

# Possible attack Method on Phishing Email Classifier Using Genetic Algorithm

Veronika Stephanie

## Abstract

Over the past few years, phishing has become a severe threat to global businesses and organizations, causing financial loss and distrust in the use of email for communications. Though different phishing detection approaches have been proposed, the growth rate of phishing scam victims keeps increasing. One reason is that people usually overlook some aspects that also play a significant part in phishing detection model effectiveness (e.g., the dataset used for training the model). This paper proposed a phishing email modification framework as a possible attack to the current phishing email classifier based on Genetic algorithm and GloVe (Global Vector) semantic analysis to generate phishing email variation that can bypass the phishing email classifier while maintaining its malicious intent and its semantics. Concurrently, this paper also presents an evaluation method to validate the refined phishing email regarding its structure, intent, and semantics.

**Keywords**— phishing email, genetic algorithm, machine learning, natural language processing, semantics analysis

## 1 Introduction

Nowadays, email has become one fundamental component that is used by most organizations to communicate (e.g., exchanging confidential business information). This results in an increase in email usage to fool email users to obtain sensitive information. This type of email is better known as a phishing email. In the past few years, phishing attack has caused significant loss to some businesses and degradation of trust in the use of email as a tool for business communication. Though users carefulness and experiences may prevent them from falling into phishing

scam, complete prevention is not possible [1]. This is proven by an increase in compromised information by 35 percent over a year since 2017 [2].

The advancement of machine learning has made it possible to restraint the increase in the phishing email scam success rate. One of the latest existing works includes a Recurrent Convolutional Neural Network (RCNN) model-based for phishing email detection called THEMIS by [3], which produces a classification accuracy result up to 99.848%. Another prior work in email phishing detection proposed by [4] also considers the use of Natural Language Processing (NLP) model in tackling the problem. This approach includes the use of word count, stopword count, and punctuations in the phishing detection process, which are usually omitted by most of the studies in this corresponding field. This model also achieve an outstanding results in giving true positive and true negative values with a success rate of 83% and 96% respectively.

Though there are many existing variation of phishing scam prevention models, the victim rate based on this attack keeps increasing over the years. One problem is due to the imbalance of publicly available phishing and benign email samples, which eventually hinders the development in this research field. In most recent studies, some researchers usually obtain the phishing email data through the institution or the company mail server [5]. Despite the urgency to develop phishing prevention methods, sharing phishing emails publicly for some institutions or businesses may not be possible as most phishing emails directed to them may contain private information. There are only few phishing email dataset that is currently publicly available. One example is the Nazario phishing email dataset [6]. Furthermore, it is apparent that the difference of the benign and the phishing emails available is significant, making it easy for phishing email detection model to classify the email type. For example, an excessive amount of url can be found in the phishing email dataset while the benign email dataset only consist of short plain text. This significant difference may not be apparent in real world scenario as phishing emails becoming more similar to that of the benign emails. As such, the existing model for phishing detection may not be effective.

Currently, phishing methods have become more common and more sophisticated. The current technologies have made it possible for the adversaries to automate their malice intent which results in the phishing attack acceleration and the ability to reach to a larger group of victims. Some methods used include mass mailing that incorporate action words to proceed to a certain link; however, this action mostly can be easily detected by the phishing classifier [7]. Some studies

have proposed a possible offense method based on popular Natural Language Processing method to deceive the phishing classifier. The purpose is to study a possible attack pattern that may be used by the attacker and to enhance the current phishing classifier. However, most prior study on email content generation has yet to successfully complete two main objective of the phishing based attack, which is to fool the phishing email classifier and its victim. Most study are focused on the prior objective while neglecting the later as automated text generation that takes into account the email structure and grammar, while maintaining its semantics and intent is usually treated as a separate objective.

The contribution in this paper aims to create a phishing email generation technique that can reconstruct phishing emails to an extent where the reconstructed email can bypass the existing phishing email detection methods while maintaining the original phishing email structure, semantics and its malice intent. This can resolve the shortage of phishing email samples and help future study to overcome this possible attack pattern. Concurrently, this paper also proposes an evaluation method to test the quality of the newly reconstructed phishing email in regards to its semantics, content structure, and phishing intent.

