the decoder.

At the end of this step, we will have two datasets: *Original Dataset*, which has been used to generate adversarial samples and a new *Synthesized Dataset* that consists of new synthesized phishing samples that mimic phishing websites.

We fed the model with only phishing samples so all of the synthesized samples are phishing. The synthesized dataset has the characteristics of phishing datasets generated by real-world attackers. We combine these two datasets to feed them into a classification algorithm that can distinguish phishing samples from the legitimate ones. This classifier is unaware of whether the samples are synthesized, which means generated by adversarial, or real, which means we got them from an existing dataset. The instances are labeled as legitimate or phishing and classifier will predict them accordingly.

TABLE I: Definition of performance metrics

Score	Formula	Description
TPR	$\frac{N_P \to P}{N_P}$	*correctly* classified phishing
PPV	$\frac{N_P \to P}{N_P \to P + N_L \to P}$	*correctly* over *total* predicted phishing
f1	$2 * \frac{TPR * PPV}{TPR + PPV}$	harmonic average of TPR and PPV
ACC	$\frac{N_L \to L + N_P \to P}{N_L + N_P}$	classified *correctly* in the dataset

The samples that have been generated by that adversarial network will be injected into our training set with correct labels.

The use of synthesized samples solves two purposes at the same time. First, we increase the dataset size and alleviate the problem of data unavailability and data collection. Second, it helps to make the existing learning algorithm resilient against adversarial attacks. We evaluate our hypothesis in Section IV in an experimental study.

C. Machine Learning Classifier

For training purposes, we use six different classifiers available in the Scikit-learn tool [37]. The classifiers that we use are *Decision Tree* (DT), *Gradient Boosting* (GB), *k-Nearest Neighbors* (KNN), *Random Forest* (RF), and *Support Vector Machine* with two kernels: *Linear* (SVM(L)) and *Gaussian* (SVM(G)) kernel. We will use scores defined in Table I to compare them.

IV. Experiments and Evaluation

A. Used Datasets

We use four publicly available phishing datasets on the Internet, and the details of these datasets are given below.

Dataset 1**: DS-1**: Shirazi *et al.* [19] published their unbiased phishing dataset in 2018. Each instance in this dataset has eight features, and all are related to the domain name of the websites.

Dataset 2**: DS-2**: Rami *et al.* [38] created this dataset in 2012 and shared it with UCI machine learning repository [39]. This set includes 30 features that are divided into five categories: *URL based*, *abnormal based*, *HTML-based*, *JavaScript based*, and *domain-name based* features.

Dataset 3**: DS-3**: In 2014, Abdelhamid *et al.* [40] shared their dataset on UCI machine learning repository [39]. The features include HTML content-based features and some features that require third-party services inquiries, such as DNS servers that perform domain-name age lookup.

Dataset 4**: DS-4**: This dataset is the most recent, from the year 2018, that is publicly available and has been created by Tan *et al.* [41] and was published on Mendeley [1] dataset library. This dataset includes 48 features, a combination of URL-based and HTML-based features.

Table II summarizes the number of instances, features, and the portion of legitimate vs. phishing instances in each dataset.

[1]https://data.mendeley.com/

TABLE II: Number of instances, features, and portion of legitimate and phishing websites in each dataset

Dataset	Data shape (#)		Instances (%)	
	Size	Features	Legitimate	Phishing
DS-1	2210	7	44.71	55.29
DS-2	11055	30	55.69	44.31
DS-3	1250	9	43.84	56.16
DS-4	10000	48	50.0	50.0

