Bypassing Detection of URL-based Phishing Attacks Using Generative Adversarial Deep Neural Networks

Ahmed AlEroud†
Information Systems
Yarmouk University, Irbid
Jordan
ahmed.aleroud@yu.edu.jo

George Karabatis
Information Systems
University of Maryland, Baltimore County (UMBC),
Baltimore, MD 21250, USA
georgek@umbc.edu

ABSTRACT

The URL components of web addresses are frequently used in creating phishing detection techniques. Typically, machine learning techniques are widely used to identify anomalous patterns in URLs as signs of possible phishing. However, adversaries may have enough knowledge and motivation to bypass URL classification algorithms by creating examples that evade classification algorithms. This paper proposes an approach that generates URL-based phishing examples using Generative Adversarial Networks. The created examples can fool Blackbox phishing detectors even when those detectors are created using sophisticated approaches such as those relying on intra-URL similarities. These created instances are used to deceive Blackbox machine learning-based phishing detection models. We tested our approach using actual phishing datasets. The results show that GAN networks are very effective in creating adversarial phishing examples that can fool both simple and sophisticated machine learning phishing detection models.

CCS CONCEPTS

• Security and privacy• Intrusion/anomaly detection and malware mitigation• Social engineering attacks

KEYWORDS

Deep Learning; Phishing; Generative Adversarial Networks; URL classification

ACM Reference format:

Ahmed Aleroud, George Karabatis. 2020. Bypassing Detection of URL-based Phishing Attacks Using Generative Adversarial Deep Neural Networks. In *Proceedings of 2020 ACM International Workshop on Security and Privacy Analytics(IWSPA'20)*, March 18, 2020, New Orleans, LA, USA , ACM, New York, NY, USA pages. https:// doi.org/10.1145/3375708.3380315

1 Introduction

Phishing is one of the most widely used attacks to lure users into releasing valuable and confidential information. In phishing attacks, adversaries try to obtain user information through social engineering by using deception and make a person divulge private information or unwittingly provide unauthorized access to a computer system or network. Phishing is not a new attack, but it is still a source of significant percentage of security incidents [1]. Compared to spam, creating a phishing attack is difficult, since it is a targeted attack, short-lived, often occurring for only a few hours, more dynamic, and moving among servers very quickly. Therefore, adversaries usually need to change their attack strategies quite often.

There are various techniques to create phishing attacks such as Spoofed URLs, Bad Domain Name, Shortened URL, Host Name Obfuscation, and Encoded URL obfuscation [2]. In other techniques such as Man in the Middle Attack (MITM) [3, 4] phishers position themselves between the victim and the legitimate site. Although SSL web traffic is generally not vulnerable to MITM, malware-based attacks can modify a system configuration to install a trusted certificate authority in which a MITM can create its own certificate for any SSL-protected site, decrypt the traffic, extract confidential information, and re-encrypt the traffic to communicate with the other side.

Phishing detection using URL features has been studied in many related works [5-7]. Several of those approaches use Machine Learning models to analyze the features of URLs to identify anomalous patterns as signs of phishing, such as using the IP addresses in the URL, long URLs to hide suspicious parts, having the "@" symbol, adding prefixes or suffixes separated by a "-" in the domain, and the existence of the "HTTPS" token in the domain part of the URL. Table 1 provides some examples on those clues.

Adversaries can still find ways to bypass URL classification algorithms using techniques that evade classifiers of URLs. Yet, machine learning-based phishing detection models are usually embedded in browsers as extensions or into mail filtering systems, therefore, they appear as a black-box to phishers or phishing toolkits. Hence, it is quite difficult to identify which classification settings and algorithms are in use.

Table 1: Anomaly clues in phishing pages

Clue in the URL	Example
Includes redirection	http://3104.nnu4urye.info?http://c43n34.com?35u3b
The path contains a URL of a known organization	http://108.179.216.140/~bankofamerica/
Special characters "-" in the host name	http://yj4yb6hmb3.x-cant-bank-you-here-of-my money.cn/yj4yb6hmb3/Oraliao_show_23Y
Long domain name	http://31837.9hzaseruijintunhfeugandeikisn.com/5/54878
Hostname is Encoded	http://www.%64isc%72%65%74%2done-%6ei%67h% 74.%63o%6d
IP is Encoded	http://0x42.0x1D.0x25.0xC2/
E-mail Address in URL	http://username@hotmail.com.fddcol.com

There are several adversarial models that can be utilized to evade phishing classifiers. Goodfellow et al. in [8] proposed Generative Adversarial Deep Neural Networks (GAN) to automate the generation of realistic data in a semi supervised manner. A GAN network consists of two Neural Nets, referred to as the Generator and the Discriminator. The Generator tries to generate examples that look real to the Discriminator, which in turn is a Neural Network used to distinguish between the generated samples and real samples. The Generator learns how to apply nonlinear transformations using a noise vector and convert real instances into adversarial versions that imitate the ground truth data. The Discriminator has been applied in different settings, mostly to differentiate between the real and fake samples created by the Generator. In other related works, the Discriminator is used to provide a guidance to the Generator so that it can fool Blackbox classification algorithms [9].

The idea of evading phishing detection classifiers is not new. Recent approaches focus on testing machine learning classification models using carefully designed instances to check if such classifiers can still identify those instances as phishing attempts through judging the highly weighted features. For instance, in [10] Pham et al. have successfully evaded a Google's Phishing Pages filter using simple rule-based models. However, "such adversarial models can have limited success when applied to more sophisticated phishing detection techniques created using relationships between the features of URLs" [11].