## 2 Literature Review

Phishing email detection has recently gained more interest due to the alarming growth rate in phishing scams. However, due to the lack of existing phishing email data, this research field’s development has been constrained. Nevertheless, since the year 2016, there were only few studies regarding the construction of phishing emails as a possible way to tackle this issue.

The current work done by [5] tries to reproduce benign and phishing emails by balancing the feature in both the benign and the phishing emails to prevent significant biases in classifying the data. Examples of the imbalance feature pointed out in the study can be seen in the Nazario email data set. Comparing both benign and phishing email features from the Nazario data set [6], it is clear that the phishing emails are mostly represented in HTML format or contains an excessive amount of link, images, and hyperlink while the benign email data set mainly consist of simple conversations in plain text data. The study proposes a benign email resource data insertion to remove these biases. The resource data are generated using six different HTML resource generators (metadata generator, hyperlink generator, image source generator, external link generator, css style generator, and

script generator). The insertion method is controlled by a control sequence to determine the resource insertion type in the email, and a quantity sequence, which determines the generator’s resource quantity to be inserted. Both the control sequence and the quantity sequence work based on the randomness principle, which means that the amount of the resources and the insertion sequence of the resources are chosen randomly. The proposed method shows a promising results based on true positive and true negative value classification. However, the random insertion of resources causes the email to have an inconsistent structure. Furthermore, the research focuses on balancing both features without considering the email semantics and the modification of the email content, which may only fool the phishing email classifier but not possibly the recipient of the emails.

Another study by [7] used Natural Language Generator (NLG) methods, specifically a Recurrent Neural Network Language Model (RNNLM) in generating phishing email. The model was trained using a set of benign emails and a specified ratio of phishing emails. Though the model is able to generate a paragraph of email body, further investigation on the semantics and the grammar within the paragraph needs to be done. Moreover, there is an absent of phishing value in most of the generated phishing emails. Amongst several emails that have been generated, only one is able to show a slight intent in obtaining the victim’s information. Also, the nature of RNNLM used in the mechanism does not always guarantee a complete generation of email sentences, resulting in an incomplete email structure.

To complete this objective of phishing emails refinement that can deceive both the email classifier as well as the email recipient, we investigated several different text generation methods that existed as possible tools to reconstruct phishing email body.

Over the past decades, researchers have tried to build Natural Text Generation (NTG) models. Building a language model is a challenging task, given that there are several things to be taken into account such as the semantics of the text, grammar, word usage, and language perception. In tackling this issue, early studies explored two major approaches: rule based and learning based [8].

Rule-based approach usually relies on the use of modules or templates with gaps to be filled with. It is best to be applied when a defined template is to be reused [9]. This approach usually results in a high quality text due to a major control that we have on the model output [9]. Historically, rule-based approaches has been used for data-to-text generation [10] and is still widely used until now, for instance, weather report. Although this technique gives us full control on what

to be produced, it is a rigid and inflexible method [8]. Its development is cost intensive and time consuming.

In contrast to rule-based approach, training based methods is more fluent and more flexible. There are many different ways to generate text based on this approach. Since early 2000s, many text generation models based on training approaches have been proposed. For example, Markov Chain models for text summarization [11], text information extraction [12], and even text classification [13]. In the process, this model produces a sequence of outcome which is determined by the probability that is hold by the previous state. This model is categorized as a statistical text generation approach that can produce a simple yet clear sentences while maintaining the semantics and syntactic characteristics of the language [14]. Nevertheless, there are uncertainties in the text prediction due to the error propagation nature in the model as a result of the uncertainties in its inputs and parameters [15].

