International Core Journal of Engineering Volume 6 Issue 10, 2020
ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

Defense Against Adversarial Attack in Malicious URL Detection

Shuchi Zhang[1], Haoyu Zhang[2], Yiming Cao[3], Qi Jin[4], Ruike Hou[5]

[1]School of Computer Science, University of Nottingham, Nottingham, NG7 2RD, United Kingdom;

[2]School of Informatics, Xiamen University, Xiamen, Fujian 361005, China;

[3]University of California, Davis, 95616, United States;

[4]Vanke Meisha Academy, Shen Zhen, 518000, China;

[5]The Webb School (TN), Bell Buckle 37020, United States.

Abstract

Machine learning has been extensively used in data analyses. Adversarial attack, however, is a huge threaten to machine learning models. By adding indiscernible perturbation or slight changes to initial data, though human beings are unable to distinguish between initial data and data after adversarial attack, models can be confused by these attacks. In image detection, for example, adversarial attack can be generated by simply covering a shadow to the initial image. Abundant research has done dealing with adversarial attack in image detection field, but few is done for malicious detection. The project mainly contains three steps to deal with the problem. First, the project trains a model which can detect malicious URLs. Then, a simulated real-life black-box adversarial is added to the datasets. In the final step, this work generates several weak defenses and assembles them to defend against the adversarial attack.

Keywords

Malicious URL detection, Adversarial attack, Assemble weak defenses.

1. Introduction

In this project, the aim is to find an effective way defending against adversarial attack. The project mainly contains three parts. First, the model is trained and can accurately categorize the URLs in the initial datasets. In this step, TF-IDF method is used to tokenize the URLs. After that, logistic regression is applied to train the model. The model can give out an accuracy of more than 95 percent. Then, a simulation of real-life black-box adversarial attack using perturbation is added to the initial dataset. Some of the characters are replaced by other characters and the sequence of some specific strings are changed. At last, the focus is finding weak defenses and assemble them to an extensible framework which can defend against adversarial attack. Three effective weak defenses are developed. The first is reverse URL method which reverses the URL to train the model. The second is adding some further perturbation to the datasets. And in the last step, the generator of seqGAN is used to make changes on initial datasets.

The purpose of the project is to find an effective way defending against adversarial attack. Hardly can find a proper dataset which contains labeled URLs and URLs with adversarial attack. Therefore, the first two steps should be done before generating method against adversarial attack. In the project, URLs have five different labels, “1” indicates benign URLs and each other has a different property. After training the model, adversarial attack is applied by adding perturbation. The accuracy of model is then decreased to 79% and it is a persuasive accuracy to indicate the power of adversarial attack.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

In the final step of the project, three weak defenses are developed. Weak defenses can provide rational improvement on the accuracy of the model separately in an extent of 5% to 12 %. These defenses are named as weak defenses because the stability of each defense cannot be ensured and need to be assembled. In the project, simple voting strategy is used. After assembling these weak defenses, the result reaches 86%, demonstrating the feasibility of the framework. The framework is extensible by adding more weak defenses and improve the robustness of the frame.

2. Literature Review

There are a series of papers related to the project.

The research (Doyen Sahoo et al, 2017) about malicious URL detection provides a comprehensive survey and a structural understanding of Malicious URL Detection techniques using machine learning [1]. The article provides a timely and comprehensive survey for a range of different audiences further.

In a paper (Juan Ramos, 2003), authors examine the results of applying Term Frequency Inverse Document Frequency (TF-IDF) to determine what words in a corpus of documents might be more favorable to use in a query [2]. Exactly, it is an appropriate way for the project to make data modeling. Meanwhile, in another paper (Chao-Ying Joanne Peng et al, 2002), author demonstrates the preferred pattern for the application of logistic methods with an illustration of logistic regression applied to a data set in testing a research hypothesis [3]. The paper inspired a lot and finally logistic regression is used in the project. Based on these two papers, model is successfully set up in the project to an accuracy of 98%. Certainly, there are some other ways applicable, such as the research (Jingling Zhao et al, 2018), large quantities of URLs are used for detecting web attacks using machine learning models [4]. Based on the dataset and feature selection methods, multi-classification of six types of URLs is explored using the random forest method, which is also compared against the gated recurrent neural networks. It is an efficient and adaptive proactive detection method worthy of learning from. Similarly, another research (Ram B. Basnet et al, 2012) demonstrated that a rule-based method for phishing detection achieves performance comparable to learning machine-based methods, with the great advantage of understandable rules derived from experience [5]. These papers give clear idea on how to use decision tree in URL detection.