Recently deep-learning adversarial models such as GAN gained quite some traction in Cyber Security. Contrary to traditional techniques, GANs apply a set of non-linear transformations on an original malicious sample to generate an adversarial example to evade classification models. GAN have shown some promising results in intrusion detection [12] and opinion spam [13]. We believe that proposing new techniques to synthesize phishing attempts is quite significant in creating a defensive mechanism that can prevent zero-day phishing attempts. While there are approaches that apply GAN for generating adversarial phishing examples [14], to the best of our knowledge, this is the first work that synthesizes URL adversarial phishing examples in order to evade sophisticated phishing detection techniques, which rely on semantic relationships between the components of URLs.

This work builds on previous research and produces the following contributions:

1. It generates URL-based sophisticated phishing examples through feature perturbation implemented using GAN. The created instances are used to deceive Blackbox machine learning-based phishing detection models. The results show that GANs are very effective in creating adversarial phishing examples that can fool machine learning phishing detection models. Therefore, our approach can still synthesize phishing URL data and evade the detectors, even when the phishing detector is created by analyzing sophisticated features to identify phishing pages.
2. The GAN models are tested based on existence or absence of malicious features in attack examples. However, data may also contain suspicious or borderline features, which cannot be classified as legitimate or malicious. Existing GAN models do not handle those features when trained on benign and phishing examples.
3. It is experimentally tested, and the results show success in generating adversarial phishing based on URL intra-relations. A significant number of the generated examples evaded different phishing detection algorithms.

2 Related Work

Countermeasures of phishing attacks are classified into behavioral and technical. Behavioral countermeasures address phishing through security awareness campaigns. Technical countermeasures utilize defensive techniques to minimize the risk of phishing attacks. The most common technical countermeasures are machine learning techniques. Most of them try to map inputs (features or variables) to desired outputs (response) using a specific function. In the case of classifying phishing URLs, a model is created to categorize URLs into phishing or legitimate ones by learning certain characteristics of the URL. URL classification approaches use features such as domain name, IP address characteristics, and geographic properties to identify phishing URLs. URL related features have been used as inputs to several classification techniques for phishing detection, such as Support Vector Machines, Naïve Bayes, and k-Nearest Neighbor. Among them, the k-Nearest Neighbor produces the best accuracy in one study [15]. Other classifiers based on textual features examine the content of suspicious material to determine whether it is legitimate or phishing. For instance, the detection of phishing in a website can operate on features extracted from the textual content of the main page, its component files, and DOM structure [16]. Several classification techniques utilize hybrid features extracted from website content and URLs in webpages for phishing detection [17-19]. Some methods within this category focus on creating dynamic, adaptive, or ensemble classifiers. Compared with static classifiers, dynamic ones focus on adapting classification rules. In [20] the authors used an Online Support Vector Machine approach that utilizes game theory and previous knowledge to create a phishing detection classifier. In [21] a similar adaptive topic model based classification has been proposed for detecting phishing e-mails. In [22], the authors have explored ensemble methods for phishing detection that relies on the decisions of more than one classifier. Most of classification approaches have been applied to detecting phishing in websites, some to emails [23, 24], and voice using Gaussian mixture [25]. Several clustering algorithms have been used for phishing detection, such as DBscan, k-means, and Self-organizing-maps. In addition to URL-based features [26] and content features [27], clustering of phishing has also incorporated features extracted from website images [28]. Clustering has been applied in detecting attacks in several communication media such as phishing e-mails [29], spoofed websites [28], and voice-based phishing attempts [30]. Adversarial models have been recently used to evade machine learning classifiers. GAN has been used in intrusion detection [12], malware detection [9], and spam detection [31], and phishing[14].

3 Approach

The GAN structure utilized in this work consists of a Generator Network, a Discriminator Network and a Blackbox Phishing Detector. Figure 1 shows the main steps of the proposed approach.

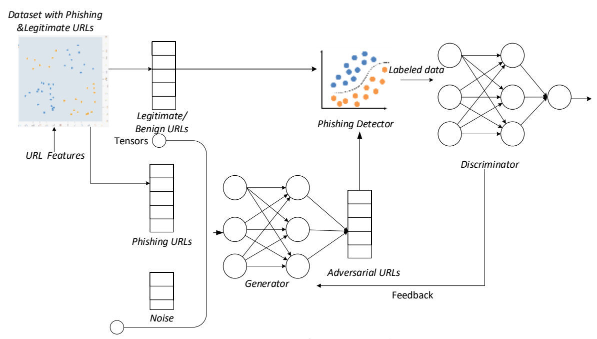

Figure 1: Overview of research steps

The feature vectors of deceptive sites contain certain URL features such as the existence of "HTTPS" token in the domain part of the URL, and a prefix or suffix separated by "-" in the domain. Some data sets are already pre-processed and have their feature vectors created. Our GAN networks work on binarized versions of those vectors. Sometimes there are borderline feature values, where a feature vector has a ternary (i.e., three-valued) feature, such that -1 describes malicious features, 1 describes legitimate features, and 0 describes a suspicious feature. In this case the data is re-encoded into two-bit binary features using the encoding, 0 ⟶ 01, 1 ⟶ 00, and -1 ⟶ 11. This encoding scheme is consistent with the one used in [9], however, in our approach we also consider the case of suspicious features. In our encoding scheme if the original feature vector contains n features in the original encoding, then, $2n$ features are created in the proposed encoding. Each feature in the original data is encoded using two columns now. The data in each of the columns contains one binary feature which can be 0 or 1. This encoding scheme is applied to both phishing and benign examples, as will be detailed in the experiments section.