Generative Adversarial Network (GAN) is another well known under this category [16]. Due to the success in computer vision task such as natural image samples generation, many tried to explore GAN's potential in text generation fields [17]. [17] proposed a method called SeqGAN, which is a sequential text generation model which employ gradient policy techniques to train its generator. RankGAN by [18] train its generator by rewarding a higher score on a more realistic text sequence in the current batch. LeakGAN by [19] proposed a GAN based techniques which considers the discriminator as a manager that provides intermediate information for the generator. These proposed methods have successfully provide a promising results in generating long text. Though that is the case, the models keeps failing to avoid common issues such as mode collapse and training instability [20]. These common issues lead the models to generate a less varied text and a great performance instability when the hyper-parameters are randomly chosen. In addition to that, a large dataset is needed to train the model as it also affects the models performance.

While the rule based approach may be costly and training based language model performance can be greatly impacted when insufficient number of training data is provided, to successfully answer the two aforementioned objectives, we considered a genetic algorithm approach for text generation while utilizing semantic analysis as its fitness function.

## 3 Background

### 3.1 Genetic Algorithm

Genetic algorithms is a evolutionary algorithm that was introduced in 1975. Just like the other evolutionary algorithm, genetic algorithm is characterized by the existence of populations and individuals. In any genetic algorithm, the first step of its implementation is to generate these two components, which then followed by evaluation, selection, crossover, and mutation process, as shown in Figure 3.1.

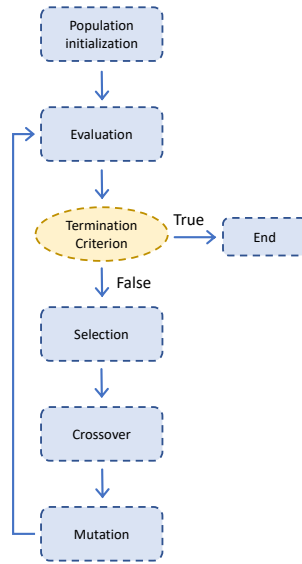


Figure 3.1: Genetic Algorithm Overview

During the population initialization process, originally, each member or individual of the population is represented as a binary string with a fixed length  $L$ . In most cases, individuals are generated randomly. After the initialization process, each individual is passed through an evaluation process to determine its fitness value. This fitness value provides measurement of each member in term of its survival rate in the next generation and reproduction opportunity rate for the next offspring. In regards to the reproduction opportunity, there are ways to select the parents for the next offspring. One way is through tournament selection. This method takes  $n$  random individuals that will compete with each other for reproduction chance. Individual with the best fitness value is selected for the reproduction process and put into the *intermediate population*.

The crossover process takes place after the selection phase is finished. Throughout the crossover process, each pair of individuals within the *intermediate population* combine their genetic information to generate a new offspring as shown in Figure 3.2

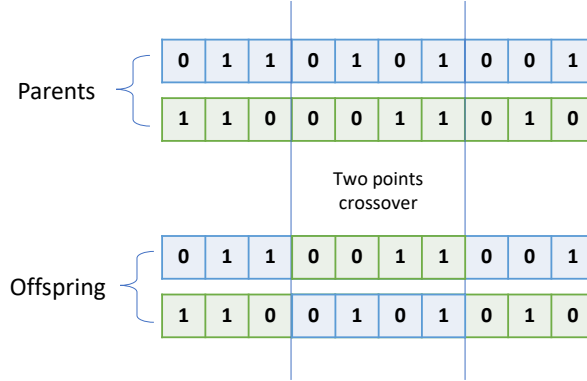


Figure 3.2: Example of genetic algorithm crossover technique

In the final phase of the genetic algorithm, the mutation happens to each of the offspring. The success rate of the mutation is determined by a defined mutation probability. This mutation purposes is to maintain the diversity of the generated offspring. Example of the mutation can be seen on Figure 3.3 where some components within the individual are mutated to another components.

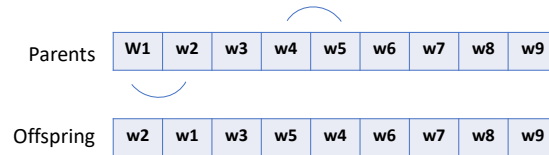


Figure 3.3: Example of genetic algorithm mutation technique

All the processes except for the population initialization are repeated until the termination criteria is met.