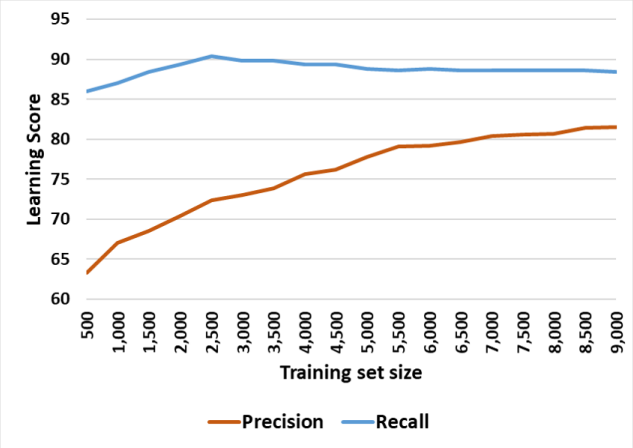


Fig. 2: Trend of precision and recall vs. different training set sizes in DS-4.

B. Performance of the Model

In the first experiment, we checked the performance of the model without considering any synthesized samples. The necessity of this step is to show that our learning model can distinguish between legitimate and phishing instances. We used six different machine learning classifiers listed in Section III. We used 80% of data for training purposes and 20% for testing in five-fold cross-validation. We ran all experiments ten times and reported mean and standard deviation for each dataset. The results are reported in Table III.

GB gives best F-1 scores for three datasets of DS-1, DS-3, and DS-4 with 0.970, 0.933, and 0.974 respectively. RF generates the best F-1 score for the DS-2 dataset with a score of 0.962. The comparison of all six classifiers together, GB with a score of 0.9555 followed by RF which is slightly less than it with the score of 0.9547.

Also, GB gives the best accuracy for dataset of DS-1, DS-3, and DS-4 with the accuracy of 0.967, 0.925, and 0.974 and RF gives the best accuracy for dataset of DS-2 with 0.967. On average, both RF and GB give the best accuracy as well for all datasets.

These results show that we were able to replicate datasets' owner experiments and get results statistically close to their study and prove our learning model is working perfectly.

For the rest of the experiments, we selected SVM with a linear kernel to show how the number of samples can affect different learning scores and then how we were able to increase the performance of this learning model with our proposed approach.

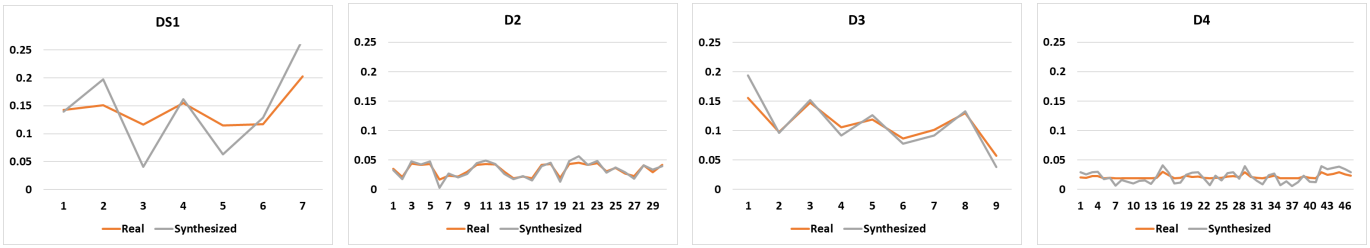


Fig. 3: Marginal Distribution of X (Column sum)

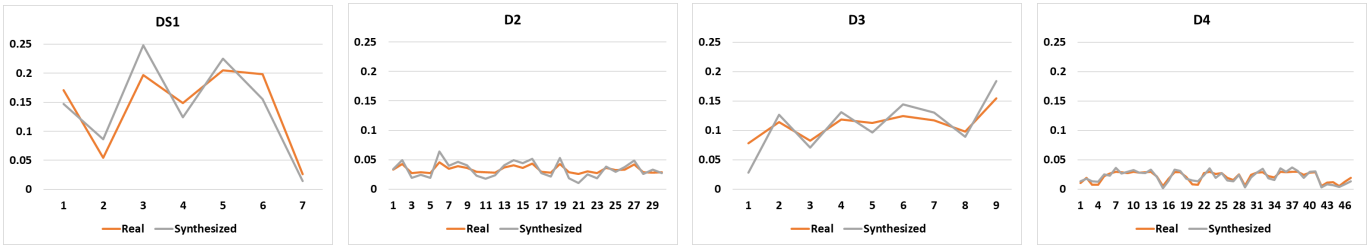


Fig. 4: Marginal Distribution of Y (Row sum)

TABLE III: The F-1 and Accuracy scores for four datasets and all six usec classifiser.