The Generator Neural Network is used to generate a perturbed version of the phishing examples and convert them into adversarial examples. The Discriminator learns to fit the phishing detector, which is implemented using a specific classification algorithm to identify phishing examples. At each round of the training process, the Discriminator sends feedback to the Generator to modify its weights during the training process to the point where it guarantees that the Generator creates enough examples to evade the phishing detector.

3.1 Generator

Samples of deceptive URL are converted to an adversarial version using the Generator G, which is a feedforward neural network. Phishing URLs consist of a feature vector f with n features and weights G_{w_g}. Both the input vector f and a noise vector s are fed to G. Using our encoding scheme, f consists of m features where $m = 2n$. The features in f take the values of 0 and 1 to identify how malicious the feature is, such that "11" denotes a very malicious feature. The Hyperparameter s is a vector with random entries in the range [0, 1). The structure of the Generator consists of three hidden layers, each with 120 neurons. Hidden layers are activated using ReLU using the formula shown in equation 1 where $g(f) = max(0, f)$.The output layer consists of $2n$ neurons, two for each feature, which are all activated using a sigmoid function as defined in equation (2) in order to return outputs between 0 and 1.

$$g(f) = \begin{cases} f, & f > 0 \\ 0, & f \leq 0 \end{cases} \quad (1)$$

$$g(f) = \frac{1}{1 + e^{-f}} \quad (2)$$

3.2 Discriminator

The Generator parameters are updated based on the feedback from the Discriminator. The resulting adversarial examples are binarized using a threshold to create a binary vector with two inputs 0 and 1. However, for backpropagation to work, non-binarized vectors are used. Vector values are normalized to be always in the range between 0 and 1. The perturbation done using GAN needs to preserve the semantics of the original data. As opposed to previous works, we assume no restrictions regarding bit flips, that is converting features of the original vectors from 1 to 0. In phishing attacks, the lifetime of phishing domains is usually short making it possible to produce new releases of the attack by removing some features and introducing others.

The weights of the Generator are updated using the gradient information from the Discriminator. The latter is a multilayer feedforward network that tries to approximate the decision function of the phishing detector, which the Generator can then use for learning. The Discriminator and phishing detector both take a URL feature vector f as an input with weights w_d. The Discriminator classifies the given URL as a phishing or a legitimate URL using a single output layer with a certain level of an uncertainty denoted by $D_{w_d}(f)$. Adam optimizer is used as an optimization function with ε=1e − 05. The training data for the Discriminator consist of adversarial samples generated by the Generator and the benign samples. The ground truth labels for the Discriminator are the predictions made by the phishing detector, not the actual labels of the samples. Training the Generator and the Discriminator aims at minimizing their loss functions which are measured differently. The predictions of the phishing detector are used as labels for the Discriminators. Therefore, the loss function of the Discriminator tries to minimize classification mismatches between the Discriminator and the phishing detector as shown in the following function:

$$L_D = -\mathbb{E}_{f \in PD_{Leg}} \log\left(1 - D_{w_d}(f)\right) - \mathbb{E}_{f \in PD_{phish}} \log\left(1 - D_{w_d}(f)\right) \quad (3)$$

Where PD_{Leg} is the set of URLs identified as legitimate by the phishing detector, and PD_{phish} is the set of URLs that are identified as phishing. The loss function of the Generator aims to minimize the probability of predicting a phishing URL as phishing by pushing the PD , and ultimately the Discriminator, to recognize it as benign based on the following function

$$L_G = -\mathbb{E}_{m \in o_{phish}, S} \log D_{w_d}\left(G_{w_g}(m,s)\right) \quad (4)$$

Algorithm 1. Generating Adversarial Phishing Examples Using GAN

```
Input:
Original legitimate and phishing examples
$F_{phish}$, $F_{leg}$
The noise $S$ for generating adversarial examples
Output:
The trained Generator $G$ and Discriminator $D$
1. Begin
2. Initialize the $G$, $D$ and $PD$
3. Select a batch $F_{phish}$ of phishing examples
4. Use the noise $S$ for the adversarial generation
   $F'_{phish} | F'_{phish} \leftarrow (F_{phish}, S)$
5. Select the batch of legitimate examples $F_{leg}$
6. Label $F'_{phish}$ and $F_{leg}$ using $PD$
7. Update $D$ weights $D_{w_d}$ along the gradient $\nabla D_{w_d} L_D$
8. Update $G$ weights $G_{w_g}$ along the gradient $\nabla G_{w_g} L_G$
9. End
```

The function is minimized based on the weights of the Generator, G_{w_g}. Minimizing L_G, aims at evading the phishing detector to classify phishing examples as benign activities. This objective is achieved if the Discriminator fits the phishing detector. We use Mini-batch gradient descent to split the training dataset into small batches that are used to calculate model error and update model coefficients. Algorithm 1 shows the major steps in the generating adversarial examples.