For the next part, in one paper (Ying Meng et al, 2020), their team develop an assembly framework which can defend against adversarial attacks in image processing [6]. That project considers 10 kinds of attacks and give 9 weeks defenses. By voting, the framework can work properly. That work gives great inspiration on this project to apply the similar method in network field. This point is similar to another paper (Olakunle Ibitoye et al, 2019), the authors introduce all kinds of adversarial attacks and an adversarial threat model to explain how adversarial attack worked on machine learning model, which offer a train of thought to defend it, like gradient masking etc. [7]. It reminded that it is a feasible way to develop adversarial attacks for URL. Lack of adversarial robustness is mentioned in the research (Aleksander Madry et al, 2019) [8]. To address this problem, authors study the adversarial robustness of neural networks through the lens of robust optimization. They also suggest the notion of security against a first-order adversary as a natural and broad security guarantee. This approach provideds a broad and unifying view on much of the prior work on this topic.

As for generative adversarial nets, the research (Lantao Yu et al, 2017) shows a new way of training generative models, Generative Adversarial Net (GAN) which uses a discriminative model to guide the training of the generative model has enjoyed considerable success in generating real-valued data [9]. In this paper, they proposed a sequence generation framework, called SeqGAN, to solve problems. Modeling the data generator as a stochastic policy in reinforcement learning, SeqGAN bypassed the generator differentiation problem by directly performing gradient policy update. It offers a new thought that train the discriminator and the generator to form new URLs under adversarial attack with the whole structure preserved. Besides, in another research (Zhen Yang et al, 2017), they apply seqGAN in practice [10]. It proposes an approach for applying GANs to NMT. The team build a conditional sequence generative adversarial net which comprises of two adversarial sub models, a

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

generator and a discriminator. The two sub models play a mini-max game and achieve the win-win situation when they reached a Nash Equilibrium. It reminds a deeper thinking about how to build and train the generator and the discriminator, maybe maximizing a reward value, by mapping states to optimal behaviors through reinforcement learning is a better scheme. In terms of GAN as a kind of unsupervised learning, it is necessary for to have clear idea about principle and application of unsupervised learning. So another paper (Jennifer G. Dy, Carla E.Brodley, 2004) is introduced. The research identifies two issues involved in developing an automated feature subset selection algorithm for unlabeled data: the need for finding the number of clusters in conjunction with feature selection, and the need for normalizing the bias of feature selection criteria with respect to dimension [11].

3. Project Overview

In the first step, dataset need to be decided. The dataset used is from CIC. The dataset should simply have two columns: one for URL and another one indicating the type of it. To keep the dataset simple and close to real life settings. All further operations are based on the initial dataset. (Fig 1)

	url	Label
99995	http://allgorltogaparty.co.ik/gallery2/maon.ph...	4
99996	http://3401.e-prontphoto.co.ik/thososessex/ond...	4
99997	http://acard4i.co.ik/prodict_onfo.php?cPath=10...	4
99998	http://aboitscotland.co.ik/ecosse/portessoe/on...	4
99999	http://alloanceleagie.co.ik/cgo-bon/phpBB2/voe...	4

Fig 1. The structure of dataset

Then, the initial model is trained to an accuracy of 98.9% and simulated adversarial attack to attack our model with an accuracy of 79.6%.