	F1					ACC				
Cls.	**DS-1**	**DS-2**	**DS-3**	**DS-4**	**Avg (F1)**	**DS-1**	**DS-2**	**DS-3**	**DS-4**	**Avg (ACC)**
DT	0.9615	0.9571	0.9027	0.9610	**0.9456**	0.9585	0.9625	0.8930	0.9610	**0.9438**
GB	**0.9705**	0.9446	**0.9333**	**0.9745**	**0.9557**	**0.9674**	0.9514	**0.9256**	**0.9745**	**0.9547**
KNN	0.9641	0.9336	0.9146	0.9028	**0.9288**	0.9602	0.9414	0.9048	0.8999	**0.9266**
RF	0.9668	**0.9624**	0.9163	0.9731	**0.9547**	0.9634	**0.9670**	0.9071	0.9733	**0.9527**
SVM(L)	0.9530	0.9165	0.9150	0.8502	**0.9087**	0.9489	0.9274	0.9048	0.8447	**0.9065**
SVM(G)	0.9470	0.9079	0.9106	0.7110	**0.8691**	0.9421	0.9201	0.8992	0.6449	**0.8516**
Max	**0.9705**	**0.9624**	**0.9333**	**0.9745**		**0.9674**	**0.9670**	**0.9256**	**0.9745**	

C. Performance of the Model Over Number of Training Set

In this experiment, we studied the effects of the learning model when it was trained with different dataset sizes. For this purpose, we selected DS-4 as one of the largest phishing datasets that have been used in this study with 10000 samples. We reserved 1000 instances with an equal number of phishing and legitimate samples and then trained the model multiple times. In each time, we increase the size of the dataset by 500 until we reached 9000 samples. Figure 2 depicts these results.

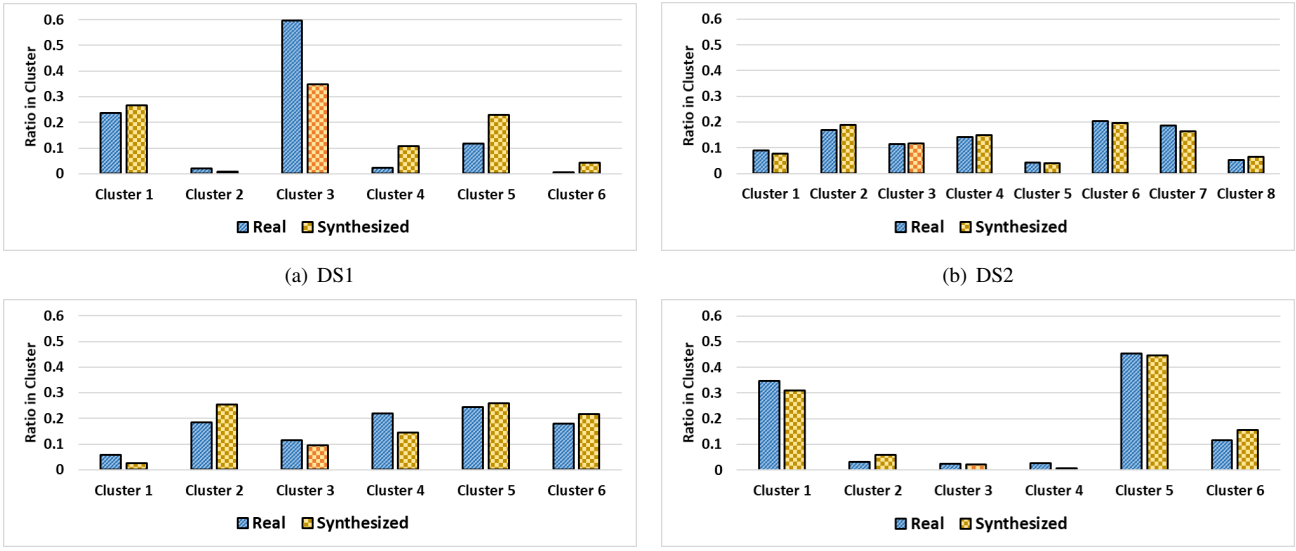
This graph shows when the size of the training set increases (from 500 samples to 9500 samples) the *precision* also increases while *recall* has over 85% of the learning score in each case. In our experiment, precision, which is the ability to label phishing samples correctly, grows gradually when the training size increases. When the number of training sets is equal to 500, the precision is less than 0.65, but it increases to 0.85 when there are 9000 samples in the training set.