The inputs are the original phishing and legitimate example. The outputs are the trained Generator and Discriminator. PD is a Machine Learning Phishing Detector. A batch of phishing examples is selected to generate the adversarial phishing examples (F'_{phish}) using the noise vector S. Both the generated adversarial phishing examples and the legitimate examples are labeled using PD. The weights of the Discriminator and Generator functions are updated based on the loss values.

3.3. Sophisticated phishing attacks

While there are sophisticated techniques to mitigate well-designed phishing attacks, adversaries may employ sophisticated approaches to create phishing URLs. Marchal et al. in [11] identified several characteristics of phishing URLs. They discovered that phishing URLs have no relationships between the low-level registered domain and the remaining part which represents (upper level domain, path, query). Specifically, the features extracted from words that compose a URL show that phishing URLs have low intra-URL relatedness. For example, in the following phishing URL: **http://school497.ru/222/www.paypal.com/29370274276105805/** there are no relationships between the top-level domain(http://school497.ru/222/) and the lower level domain(ww.paypal.com/29370274276105805/)

Intra-URL relatedness can be measured using similarity such the Jaccard Similarity measure [32]. Such measures are used to create machine learning classifiers, which classifies each URL with no intra-relation as a phishing URL. The URLs with good intra-relation scores are classified as legitimate. To discover those relationships, each URL is divided into a registered domain and remaining part. The registered domain consists of the main level domain and the public suffix. The remaining part consists of all other related words extracted from the URL. For instance for the following URL: ***http://sezopoztos.com/paypalitlogin/us/webscr.html?cmd=loginrun***, the keywords of the registered domain and the remaining part of the URL are

$RD_{url} = \{sezopoztos;\ sezopoztos.com\}$
$REM_{url} =$
$\{paypal;\ it;\ login;\ us; web;\ src;\ html;\ cmd; login;\ run\}$

Terms in the above set are categorized into associated and related terms. For a specific term, related and associated terms are used to identify intra URL relations using the sets of keywords REL_{rd}, REL_{rem}, AS_{rd}, AS_{rem}. For instance the term PayPal is a term in the lower level domain of the URL above. By using Google Trends to discover related terms [33], the following sets are related to PayPal.

$AS_{rem} = \{amazon, fees, login\}$
$REL_{rem} = \{amazon, paypal, fees, ebay, uk, login\}$

Many other intra-URL relatedness features can be used to predict the susceptibility of URLs. Phish Storm approach uses the set of features shown in Table 2 [11].

Table 2: Intra-URL relationships used to create sophisticated phishing classifiers

Feature	Description
$Card_{rem}$	Number of words in the remaining part of a URL
$Ratio_{\text{Arem}}$	Association between word in the REM_{url}
$Ratio_{\text{Rrem}}$	ratio of related words for words for words in the REM_{url}
mld_{res}	Search engine results when the main level domain (mld) is searched using Search engines
$Ranking$	Ranking of the mld
J_{RR}	Jaccard similarity between REL_{rd} and REL_{rem}
J_{RA}	Jaccard similarity between REL_{rd} and AS_{rem}
J_{AA}	Jaccard similarity between AS_{rd} and AS_{rem}
J_{AR}	Jaccard similarity between AS_{rd} and REL_{rem}
J_{ARrd}	Jaccard similarity between AS_{rd} and REL_{rd}
J_{ARrem}	Jaccard similarity between AS_{rem} and REL_{rem}

Classifiers such as Phishstorm give high susceptibility score to URLs with low semantic relatedness among its words. However, phishers may create sophisticated attacks trying to bypass classifiers like Phishstorm. For instance, using Phish Tank [34], we found a significant number of phishing URLs with good intra URL relationships. A sample of them is given in Table 3.

Table 3: Intra-URL relationships- Source Phish Tank

http://gillianschultze.com/gillianschlock/error/PPL-03265-965875-9653326-9658745-965874321-96587/Payp/
https://travanleo.com/travanleoadmin_panel/upload/blog/redirect.php
https://paylapal.000webhostapp.com/secure/PayPalLetterByYashvir.html

Even worse, many adversaries consider semantic relationships between URLs. The following URL shows how the adversary creates a phishing URL considering semantic relationships between the words "office" and "Microsoft" at top level domain and the rest of the URL.
https://000office005.tk/microsoftonline.com/microsoftonline.com/cmdlogin=bc955836ebda3b915ae2cc804f4046ba/ih5mw72jik40f9c8brh2gnn7.php?
Adversaries can even leverage a search engine query data to evade systems that rely on such relatedness and ranking measures to discover phishing attacks. Using our encoding scheme, adversarial examples are created by considering modifying those features as detailed in our experiments.

4 Experiments and Evaluation

4.1 Datasets

Two datasets are used in our experiments. The first one contains 4898 phishing websites and 6157 legitimate pages. It contains 30 optimized URL-based features of phishing websites and 30 URL related features. The main features in the dataset include having IP address in the URL, its length, using shortening services, double slash redirection, prefix, suffix, having subdomain, domain registrations length, SSL final state, and port . The dataset is collected mainly from PhishTank and MillerSmiles archives [35]. The Feature Vectors are ternary, where -1 describes the malicious features, 1 describes legitimate features, and 0 describes suspicious features. We re-encode ternary features into two-bit binary features using the encoding, 0 ⟶ 01, 1 ⟶ 00, and -1 ⟶ 11.

