

DSCI 550 Project 2 - Report

Team Name: MIMECRAFT

Team Member: Joshua Huang, Saumya Shah, Sungho Lee, Xinran Liang, Yue Liu

Implementing TTR Algorithm

Taking the output results from our Assignment 1, we filtered the emails based on attack types such that if the value of the URL Label column is "bad", then the attack type is malware. Since each email was tagged with multiple attack types we labeled them based on a priority starting with Credential Phishing as highest priority, then Malware, then Social Engineering, and lastly Reconnaissance. Thus we got our 4 sets of attack types consisting of 200 emails each for our next tasks.

Once parsing the original email text through Tika, we could obtain XHTML format in the "X-tika:content" part. It consists of several metadata (starting with "<meta name= blah blah />"), title and body part. To implement the Text to Tag Ratio, we simply divided it into line by line, and checked if there are any tags and the texts in the given line. For example, a line "<body><p>FROM:MR. JAMES NGOLA." consist of two tags and 19 letters (ignore space) so the ratio became $19/2 = 9.5$. We kept all the lines if the ratio of the line is greater than or equal to 1. The result showed that our TTR kept the title and the body of the text.

Generation of fake emails using GPT-2

Based on the output file obtained from the above step, we had a separate content into different attack type files and leveraged them to train our gpt-2 model. Along with email content the training file also included the "From" and "Title". We set the model training parameters to use the 355M "medium" model, with temperature set to 0.8. Although we required 200 fake emails per attack type, we generated 600 samples to accommodate for any "bad" emails generating garbage values. We excluded emails that were too long (> 400 tokens) or too short (< 20 tokens). Several emails consisted of characters like "@" or "\n" randomly, so counted the occurrences and excluded such emails.

We observed all the emails to follow a structural format of "From:", "Title" and email content. While some emails were remarkably accurate and could not be distinguished from those written by a human, few were simply blank with random spacing and abrupt terminations. Several features like all-capitalized email text were observed since our training dataset also had several emails having all-capitalized email text and so did our fake email dataset replicate it. The other prominent features of including numbers and percentages within the email body were also observed. For Credential phishing specifically, the attack style of asking for personal information is observed in most of the fake emails.

Attack Types	Sample size for training GPT-2	Output Sample size after clean-up
Reconnaissance	775	314
Social Engineering	740	329
Malware	1085	309
Credential Phishing	998	239

Generation of fake Phish Iris & Facial Images using DCGAN

Task	Sample size for training	Training parameters
Reconnaissance	292	Noise_size = 50 LR_D = 0.000003 LR_G = 0.0001 Batch_size = 7 Epochs = 1500
Social Engineering	115	Noise_size = 50 LR_D = 0.000003 LR_G = 0.0001 Batch_size = 5 Epochs = 1500
Malware	233	Noise_size = 50 LR_D = 0.000003 LR_G = 0.0001 Batch_size = 7 Epochs = 1500
Credential Phishing	673	Noise_size = 100 LR_D = 0.000003 LR_G = 0.0001 Batch_size = 20 Epochs = 1500
Face generating	100,000	Noise_size = 100 LR_D = 0.00004 LR_G = 0.0002 Batch_size = 64 Epochs = 50

We first separated the Phish Iris dataset into four attack type groups based on the content. We use <https://www.birme.net/> to pre-process the Phish Iris images, as they need to be resized, cropped, renamed and converted to jpg format.

Then, we test the AnimeFaces generator code provided by the professor. It requires Tensorflow 2.2, CUDA 10.1 and cuDNN 7.6.5 to run on local PC in GPU mode. We soon realize that the input image size is hard coded to 64x64. Hence, we decide to switch to another DCGAN model - the Face generator. This model allows 128x128 size input images and can adjust discriminator and generator learning ratio, batch size, noise size and other parameters. But the Tensorflow version control is pretty tricky. For this model, we have to pip install gast 0.2.2 and numpy 1.16.4 before TensorFlow-GPU 1.14. And we also need to downgrade to CUDA 10.0 and cuDNN 7.6.5 to make it work with TensorFlow 1.14.