The same pattern is seen for the *recall*, while that trend is increased with less gradient. This demonstrates that the size of the training dataset is important to get good precision and recall.

D. Comparison Between Real and Synthesized Phishing Samples

We need to demonstrate that the generated synthesized samples are similar to real phishing ones. This similarity could be checked at two levels: *feature* level and *instance* level. At the feature level, we need to ensure that the values assigned to the features in the synthesized samples are similar to real instances. We have done this through marginal distributions.

To calculate marginals, the transition probabilities were determined for normalized feature values (described in Subsection III-B) to maintain the consistency across different ranges of values in features. The calculated marginal distributions are shown in Figure 3 and Figure 4 that further ensures the capability of the data generation technique. We also calculated the *Euclidean distance* between the marginal probabilities of the real and synthesized phishing data. The values are depicted in Table IV. The *Euclidean distances* are less than 0.13 across all the datasets which is very low. The minimum *Euclidean distance* of the column sum is 0.027 for the DS-4 dataset and the row sum is 0.025 for the DS-2 dataset. DS1 has the least number of features among datasets. We believe that is why the autoencoder slightly less performs on capturing the underlying multi-dimensional structure and transforming to the compressed latent code in DS-1.



(a) DS1

(b) DS2

(c) DS3

(d) DS4

Fig. 5: Ratio of real and synthesized phishing samples in each cluster in four datasets.

TABLE IV: *Euclidean distances* between real and synthesized phishing data.

Dataset	Euclidean distance	
	X (Column sum)	Y (Row sum)
DS-1	0.085	0.123
DS-2	0.042	**0.025**
DS-3	0.069	0.048
DS-4	**0.027**	0.050

After we checked the similarity of synthesized samples and real phishing samples at the feature level, we need to check the similarity at the instance level to make sure the final results of AAE are also similar. For this purpose, we created a clustering model based on real phishing samples and then tested on synthesized ones. We calculated what ratio of samples belongs to each cluster.

We used KNN for clustering real phishing samples and used *elbow* method to get the optimum numbers of clusters for each dataset. The optimum number of clusters for DS-1, DS-3, and DS-4 is 6 clusters, and DS-2 is 8 clusters. Figure 5 shows the results of this experiment.

As Figures 5(a), 5(b), 5(c), and 5(d) show, the ratio of synthesized samples is similar to ratio of real phishing samples and there is not any significant difference among them. This proves our adversarial generator can synthesized samples in the same distribution as the real ones.

Computational Complexity With the recent advancement of machine learning libraries and the use of the GPUs have significantly reduced the training time for deep neural networks. We can train an adversarial autoencoder network within a few minutes on a machine with a 3.3GHz CPU and 64GB RAM speed (without GPU support). For better accuracy, we need a sufficient amount of training data that is entirely representative of the underlying data distribution. However, there are situations like phishing websites detection and behavioral health, social network analysis where sufficient data collection is problematic due to various restrictions. Our proposed technique improves the data while producing a representative and sufficiently large dataset in the particular data domain.