Table 4: Encoding the features of dataset 2

Feature	Phishing	Legitimate
$Card_{rem}$	Between 1st Q and ≤ Median→ (01) >Median and ≤ Max → 11	00
$Ratio_{\text{Arem}}$	11	Between 1st Q and ≤ Median→ (01) >Median and ≤ Max → 00
$Ratio_{\text{Rrem}}$	11	Between 1st Q and ≤ Median→ (11) >Median and ≤ Max 00
mld_{res}	0→ 00 1→ 11	0→ 00 1→ 11
$Mld.ps\_res$	0→ 00 1→ 11	0→ 00 1→ 11
$Ranking$	11	00
J_{ARrd}	11	Between 1st Q and ≤ Median→ (01) >Median and ≤ Max → 00
J_{ARrem}	11	Between 1st Q and ≤ Median→ (01) >Median and ≤ Max → 00
J_{RA}	11	Between 1st Q and ≤ Median → (01) >Median and ≤ Max→ (00)
J_{RR}	00	Between 1st Q and ≤ Median → (01) >Median and ≤ Max→ (00)
J_{AA}	00	Between 1st Q and ≤ Median→ (01) >Median and ≤ Max → (00)
J_{AR}	00	Between 1st Q and ≤ Median → (01) >Median and ≤ Max→ (00)

- **1st Q: 1st Q quantile**
- **Median: Median value of the feature**
- **Max: maximum value of the feature**

The second dataset consists of 48,009 phishing URLs that were collected from Phish Tank phishing repository [11]. The phishing URLs represent the ground truth to evaluate our approach. The legitimate dataset consists of 48,009 URLs that were collected from the Open Directory Project (DMOZ) which is a directory of the Web containing more than two million URLs. The dataset consists of 12 features that were encoded according to our encoding scheme as shown in Table 4. This dataset is used to prove that adversarial models can work even when phishing detectors consider inta-URL relations to detect attacks.

4.2 Results

We conducted several experiments using the created GAN as follows:

1. **Experiment 1:** Measuring the Generator-Discriminator loss when varying Epoch values. This experiment is conducted on the first dataset
2. **Experiment 2:** Measuring the detection rate of phishing attacks before and after using GAN on different classifiers. As

part of this experiment, we also measured the false positive rate for the benign examples. This experiment is conducted on the first dataset

3. **Experiment 3:** Measuring the detection rate of phishing attacks when sophisticated intra URL attacking techniques are used to create attacks. This experiment is conducted on the second dataset

In each experiment, the dataset was split into 70% training, and 30% testing parts. Experiments are conducted on core i5 CPU with 2.8 GHZ. We utilized a modified version of the implementation of the GAN that creates Adversarial malware examples [9, 36, 37]. The frameworks are written in PyTorch.

4.2.1 Generator-Discriminator Loss functions when varying the Epoch.

We used several classifiers to measure the Generator-Discriminator loss for the values of $Epoch$ between 1 and 100. This experiment was conducted on the first dataset.

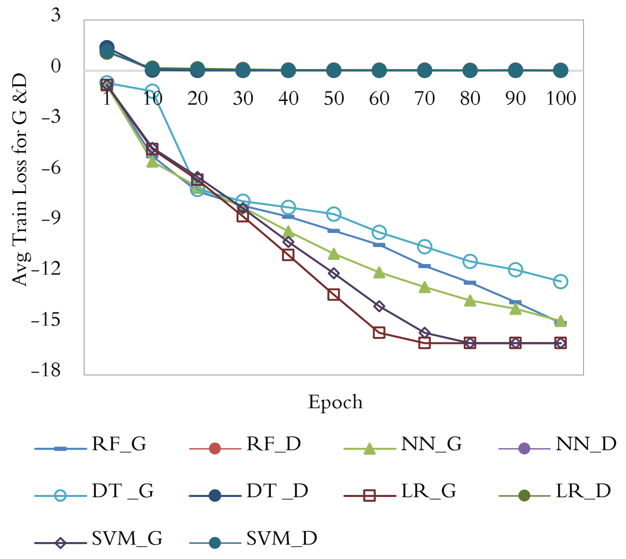

Figure 2: Avg train loss for Generator and Discriminator when varying epoch values, using different pd/ where RF- Random Forest, NN-Nearest Neighbor, DT- Decision Tree, LR-Logistic regression, SVM- Support Vector Machine. Experiment is conducted on dataset1

The results of this experiment are shown in Figure 2. For many classifiers the convergence of the Discriminator occurs at $Epoch = 10$. For Generator the loss function converges for both SVM and LR at $Epoch = 100$. It is also noticed that those classifiers have better learning rate than other classifiers due to lower average training loss for the Generator. Results on the second dataset are similar to the ones shown in Figure 2 (first dataset). In fact, it is possible that the proposed GAN will have a negative loss. As the discriminator is trained, it will map the real example to smaller values and the fake examples to larger values.

4.2.2 PD Results Using Original and Adversarial Data.

In this experiment, we tested all PDs on $Epoch$ = 2 and 100. We compared the detection rate(True Positive Rate) using the original(O) and the adversarial/modified datasets(M) (one column per detector) and reported the results for the first datasets in Figures 3 and 4.