Please see the detailed training parameters setup table on the left side. All training parameters are tested and modified to make the G loss and D loss about the same around 0.6 - 0.8 at the end of the training.

Unfortunately, the result is still very blurry even with very big epoch numbers. We believe it is due to our relative small training dataset size and limited computational power(NVIDIA Geforce GTX 1080Ti GPU).

GROVER Implementation

We test the GROVER model on a local PC in GPU mode but encounter multiple issues. Clearly, training the discriminator of the GROVER model requires TPU. Hence, we switched to Google Colab pro to continue the journey.

In order to prepare the qualified training dataset, we pull together both the original fraud emails from the assignment one and some emails we generated from the GPT2 model earlier. We only keep the email content as the 'article' key value. For the 'label' key value, we assign 'human' to the original fraud emails and 'machine' to the GPT2 generated emails. Finally, we set the 'split' key value to 'train' for all the emails in the training input dataset.

Thanks to the codes provided by Yifeng (Carl) Shi on [slack](#), we successfully mount the TPU in Google Colab to train our own GROVER discriminator model.

After training, we pull all 2,400 GPT2 generated emails together to set up a testing dataset. This time, we assign 'machine' as the 'label' key value and 'test' as the 'split' key value to all emails. We then run the newly trained discriminator model to predict if an email in the testing dataset is written by human or machine. The result is very accurate.

Tika Image Caption for the Phish Iris Image

Data preparation: From the DCGAN step, we've got Phish Iris images in .png type for all emails and, to indicate the division of 4 categories, we saved images into 4 different zip files. Besides, because images in the same category are hard to be identified, we then named them by numbers (from 001 to 850).

In the reconnaissance image folder, images' average size is 54KB, in social engineering folder, it's 52KB, in malware folder, it's 53KB and in credential phishing folder, the average is 48KB. Based on the difference between images' size, if we got a batch of images within the same type, we may be able to figure out their type by checking their average size. This performs like a "reverse categorization".

Captioning process: we traversed every image in folders, uploaded it to the REST API, and got return values one by one. The returned value is a list of dictionaries; under the key, "sentence" in dictionaries, we found descriptions for the input image. While collecting returned values, we thought of a potential improvement by adding the batch processing or parallel processing functionalities into the tika docker. With those functionalities, we may just need to input a folder path and will get results back together in json format. This improvement will reduce time costs when we are doing captioning on tremendously many images.

Generating attacker and victim Replies using GPT-2

For generation of fake reply emails we used the Enron data set having a consolidated corpus of all reply emails. As our first step we classified the Enron email data corpus into different attack types by running the code from our Assignment 1. Thus, we got our training sample set with count as - Social Engineering: 634, Malware: 4041, Reconnaissance: 422, Credential Phishing: 629 emails. Using gpt-2 and the training data from Enron we got a trained model, one for each attack type and used this for the next step. Then we used a context based gpt-2 model to generate the fake replies. We used a context file containing the attacker emails so that our responses have a similar context and we used the reply email trained model from above. Thus we get our first set of attacker to victim email and response pair.

For generating the next response from attacker to victim we used the same reply trained model for each attack type from Enron data set. This time our context file included the victims responses in the context based gpt-2. Thus we obtain the fake replies back from attacker to victim.

Generally, we did not observe much relation or context based similarity between the response emails. The emails followed the same style and patterns as we obtained while generating the first fake reply emails. The observations here are similar to the ones discussed in section 'Generation of fake emails using GPT-2'. Below is one example of email responses:

Attacker to Victim	Victim to Attacker	Attacker to Victim
<pre>From: fred owell Title: GOOD DAY GOOD DAY I am a citizen of israel,the land of 2000 years.I came to know of you in my work as a Civil Engineer. I am now pleaded to contact you to serve as my partner.I have a project to undertake. I want to establish a business in your country and I need your partnership in this regard. My name is Fred owell from southern province of engr. and am the manager of a rubber and timber business in my country, Singapore. I have a consignment containing more than ten(10) tons of which I intend to ship out to any country of your choice for immediate shipment. I have been given the task of investigating and finding a reliable trustworthy person who can assist me in this regard. My investigation proved to me that he is a trustworthy person who can assist me in this regard. I know that this proposal will come to you as a surprise, but believe it or not,it was God-ordained that I should seek for partners in this regard. I have considered all the risks involved and the difficulty of transporting such a huge amount of money to a country of anywhere. But it needs a partner who can stand on as the beneficiary and partner to assist me in this regard. I need your strong assurance and trust to help me secure this fund for shipment out of my country. Please acknowledge your request by a response through my email address below and include your telephone,fax and mobile phone numbers for easy communication. Please I shall use this contact to get into detailed information necessary to help you get this transaction off the ground. Please this transaction is 100% risk free. Your role in this transaction is absolutely essential that you remain totally confidential. Thanks for your anticipated partner cooperation. Yours faithfully, Fred owell</pre>	<pre>Title: Re: GOOD DAY Dear fred owell, What do you recommend for delivery? The estimated delivery time is 12 March. This time will allow shipping from Hong Kong (1-2 days) and delivery of the package by Hong Kong air to Singapore (12 March). It has also been suggested you should choose "Hong Kuei, East - Hong Kong" as it has the cheapest shipping options and is the easiest choice. Best wishes you have enjoyed your holiday in a pleasant setting. Regards</pre> <p>Regards, E </p>	<pre>From: fred owell Title: Re: Re: GOOD DAY Message: Thanks bud. You have been a source of wisdom to me on getting your email address. I will keep in touch. Cheers and Blessings, Edward Owell</pre>

Here the attacker is asking the victim to assist him in some shipment of huge amounts of money and the victim replies back asking him about the delivery recommendations. In response to this the attacker thanks the victim for agreeing with him. Hence we observe a flow in the conversation.

Additional Observations

If questions from the assignment instruction are not here, they have been already addressed in previous analysis sections.

1. Did GPT-2 trained on specific attack emails generate realistic replies and conversation?

The overall meaning of the emails being generated did not make sense in a good way, but we find that part of the emails sentence are related to the previous emails. Here is one example email, generated from Credential Phishing.

AttackerType_index: CrePhi. 76-1 <Initial email from attacker to victim>

... Dear Friend, I wrote to seek your help for a money transfer. ...

... I am requesting your assistance as the person who will act as the new owner of the fund...

AttackerType_index: CrePhi. 76-2 <reply email from victim to attacker>

... Could you tell me the name of the company that holds the accounts and what is their principal business line?

Also, could you point me to someone who can explain the concepts better. ...

... I suggest that you contact the companies directly. ...

AttackerType_index: CrePhi. 76-3 <reply email from attacker to victim>





... I'd like to call you. I can't find your phone number. ...

2. What was the quality of the generated Phish Iris data representing multimedia attacks?

The quality of the generated Phish Iris images is very poor. It is because the training dataset is very limited, and we run it on a local PC which has limited computational power.

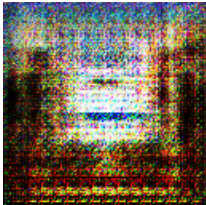
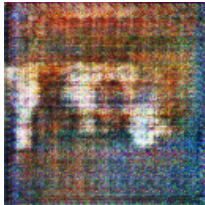
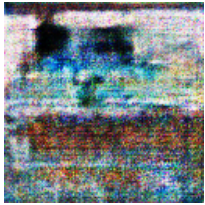
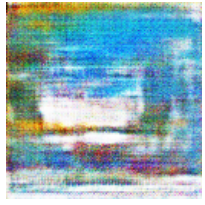
3. Did DCGAN generate believable faces? How could you have improved on it?

Yes, the DCGAN generated some believable faces. We can increase the epoch number to improve the result if we have a device with higher computational power.