E. Poisoning Improvements

In this experiment, we analyzed two hypotheses: (i) if the synthesized samples can bypass the model at a rate more than real samples and (ii) if adding new synthesized samples into the training set could immune the learning algorithm against such adversarial attacks.

Toward this, we first trained models with only the real samples without knowing any synthesized instances and called this our *Initial Model*. We tested this model with two sets of real and synthesized samples and calculated the detection ratio for both cases. The real detection ratio shows the model's capability to detect phishing samples and could be used as a ground-truth for comparing. The synthesized detection ratio demonstrates how many of the synthesized samples were detected and how many were bypassed.

Figure 6 shows the detection rate in four different testing sets when an SVM with a linear kernel has been trained. We compared real samples and synthesized samples for the first goal, where only real samples were used for training (Initial model). Results reveal that two models, DS-1 and DS-4, are vulnerable against adversarial samples. Detection rate for synthesized samples shows a decrease of 9% and 28% for DS-1 and DS-4, respectively; A significant decline for these two models. However, adversarial samples did not affect the two other models.

As we explained in Section III-B, injecting synthesized samples may make the learning model resilient against exploratory

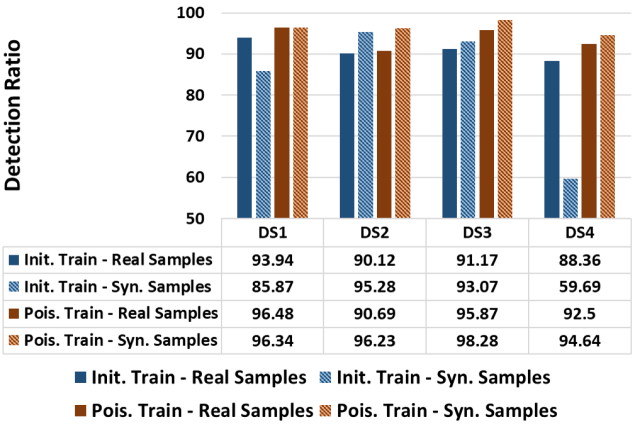


Fig. 6: Detection rate of real and synthesized samples with initial and poisoned model for four datasets.

attacks. To evaluate the second hypothesis, we injected 20% of synthesized samples into the training set, retrained the model, and called it *poisoned model*. We then tested this model with two types of samples: real and synthesized. The results, depicted in Figure 6, can be discussed in three folds.

First, injecting synthesized phishing samples into the training set could make the classifier less vulnerable to such attacks. For all four datasets, the detection rates for synthesized samples with the injected model are better than those tested with the initial model (a model without having any synthesized samples in the training set.) Albeit this improvement is not unified for all datasets. For example, in DS-1, the detection rate for synthesized samples improved by more than 10% (from 85% to 96%) when tested with a poisoned model. In the same situation for DS-4, a significant improvement of more than 34% has happened.

Another observation is that none of the studied datasets experienced a decrease in the detection rate for real samples, but some have seen some improvements. The detection rate for real samples has been improved by 2, 4, and 5 percentages for DS-1, DS-3, and DS-4. This proves that injecting adversarial samples at least does not negatively affect the model's ability to detect real samples, but it improves the model in many cases. While this was not an initial goal for this study, it slightly improved the total model's performance.

Finally, in four datasets, detection ratio for synthesized samples are equal or better than real samples when the poisoned model was used. This is a luminous result that shows adversarial attacks against these models have been mitigated, and poisoned models are less vulnerable to adversarial attacks.

Note that, while DS-4 and DS-2 are the two largest datasets, but synthesized samples provided better results for DS-3. It reveals the dataset's size solely is not a deciding factor regarding the quality of the dataset and that the data contents also play a role.