In the final step, defense framework is developed to our model. The basic idea is creating several weak defenses and assembling them. The reason to name these defenses as weak defense is the efficiency of these defenses may not be guaranteed. It is for the sake of that in real life, how adversarial attack is attacked is not known to the developers of defense system. Therefore, different ways to alter the URLs after adversarial attack need to be tested and decide which weak defenses can be assembled to final framework. Finally, three useful weak defenses are found. The first is reverse-URL method. The idea is to reverse the URL and combine it with the initial one to train the model. The accuracy can reach 85 percent. The second weak defense is GAN. In the project, we decided to choose seqGAN (sequence generative adversarial networks) which is a kind of unsupervised learning as our strategy. In order to get appropriate results, the structure of URLs is retained which includes HTTP protocol, focus on word segmentation, and then pretrain the generator and discriminator. After pretraining, we enter the original data and the output of the generator into the discriminator. In the process of training, some seasonable strategies is added to make it apply to URL. To get a satisfying result, it is necessary to repeat the process for several times. The last is perturbation. Like the change have done on initial dataset, some further perturbation and the accuracy can be improved. After creating these weak defenses, weak defenses are assembled by voting. The result with most supporters will be marked as final result. It is confident say that the result is promising with the accuracy of 86 percent.

The project contains mainly five steps. Except for the procedure have been mentioned in project overview, the team did some research on different topics and spent some time on finding proper datasets. Shuchi and Haoyu are responsible for the task arrangement. Shuchi did jobs on doing research and read through related papers on the topic. For the coding part, he created the frame for

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

the whole project including loading the dataset, training the initial model, simulating adversarial attack, creating some weak defenses and assemble these defenses. Haoyu provided the idea of how to create the defense framework and provided the initial datasets. In the coding part, he did a lot on trying to use GAN in adversarial attack and finally created some weak defenses. Qi attempted to find a proper way to use GAN in our project and also gave some idea on simulating adversarial attack. Ruike and Yiming gave some useful idea on weak defenses and improve the performance.

4. Technical Section

The project is mainly divided into three key sections. The first step is to train the initial model which can distinguish different kinds of URLs without adversarial attack. The second step is to simulate real-life black-box adversarial attack. And finally, an extensible framework is created to defend adversarial attack.

4.1 Train Initial model

The initial datasets are stored in a csv file, all URLs are in the first column and all URLs have a label indicating its property. Therefore, all these URLs should be firstly loaded from the csv file and put all URLs in the list called "url_list", labels in another list called "y". The "url_list" is in the form of:(Fig 2)

```
url_list.head()
```

```
0    http://1337x.to/torrent/1048648/American-Snipe...
1    http://1337x.to/torrent/1110018/Blackhat-2015-...
2    http://1337x.to/torrent/1122940/Blackhat-2015-...
3    http://1337x.to/torrent/1124395/Fast-and-Furio...
4    http://1337x.to/torrent/1145504/Avengers-Age-o...
```

Fig 2. URLs

As can be seen from the image above, URLs are in the form of string and is unable to train the model. Therefore, these URLs should be tokenized first. TF-IDF (term frequency–inverse document frequency) method is used in our project. TF-IDF can indicate the frequency of a specific word in the text. TF is the frequency of a word in the whole text. IDF (inverse document frequency) is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. Mathematically, IDF can be expressed as:

$$idf(t, D) = log \frac{N}{|\{d \in D : t \in d\}|}$$

TF-IDF is applicable for that malicious URLs always have a distinctive format and the frequency of some words is extremely high. For example, for example, in malware URLs, the character '%' occurs more frequently than other characters. (Fig 3)

Where N means total number of documents in the corpus and denominator means number of documents where the term t appears. In sklearn, the TfidfVectorizer will deal with all URLs and convert strings to sparse matrices. The data is then able to train the model. To be specific, the URL "www.foo.com/1" will be tokenized to ['www','ww.','w.f','.fo','foo','oo.','o.c','.co','com','om/','m/1']. And in the final matrix, the value for TF-IDF of each sub string will be replaced one by one in order. (Fig 4)