Good result		Bad result	
			
face_generator-0194	face_generator-1097	face_generator-0108	face_generator-1478

4. Did the Image captions for the Phish Iris datasets make sense?

The captions are not really precise on reflecting the objects that we are expecting. For example, credential_phishing-609.png has the captions: "a group of people riding on top of a large body of water; a group of people riding on top of a wave; a group of people riding on top of a wave in the ocean". However, in credential phishing emails, what people usually could see should be the login pages or options for entering account and password information, or other things related. (More examples seen below)

			
social_engineering-105.png	malware-069.png	reconnaissance-162.png	credential_phishing-195.png
Caption: a room filled with lots of luggage and a window	Caption: a group of birds sitting on top of a wooden deck	Caption: a black and white dog sitting on top of a bed	Caption: a bird is perched on the edge of a body of water

5. Was Grover able generally to identify the falsified attacks?

Yes, our GROVER discriminator model trained on more than 4,000 emails is able to correctly identify the falsified emails generated by the GPT2 model.

6. Was GPT-2 more effective training on only the TTR text, or did the other additional features help? Did you try them?

Besides email body content, we included the “title” and the “from” field in our training dataset. We observed that the fake emails follow the exact pattern as specified in the training dataset.

7. Also include your thoughts about the ML and Deep Learning software like GPT-2, Grover, Image Captioning, DCGAN, etc. – what was easy about using it? What wasn't?

In general, all of the ML and Deep learning softwares we used in this assignment are open source research softwares. A common issue for this type of software is lacking detailed technical documentation and often encountering package/setup outdated issues. And they all demand very high computational resources. The GPT-2 and DCGAN are pretty straightforward and easy to set up. But generating acceptable outcomes requires hours of GPU time. On the other hand, the set up process for Grover and image captioning is a little tricky, as Grover discriminator requires TPU and image captioning requires docker.

References

USCDataScience/tika-dockers: <https://github.com/USCDataScience/tika-dockers>

Contributors: Chris A. Mattmann, JPL & USC, Thejan Wijesinghe, University of Moratuwa

GROVER: <https://github.com/rowanz/grover.git>

Contributor: Rowan Zellers

GROVER - TPU set up in Google Colab: <https://uscdatasience.slack.com/archives/C01QKMLGQF9/p1617430619214900>

Contributor: Yifeng (Carl) Shi

Phish Iris dataset: <https://web.cs.hacettepe.edu.tr/~selman/phish-iris-dataset/>

Contributor: F.C. Dalgic, A.S. Bozkir and M. Aydos

Face generator: https://github.com/gsurma/face_generator

Contributor: Greg (Grzegorz) Surma

Face generator - Celebrities-100k training dataset: <https://www.kaggle.com/greg115/celebrities-100k>

Contributor: Greg (Grzegorz) Surma

AnimeFaces: <https://github.com/chris mattmann/DCGAN-AnimeFaces>

Contributor: Chris Mattmann

GPT-2: <https://github.com/minimaxir/gpt-2-simple>

Contributor: Max Woolf (@minimaxir)

Context Based GPT-2: <https://github.com/bengum/gpt-2>

Contributor: Ben Gumser

Contribution:

- **Joshua Huang:** Design TTR algorithm, pre-process phish iris images, use DCGAN face generator model on local PC to train and generate fake faces and phish iris images, use GROVER model on Google Colab to train and run discriminator
- **Saumya Shah:** Generation of fake emails using GPT-2, Generation of fake attacker -> victim response emails using GPT-2, Kaggle to generate fake faces however for performance we switched to a local GPU PC
- **Sungho Lee:** Generation of inputs for GPT-2 & Grover, Design TTR algorithm, Generation of fake emails using GPT-2(initial email & reply emails), generation of new_attacks.tsv & featureTable_v2.tsv.
- **Xinran Liang:** Design TTR Algorithm, Install tika-dockers and do image captioning, extract flags from GROVER's output
- **Yue Liu:** Generation of fake emails using GPT-2, phish iris images, Use GROVER model on Google Colab to train and run discriminator.