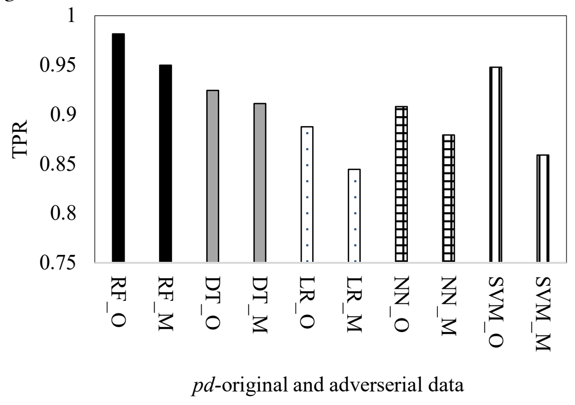

Figure 3. TPR using Original (O) vs. Adversarial modified (M) data/different pds ($Epoch = 2$)/Dataset 1- RF_O: RandomForest_Orignal sample, RF_M: Random Forest_Modified

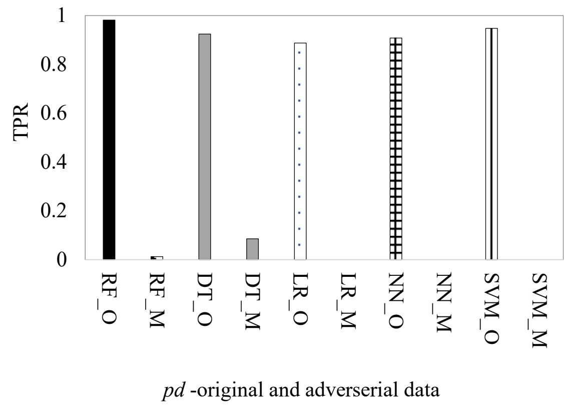

Figure 4: TPR using Original (O) vs. Adversarial modified (M) data with different pd ($Epoch = 100$)

Even when $Epoch = 2$ (as shown in Figure 3), there are adversarial examples that are classified as benign, although the differences between TPR values on the original and adversarial data do not seem very significant.

Increasing the $Epoch$ to 100, leads to a decline in TPR when testing adversarial examples. In fact, when $Epoch$ value is increased to 100, some classifiers were unable to identify any of the adversarial examples leading to zero TPs. To validate this result we noticed an increase in the values of False Positives (FP) as demonstrated in Figure 5 which shows the percentage of benign examples classified as phishing. As the values of $Epoch$ increase the FP rate increases as well, indicating that the detector tends to misclassify benign examples when the Generator converges.

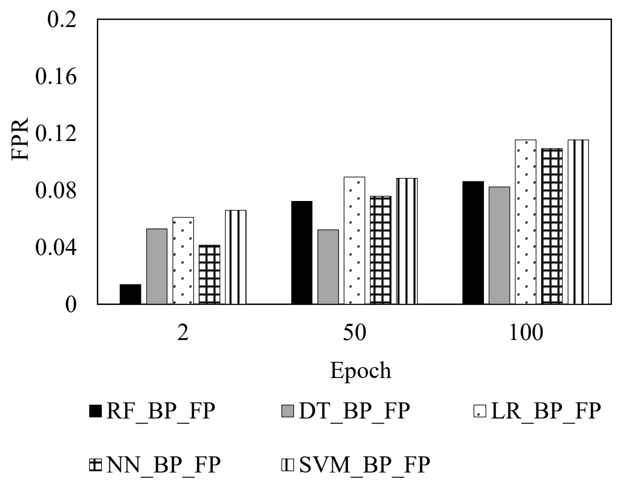

Figure 5: FPR using different pd ($Epoch = 2, 50, 100$)/ (RF_BP_FP: RF: Random Forest Classifier, BP: Benign as phishing, FP: False Positives)

4.2.3 PD Results on Original and Adversarial data: Sophisticated phishing detectors

This experiment examines whether is it possible to evade sophisticated machine learning classifiers that rely on intra-similarity functions between URL features. Adversaries may assume that pd has advanced similarity-based detection techniques to discover complex phishing attacks. This complicates the creation of adversarial examples. This experiment examines if we can still create those examples using GAN on the second dataset.

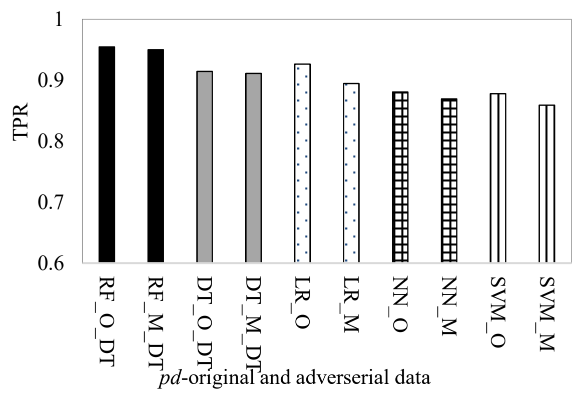

Figure 6: TPR using Original (O) vs. Modified(M) data with different pd ($Epoch = 2$)/dataset2

We tested all pds on $Epoch$ = 2 and 100. We then measured TPR on the original and the adversarial datasets (one column per detector). The results are shown in Figures 6 and 7. This experiment is conducted on the second dataset, which utilizes higher level features and similarity functions to create the phishing detector. There are some adversarial examples that cannot bypass the detector as demonstrated in both figures, even at high values of $Epoch$. However, it is also noticed that a significant percentage of adversarial examples can still bypass the detectors created on high level features. Specifically, GAN can still work well at higher $Epoch = 100$.