The dataset from "FinalDataIni.csv" has totally 100000 rows of data. The first 50000 rows are the initial dataset needed to train the initial model and the last 50000 rows are generated by changing

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

some of the characters in the initial dataset and add to the end of the file. The last 50000 rows of data will be used to verify the feasibility of adversarial attack. Therefore, only first 50000 rows are needed to train and test the model. A sparce matrix of 50000 elements is needed. The following two code blocks are trying to generate the sparce matrix needed. (Fig 5)

```
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%96%87%E6%98%8E%E7%A4%BC%E4%BB%AA%20%E5%89%AA%E8%B4%B4%E7%94%BB/,2
%E6%A0%91%E5%8F%B6%E7%B2%98%E8%B4%B4%E7%94%BB%20%E5%9B%BE%E7%89%87/,2
%E6%A0%91%E5%8F%B6%E7%B2%98%E8%B4%B4%E7%94%BB%20%E5%9B%BE%E7%89%87/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
%E6%AF%9B%E7%BA%BF%E8%B4%B4%E7%94%BB%20%E5%9C%BA%E6%99%AF%E5%9B%BE/,2
```

Fig 3. Samples of malware URLs

```
# Using Tokenizer
vectorizer = TfidfVectorizer()

# Store vectors into X variable as Our XFeatures
X = vectorizer.fit_transform(url_list)

X

<100000x73974 sparse matrix of type '<class 'numpy.float64'>'
        with 1285179 stored elements in Compressed Sparse Row format>
```

Fig 4. Structure of sparse matrix after dealt by tokenizer

```
X_initial = X[0]
for i in range(49999):
    ele = X[i + 1]
    X_initial = vstack((X_initial, ele))
```

```
y_initial = []
for i in range(50000):
    y_initial.append(y[i])
```

Fig 5. Code blocks dealing with data

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

After retrieving the data needed, the dataset is divided to training data and test data with a ratio of 80:20. Finally, with the help of logistic regression in sklearn, our initial model is able to detect different kinds of URLs with an accuracy of 98.96%, which we think is a promising result.(Fig 6)

```
# Accuracy of Our Model
print("Accuracy of our model is: ",logit_initial.score(X_initial_test, y_initial_test))
```

```
Accuracy of our model is:  0.9896
```

Fig 6. Accuracy of initial model

4.2 Simulate Real-life Black-box adversarial attack

In the second step, adversarial attack is simulated. Adversarial attack is dangerous to machine learning models. In real-life, it threatens image detection. By adding indiscernible changes to the initial image, the model will not be able to detect the image. For example, in the field of image detection to detect numbers, an image is initially recognized as 7. Then, researchers manually add a shadow to the image, though to human beings, it still looks like 7, but the model recognizes it as 9. (Fig 7)

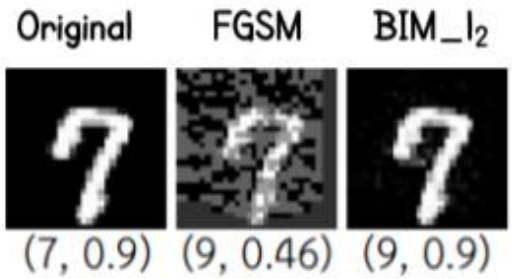

Fig 7. Weak defenses in image detection

Though not much research is done in the field of adversarial attack in malicious URL detection, we think it worth to be done. Our initial idea is to use the generator in GAN to generate the dataset in adversarial attack. However, the result given under this idea is not ideal. The initial data is largely damaged, the similarity of the generated data is lower than our expectation which does not seems like a dataset for adversarial attack. Since data after adversarial attack should close to the initial data and inspired by adjacency swap method in shortest route problem, the adversarial attack is finally done by changing some random characters in URL and swap the sequence of some string. All 'x's and 'i's are swapped and all "cd" are changed to "dc". The URLs after adversarial attack are still similar to the initial one:(Fig 8)

```
http://ok~.y^/xs/bist/q-%D0%B2%D0%B5%D0%BB%D0%BE%D1%81%D0%B8%D0%BF%D0%B5%D0%B4/?page=9
http://ok~.y^/is/bxst/q-%D0%B2%D0%B5%D0%BB%D0%BE%D1%81%D0%B8%D0%BF%D0%B5%D0%B4/?page=9
```