V. Conclusion and Future Work

Supervised machine learning is a promising approach for phishing detection. However, sufficient volumes of data regarding phishing websites are unavailable and often infeasible to obtain. Towards this end, we demonstrated how Adversarial Autoencoders can be used for synthesizing samples that mimic data of real phishing websites. We compared the similarity of the features and instances of the generated data to ensure that the generated data may be realistically generated by the attacker. We used four publicly available datasets for our experiments. Our experiments revealed that the learning algorithms work better when they are trained with larger volumes of data. Injecting synthesized data in the training set improved the accuracy and recall of the learning algorithms. Moreover, the learning algorithms that included some synthesized data also were significantly more robust to exploratory attacks. Our future work involves the use the AAE for other network security related applications. We also plan to explore other data quality issues that will make the learning algorithms more robust against attacks.

Acknowledgement

We thank Sid Sutton who helped us with implementation. This work was supported in part by NSF with award number CNS 2027750, CNS 1650573, CNS 1822118 and funding from CableLabs, Furuno Electric Company, SecureNok, AFRL, American Megatrends, Statnett, Cyber Risk Research, and NIST. This work was also supported by the U.S. Department of Justice, Office of Justice Programs/National Institute of Justice under Award 2017-ZA-CX-0002. Opinions or points of view expressed in this article are those of the authors and do not necessarily reflect the official position of policies of the funding agencies.

References

[1] R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *Conference on Human Factors in Computing Systems*, 2006.

[2] S. Singh, A. K. Sarje, and M. Misra, "Client-side counter phishing application using adaptive neuro-fuzzy inference system," in *International Conference on Computational Intelligence and Communication Networks*, 2012.

[3] G. Ho, A. Cidon, L. Gavish, M. Schweighauser, V. Paxson, S. Savage, G. M. Voelker, and D. Wagner, "Detecting and characterizing lateral phishing at scale," in *USENIX Security Symposium*, 2019.

[4] F. B. of Investigation (FBI), "Business e-mail compromise 12 billion dollar scam." https://www.ic3.gov/media/2018/180712.aspx, (accessed July 2, 2020).

[5] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: a literature survey," *IEEE Communications Surveys & Tutorials*, 2013.

[6] G. Chen and G. Wang, "A supervised learning algorithm for spiking neurons using spike train kernel based on a unit of pair-spike," *IEEE Access*, 2020.

[7] A. Niakanlahiji, B.-T. Chu, and E. Al-Shaer, "Phishmon: A machine learning framework for detecting phishing webpages," in *Intelligence and Security Informatics*, 2018.

[8] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from urls," *Expert Systems with Applications*, 2019.

[9] J. Mao, J. Bian, W. Tian, S. Zhu, T. Wei, A. Li, and Z. Liang, "Phishing page detection via learning classifiers from page layout feature," *EURASIP Journal on Wireless Communications and Networking*, 2019.

[10] A. K. Jain and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," *Telecommunication Systems*, 2018.

[11] J. Kirchner, A. Heberle, and W. Löwe, "Classification vs. regression-machine learning approaches for service recommendation based on measured consumer experiences," in *IEEE World Congress on Services*, 2015.

[12] Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, "Systematization of knowledge (sok): A systematic review of software-based web phishing detection," *IEEE Communications Surveys Tutorials*, 2017.

[13] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: when to warp?," in *Digital Image Computing: Techniques and Applications*, 2016.

[14] S. Abt and H. Baier, "Are we missing labels? a study of the availability of ground-truth in network security research," in *2014 Third International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, pp. 40–55, IEEE, 2014.

[15] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, "Adversarial machine learning," in *ACM workshop on Security and artificial intelligence*, 2011.

[16] J. Hong, T. Kim, J. Liu, N. Park, and S.-W. Kim, "Phishing url detection with lexical features and blacklisted domains," 2020.