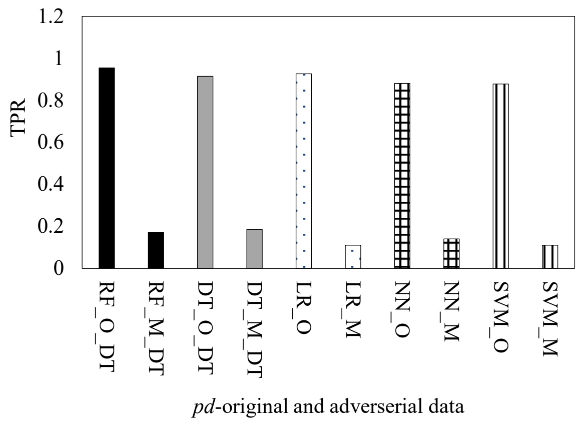

Figure 7: TPR using Original (O) vs. Modified(M) data with different pd ($Epoch = 100$)/dataset2

5 Conclusions and Future Work

This paper presents an approach that evade the existing URL phishing detection approaches via Generative Adversarial networks. The results of the experiments demonstrated that GAN is quite effective even when used to deceive classifiers that are created to defeat sophisticated attacking attempts, such as those utilizing intra-URL relationships. The results also show that adversaries may focus on highly ranked features and can still bypass URL machine learning-based phishing detection techniques. The most important implication of this work is to create countermeasures to mitigate adversarial examples. GAN has not been tested on graph-based phishing detection techniques; therefore, we may still need to test our approach on those phishing detection mechanisms.

ACKNOWLEDGMENTS

The authors would like to thank Réseaux IP Européens (RIPE) for supporting this work. We would also like to thank the programmers of MalwareGAN approach for providing their code and experimental results.

REFERENCES

[1] Howarth, F. *The role of human error in successful security attacks.* (2014). Retrieved on 20/10/2019 from https://securityintelligence.com/the-role-of-human-error-in-successful-security-attacks
[2] Berghel, H., Carpinter, J. and Jo, J.-Y., 2007. Phish phactors: Offensive and defensive strategies. *Advances in Computers*, 70 (2007), 223-268. doi: https://doi.org/10.1016/S0065-2458(06)70005-5
[3] Joshi, Y., Saklikar, S., Das, D. and Saha, S., 2008. PhishGuard: a browser plug-in for protection from phishing, In *Proceedings of the 2nd International Conference on Internet Multimedia Services Architecture and Applications*, Bangalore, India. IEEE. *doi:* https://doi.org/10.1109/IMSAA.2008.4753929

[4] Bicakci, K., Unal, D., Ascioglu, N. and Adalier, O., 2014. Mobile authentication secure against man-in-the-middle attacks, In *Proceedings of the 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud'14)*, NW Washington, DC, USA. IEEE. *doi*: https://doi.org/10.1109/MobileCloud.2014.2

[5] Li, Y., Yang, Z., Chen, X., Yuan, H. and Liu, W., 2019. A stacking model using URL and HTML features for phishing webpage detection. *Future Generation Computer Systems*, 94 (2019), 27-39. doi: https://doi.org/10.1016/j.future.2018.11.004

[6] Jain, A. K. and Gupta, B., 2018. PHISH-SAFE: URL features-based phishing detection system using machine learning, in M. U. Bokhari, N. Agrawal and D. Saini eds. *Cyber Security*, Springer, 467-474. *doi*: https://doi.org/10.1007/978-981-10-8536-9_44

[7] Shirazi, H., Bezawada, B. and Ray, I. *Kn0w thy doma1n name: unbiased phishing detection using domain name based features.* ACM, City, 2018 *doi*: https://doi.org/10.1145/3205977.3205992

[8] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets, In *Proceedings of the Advances in neural information processing systems,* Cambridge, MA, United States. *doi*: https://doi.org/10.1016/j.neunet.2006.08.010

[9] Hu, W. and Tan, Y., 2017. Generating adversarial malware examples for black-box attacks based on GAN. *arXiv preprint arXiv:1702.05983* (2017).

[10] Pham, C., Nguyen, L. A., Tran, N. H., Huh, E.-N. and Hong, C. S., 2018. Phishing-Aware: A Neuro-Fuzzy Approach for Anti-Phishing on Fog Networks. *IEEE Transactions on Network and Service Management*, 15, 3 (2018), 1076-1089. doi: https://doi.org/10.1109/TNSM.2018.2831197

[11] Marchal, S., François, J., State, R. and Engel, T., 2014. Phishstorm: Detecting phishing with streaming analytics. *IEEE Transactions on Network and Service Management*, 11, 4 (2014), 458-471. doi: https://doi.org/10.1109/TNSM.2014.2377295

[12] Lin, Z., Shi, Y. and Xue, Z., 2018. IDSGAN: Generative adversarial networks for attack generation against intrusion detection. *arXiv preprint arXiv:1809.02077* (2018).

[13] Aghakhani, H., Machiry, A., Nilizadeh, S., Kruegel, C. and Vigna, G., 2018. Detecting deceptive reviews using generative adversarial networks, In *Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW).* IEEE.