Fig 8. Comparison between initial URL and URL after adversarial attack

But the accuracy of our model has been decreased to less than 80%. (Fig 9)

```
print("Accuracy of our model is: ",logit_initial.score(X_attack, y_attack))
```

```
Accuracy of our model is:  0.79938
```

Fig 9. Accuracy of classifier after applying adversarial attack

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

4.3 Weak Defenses

Inspired by Athena framework in image detection field, the project decided to create a defend system by assemble several weak defenses. Weak defenses may not perform effectively to defend against the adversarial attack, but each could to some extent improve the accuracy and finally by voting, the overall accuracy should be increased. It should be an extensible framework that later developers can create more weak defenses to the framework. Our team has tried four different kinds of weak defenses but finally accepted three of them which perform positively.

4.3.1 Reverse URL

The first weak defense we use id to reverse the URL and combine it with the initial URL. Such a method is widely used in image detection field. In Athena framework, one of the weak defenses to defend against adversarial attack is to reverse the image and then train the model. The accuracy of the model can improve. In the framework, a reversed URL is generated by method "reverseString()": (Fig 10)

```
def reverseString(s):
 if s == "":
    return s
 else:
    return reverseString(s[1:])+s[0]
```

Fig 10. Method to reverse string

Then the generated string is combined with the initial one. For instance, "www.abc.com" will be replaced by "www.abc.commoc.cba.www". After dealing with all data in the dataset, same jobs are done as training the initial model, and the accuracy is increased to 84.316%.

It might seem hard to explain the reason why the accuracy is increased. But it might make sense that after reverse the URL and combine it with the initial one, the information contained in each URL is more than the initial URL. It might be the reason of the increase of the accuracy.

4.3.2 Perturbation

The second weak defense selected in the final defense framework is perturbation. The dataset for adversarial attack is generated by adjacent swap combined with character swap. It is a special type of perturbation. Inspired by the denoise method in Athena framework, all the 'i' is replaced with 'u' in the initial dataset and attacked dataset. (Fig 11)

```
url_list_per = []
for i in range(len(url_list)):
    url_list_per.append(url_list[i].replace("i",'u'))
```

Fig 11. Method to replace character

Then train the model and test the same as what has done in training the initial model. The result is the accuracy can reach 91.164%.

4.3.3 Sequence Generative Adversarial Nets

SeqGAN, a kind of unsupervised learning, is used to get the data under adversarial attack. Based on the confrontation, real data and the output of the generator G will be used to train the discriminator D. However, the output is discrete, it is difficult for D to update G gradiently, so D needs a complete sequence to score, which is to use MCTS (monte carlo search tree) to complete possibilities of every movement. D translates reward back to G according to the complete sequence. This is a way of reinforcement learning, and a trained network can produce the next optimal action (Fig 12).

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

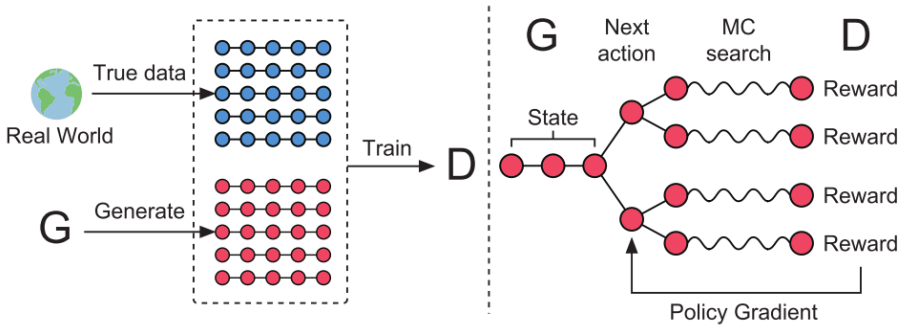

Fig 12. The overall process of seqGAN [9]