[17] V. Patil, P. Thakkar, C. Shah, T. Bhat, and S. Godse, "Detection and prevention of phishing websites using machine learning approach," in *International Conference on Computing Communication Control and Automation*, 2018.

[18] X. Zhou and R. Verma, "Phishing sites detection from a web developer's perspective using machine learning," in *Proceedings of the 53rd Hawaii International Conference on System Sciences*, 2020.

[19] H. Shirazi, B. Bezawada, and I. Ray, ""kn0w thy doma1n name": Unbiased phishing detection using domain name based features," in *Access Control Models and Technologies*, 2018.

[20] R. L. Figueroa, Q. Zeng-Treitler, S. Kandula, and L. H. Ngo, "Predicting sample size required for classification performance," *BMC Medical Informatics and Decision Making*, 2012.

[21] B. Settles, "Active learning literature survey," tech. rep., University of Wisconsin-Madison Department of Computer Sciences, 2009.

[22] H. Shirazi, B. Bezawada, I. Ray, and C. Anderson, "Adversarial sampling attacks against phishing detection," in *IFIP Annual Conference on Data and Applications Security and Privacy*, 2019.

[23] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, "Passgan: A deep learning approach for password guessing," in *Applied Cryptography and Network Security*, 2019.

[24] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on gan," 2017.

[25] V. Mirjalili, S. Raschka, A. Namboodiri, and A. Ross, "Semi-adversarial networks: Convolutional autoencoders for imparting privacy to face images," in *Conference on Biometrics (ICB)*, 2018.

[26] B. W. Hung, A. P. Jayasumana, and V. W. Bandara, "INSiGHT: A system to detect violent extremist radicalization trajectories in dynamic graphs," *Data & Knowledge Engineering*, 2018.

[27] B. W. Hung, A. P. Jayasumana, and V. W. Bandara, "Finding emergent patterns of behaviors in dynamic heterogeneous social networks," *IEEE Transactions on Computational Social Systems*, 2019.

[28] S. R. Muramudalige, B. W. K. Hung, A. P. Jayasumana, and I. Ray, "Investigative graph search using graph databases," in *2019 First International Conference on Graph Computing (GC)*, pp. 60–67, 2019.

[29] B. K. Beaulieu-Jones, Z. S. Wu, C. Williams, R. Lee, S. P. Bhavnani, J. B. Byrd, and C. S. Greene, "Privacy-preserving generative deep neural networks support clinical data sharing," *bioRxiv*, 2018.

[30] J. Klausen, R. Libretti, B. W. K. Hung, and A. P. Jayasumana, "Radicalization trajectories: An evidence-based computational approach to dynamic risk assessment of "homegrown" jihadists," *Studies in Conflict & Terrorism*, 2018.

[31] S. R. Muramudalige, A. P. Jayasumana, and H. Wang, "Adversarial data generation of multi-category marked temporal point processes with sparse, incomplete, and small training samples," 2020. Submitted.

[32] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *International Conference on Neural Information Processing Systems*, 2014.

[33] J. Yan, "Recent advance in temporal point process: from machine learning perspective," *SJTU Technical Report*, 2019.

[34] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," *arXiv preprint arXiv:1703.06490*, 2017.

[35] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[36] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, "Adversarial machine learning," in *ACM Workshop on Security and Artificial Intelligence*, 2011.

[37] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, 2011.

[38] R. M. Mohammad, F. Thabtah, and L. McCluskey, "An assessment of features related to phishing websites using an automated technique," in *Internet Technology And Secured Transactions*, 2012.

[39] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository." http://archive.ics.uci.edu/ml (Accessed 2019-05-12).

[40] N. Abdelhamid, A. Ayesh, and F. Thabtah, "Phishing detection based associative classification data mining," *Expert Systems with Applications*, 2014.

[41] C. L. Tan, "Phishing dataset for machine learning: Feature evaluation," 2018. https://data.mendeley.com/datasets/h3cgnj8hft/1 (Accessed 2019-05-12).