[14] Anderson, C., 2019. Adversarial Sampling Attacks Against Phishing Detection, In *Proceedings of the 33rd Annual IFIP WG Conferences. Data and Applications Security and Privacy(DBSec2019),* Charleston, SC, USA. Springer. *doi*: https://doi.org/10.1007/978-3-030-22479-0_5

[15] Huh, J. and Kim, H., 2012. Phishing Detection with Popular Search Engines: Simple and Effective *Foundations and Practice of Security*, 6888 (2012), 194-207. doi: https://doi.org/10.1007/978-3-642-27901-0_15

[16] Zhang, Y., Hong, J. and Cranor, L., 2007. Cantina: a content-based approach to detecting phishing web sites, In *Proceedings of the 16th international conference on World Wide Web,* Banff, Alberta, Canada. ACM. *doi*: https://doi.org/10.1145/1242572.1242659

[17] Abu-Nimeh, S., Nappa, D., Wang, X. and Nair, S., 2007. A comparison of machine learning techniques for phishing detection, In *Proceedings of the Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit,* Pittsburgh, Pennsylvania. ACM. *doi*: https://doi.org/10.1145/1299015.1299021

[18] Basnet, R. B., Sung, A. H. and Liu, Q., 2011. Rule-based phishing attack detection, In *Proceedings of the International Conference on Security and Management (SAM 2011),* Las Vegas, NV.

[19] Miyamoto, D., Hazeyama, H. and Kadobayashi, Y., 2008. An evaluation of machine learning-based methods for detection of phishing sites, in *Advances in Neuro-Information Processing*, Springer, 539-546. *doi*: https://doi.org/10.1007/978-3-642-02490-0_66

[20] L'Huillier, G., Weber, R. and Figueroa, N., 2009. Online phishing classification using adversarial data mining and signaling games, In *Proceedings of the Proceedings of the ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics.* ACM. *doi*: https://doi.org/10.1145/1599272.1599279

[21] André, B., Gerhard, P., Luigi, D. A. and Domenico, D., 2010. A real-life study in phishing detection, In *Proceedings of the Proceedings of the Conference on Email and Anti-Spam (CEAS'10),* Perth, Australia.

[22] Sanglerdsinlapachai, N. and Rungsawang, A., 2010. Web phishing detection using classifier ensemble, In *Proceedings of the Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services,* NW Washington, DCUnited States. ACM. *doi*: https://doi.org/10.1145/1967486.1967521

[23] Gansterer, W. N. and Pölz, D., 2009. E-mail classification for phishing defense, in *Advances in Information Retrieval*, Springer, 449-460. *doi*: https://doi.org/10.1007/978-3-642-00958-7_40

[24] Saberi, A., Vahidi, M. and Bidgoli, B. M., 2007. Learn to detect phishing scams using learning and ensemble? methods, In *Proceedings of the Proceedings of the IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Workshops,* NW Washington, DCUnited States. IEEE Computer Society.

[25] Chang, J.-H. and Lee, K.-H., 2010. Voice phishing detection technique based on minimum classification error method incorporating codec parameters. *Signal Processing, IET*, 4, 5 (2010), 502-509. doi: https://doi.org/10.1049/iet-spr.2009.0066

[26] Cheng, H., Wang, P. and Pu, S., 2011. Identify fixed-path phishing attack by STC, In *Proceedings of the Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference,* Perth Australia. ACM. *doi*: https://doi.org/10.1145/2030376.2030396

[27] Liping, M., John, Y. and Paul, W., 2009. Establishing phishing provenance using orthographic features, In *Proceedings of the eCrime Researchers Summit(eCRIME'09),* Tacoma, WA IEEE. *doi*: https://doi.org/10.1109/ECRIME.2009.5342604

[28] Kuan-Ta, C., Jau-Yuan, C., Chun-Rong, H. and Chu-Song, C., 2009. Fighting Phishing with Discriminative Keypoint Features. *IEEE Internet Computing*, 13, 3 (2009), 56-63.

[29] Yearwood, J., Webb, D., Ma, L., Vamplew, P., Ofoghi, B. and Kelarev, A., 2009. Applying clustering and ensemble clustering approaches to phishing profiling, In *Proceedings of the the 8th Australasian Data Mining Conference (AusDM'09),* Melbourne, Australia. CRPIT *doi*: https://doi.org/10.1109/TrustCom.2013.76

[30] Jung, C. and Lee, K., 2010. Voice phishing detection technique based on minimum classification error method incorporating codec parameters. *Signal Processing, IET,* 4, 5 (2010), 502-509. doi: https://doi.org/10.1049/iet-spr.2009.0066

[31] Stanton, G. and Irissappane, A. A., 2019. GANs for Semi-Supervised Opinion Spam Detection. *arXiv preprint arXiv:1903.08289* (2019). doi: https://doi.org/10.24963/ijcai.2019/723

[32] Choi, S.-S., Cha, S.-H. and Tappert, C. C., 2010. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8, 1 (2010), 43-48. doi: https://doi.org/10.13053/CyS-20-3-2457

[33] *Google Trends.* (2019). Retrieved on 20/09/2019 from https://trends.google.com/trends/?geo=US

[34] OpenDNS, L. *PhishTank: An anti-phishing site.* (2016). Retrieved on 20/12/2019 from https://www. phishtank. com

[35] Mohammad, R. M., Thabtah, F. and McCluskey, L., 2014. Intelligent rule-based phishing websites classification. *IET Information Security*, 8, 3 (2014), 153-160. doi: https://doi.org/10.1049/iet-ifs

[36] Yanminglai. *Malware-GAN.* (2019). Retrieved on 22/07/2019 from https://github.com/yanminglai/Malware-GAN

[37] Hammoudeh, Z. *Adversarial Malware Generation Using GANs.* (2019). Retrieved on 20/07/2019 from https://github.com/ZaydH/MalwareGAN