The specific progress of the algorithm is as follows:(Fig 13)

Algorithm 1 Sequence Generative Adversarial Nets

Require: generator policy G_θ; roll-out policy G_β; discriminator D_ϕ; a sequence dataset $\mathcal{S} = \{X_{1:T}\}$

Initialize G_θ, D_ϕ with random weights θ, ϕ.
Pre-train G_θ using MLE on $\mathcal{S}$
$\beta \leftarrow \theta$
Generate negative samples using G_θ for training D_ϕ
Pre-train D_ϕ via minimizing the cross entropy
repeat
  for g-steps **do**
    Generate a sequence $Y_{1:T} = (y_1, \ldots, y_T) \sim G_\theta$
    for t in $1 : T$ **do**
      Compute $Q(a = y_t; s = Y_{1:t-1})$ by Eq. (4)
    end for
    Update generator parameters via policy gradient Eq. (8)
  end for
  for d-steps **do**
    Use current G_θ to generate negative examples and combine with given positive examples $\mathcal{S}$
    Train discriminator D_ϕ for k epochs by Eq. (5)
  end for
  $\beta \leftarrow \theta$
until SeqGAN converges

Fig 13. The specific progress of seqGAN [9]

First of all, we tried to make word segmentations on the basis of punctuations like '/', '-', ':' and so on, which aimed to ignore these punctuations and preserve the basic structure of URL (Fig 14).

```
# ignore specific characters
def ifIgnored(x, ignored=string.punctuation):
    if x not in ignored:
        return False
    else:
        return True
```

Fig 14. Ignore specific punctuations

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

Besides, we pretrained the generator with long short-term memory algorithm, and then pretrained the discriminator with the original data and output of the generator. After that, we tried the process that trained the generator based on reward and then trained the discriminator again with new output of updated generator. In this process, we made specific rules where only a part of URL would be intercepted to be attacked and not every character would be changed. The intercepted part would be changed according to ASCII Code.

Code for this part is in the package "GAN".

4.3.4 Information Extraction

It is tried to use information extraction as one of our weak defenses in the final framework. The basic idea was that some of the irrelevant information in the URL might confuse our model and affect the accuracy such as "http" and ".com". And tests on trying to remove these strings in our dataset and then train the model again. Unfortunately, the accuracy is not improved but decreased to 79.81%. Such a failure might be caused by a loss of information in information extraction procedure. Some of the property might lose even these strings seems to be irrelevant.

4.4 Assemble Weak Defense

Voting, the simplest but useful and fair method is used to assemble our weak defenses. The result of a specific URL will be the most common result among all weak defenses. For the sake of that sklearn does not support such an operation, the framework is created as:(Fig 15)

```
result = []
for i in range(49999):
    result_count = [0, 0, 0, 0, 0]
    key_result = logit_key.predict(X_key_attack[i])[0]
    result_count[key_result] += 1
    gan_result = logit_gan.predict(X_gan_attack[i])[0]
    result_count[gan_result] += 1
    per_result = logit_per.predict(X_per_attack[i])[0]
    result_count[per_result] += 1
    temp = max(result_count)
    for j in range(5):
        if temp == result_count[j]:
            result.append(j)
```

Fig 15. Use voting to decide final result

The first step is to decide the result for all URLs and store it in the list "result". Then, compare "result" with the initial label which is stored in "y_initial". The accuracy can be calculated and reaches 85.6%. Which is not a huge increase but still promising comparing to 79%.

5. Evaluation of Results

There are totally three different weak defenses accepted in the final defense framework. The first is reverse URL weak defense. The reason for its feasibility is after reversing the URL and combining it with the initial URL, the information contained in each URL is increased. The second method is perturbation, which can be applied by replacing characters or changing the sequence of some string. In our project, 'i' is replaced with 'u' and the accuracy increased more than 10 percent. To be honest, such an improvement is abnormal. Some other replacements are tried and though the accuracy improved in these trials but is not as much as this one (most of the accuracy is between 84-87 percent). But in our perspective, such a method is acceptable because the accuracy increased in most of the trials. The last weak defense is using seqGAN and its generator to generate a new dataset and then

ISSN: 2414-1895 DOI: 10.6919/ICJE.202010_6(10).0053

feed data into the model. The generator of seqGAN can change the dimension of initial data but does not change the key information of initial data so it can be applied as a weak defense.

We are confident with the result we have got overall and in the final defense framework, we think these weak defenses are extremely important influencers. weak defenses should be stable because in real-life, no one will have the idea how adversarial attack is generated thus weak defenses should be able to deal with most of these attacks.

It is interesting to notice that the final accuracy is lower than some of the weak defenses. It is because voting is used to decide the result and since there is no large number of weak defenses, the final accuracy might be lower than some of the weak defenses.

6. Conclusion

In conclusion, our group focus on the problem about defending attack towards malicious URL detection problem. Initially, a model is successfully trained and can detect malicious URLs with 98%. And adversarial attack is simulated in the project to attack the model trained. To deal with such a problem, an extensible framework is developed, and the accuracy is improved. As few research in the field has been done, the project can inspire other researchers dealing with adversarial attack in fields other than image detection.

References

[1] D. Sahoo, C. Liu and S. Hoi, "Malicious URL Detection using Machine Learning: A Survey", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/1701.07179. [Accessed: 07- Aug- 2020].

[2] J. Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries", Semanticscholar.org, 2020. [Online]. Available: https://www.semanticscholar.org/paper/Using-TF-IDF-to-Determine-Word-Relevance -in-Queries-Ramos/b3bf6373ff41a115197cb5b30e57830c16130c2c.[Accessed:07-Aug- 2020].

[3] C. Peng, K. Lee and G. Ingersoll, "An Introduction to Logistic Regression Analysis and Reporting", The Journal of Educational Research, vol. 96, no. 1, pp. 3-14, 2002. Available: 10.1080/ 00220670209598786.

[4] J. Zhao, N. Wang, Q. Ma and Z. Cheng, "Classifying Malicious URLs Using Gated Recurrent Neural Networks", 2020.

[5] R. Basnet, Cs.nmt.edu, 2020. [Online]. Available: https://www.cs.nmt.edu/~rbasnet/research/ Rule Based Phishing Attack Detection Final.pdf. [Accessed: 07- Aug- 2020].

[6] Y. Meng, J. Su, J. O'Kane and P. Jamshidi, "Ensembles of Many Diverse Weak Defenses can be Strong: Defending Deep Neural Networks Against Adversarial Attacks", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2001.00308. [Accessed: 07- Aug- 2020].

[7] O. Ibitoye, R. Abou-Khamis, A. Matrawy and M. Shafiq, "The Threat of Adversarial Attacks on Machine Learning in Network Security --A Survey", arXiv.org, 2020. [Online]. Available: https://arxiv.org/ abs/ 1911.02621. [Accessed: 07- Aug- 2020].

[8] A. Madry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks", arXiv. org, 2020. [Online]. Available: https://arxiv.org/ abs/1706.06083. [Accessed: 07- Aug- 2020].

[9] L. Yu, W. Zhang, J. Wang and Y. Yu, "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/1609.05473. [Accessed: 07- Aug- 2020].

[10] Z. Yang, W. Chen, F. Wang and B. Xu, "Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets", arXiv.org, 2020. [Online]. Available: https://arxiv.org/ abs/ 1703. 04887. [Accessed: 07- Aug- 2020].

[11] D. Jennifer, Jmlr.org, 2004. [Online]. Available: http:// jmlr.org/ papers/ volume5/ dy04a/ dy04a. pdf. [Accessed: 07- Aug- 2020].