### 3.2 Semantic Similarity with GloVe (Global Vector)

Semantic similarity analysis is one domain in Natural Language Processing (NLP). It enables us to tell whether two texts represent a similar meaning. This can be done by comparing each components within the text. GloVe [21] is an

unsupervised learning algorithm that aims to capture words' relation and meaning by generating vector representation of word in the Euclidean space based on the words co-occurrence [22].

To give an in depth explanation on GloVe mechanism, we denote some notations. Assume that there is a matrix  $X$  used to store word-to-word co-occurrence  $X_{ij}$ , where  $X_{ij}$  represents the  $j$  word occurrence in the same window context of the word  $i$ . Let  $X_i = \sum_k X_{ik}$  be the number of times any word appears in the context of  $i$  and let  $P_{ij} = P(j|i) = X_{ij}/X_i$  be the probability of the word  $j$  occurs in the context of  $i$ . An embedding of a (target) word  $i$  is written  $w_i$ , while an embedding of a context word  $k$  is written as  $\hat{w}_k$ .

Initial study of the GloVe technique suggests to learn embeddings as to satisfy the Equation 3.1.

$$w_i^T \cdot \hat{w}_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i) \quad (3.1)$$

Assuming that  $\log(X_i)$  is a bias term of the word  $i$  [23], the equation can be further simplified as shown in Equation 3.2

$$w_i^T \cdot \hat{w}_k + b_i + \hat{b}_k = \log(X_{ik}) \quad (3.2)$$

, where  $b_i$  is the bias of the word  $i$  and  $\hat{b}_k$  is the bias for  $k$  to maintain the symmetry.

The author defined that the cost function using a least squares mechanism as follows:

$$J = \sum_{i,k} (w_i^T \cdot \hat{w}_k + b_i + \hat{b}_k - \log(X_{ik}))^2 \quad (3.3)$$

However, this function has a drawback where it weights all co-occurrences equally (e.g., stopwords that occurs frequently such as 'the', which contribute less in representing the meaning of the word, weighted the same as the word that actually contributes to the meaning of the word), which leads to rare co-occurrences words being treated as noisy data and carry less information.

To address this problem, the author introduced a weighted function  $f(X_{ij})$  into the cost function gives us the model

$$J = \sum_{i,k}^V f(X_{ij}) (w_i^T \cdot \hat{w}_k + b_i + \hat{b}_k - \log(X_{ik}))^2 \quad (3.4)$$

, where  $V$  is the size of the vocabulary.



## 4 Methodology

In this section, we discuss data pre-processing technique used on the collected phishing email, and the proposed phishing email reconstruction framework.

### 4.1 Data Retrieval and Preprocessing

Datasets that were used in this project were phishing email collected by Cornell University in "Phish Bowl" database [24] and IWSPA phishing email dataset [25]. Since the data structure and the content of the emails were not aligned with what was needed, we considered several pre-processing steps:

- Grammar check: Since most of the collected phishing data had purposely misspelled some words, and it can greatly affect the performance of our proposed architecture, we tried to use language tool python library to fix the spelling and grammar errors.
- Unnecessary email removal: Some emails that were retrieved had a totally random content or had no meaningful text with random alphabets. To remove these emails, we tried to see the percentage of words that can be detected by the spelling checker library, enchant, python.
- Entity recognition: We also tried to replace known entities, such as organizations name, date, and products to corresponding labels (e.g., *< ORG >*, *< DATE >*, *< PRO >*) using en-core-web library from Spacy python.
- Email structure refinement: Email structure is an important aspect in contributing to the final output, hence, we tried to manually check all of the pre-processed email and re-structure all of the email to a decent email structure, meaning that the emails have to have at least a greeting, body and closing.

### 4.2 Architecture Overview

Figure 4.1 shows an overview of the proposed mechanism. As can be seen from the figure, the process can be divided into 3 major processes, namely, email filtering and sentence selection, sentence generation, and evaluation and sentence replacement. These processes are discussed further in Section 4.2.1, Section 4.2.2, and Section 4.2.3 respectively.

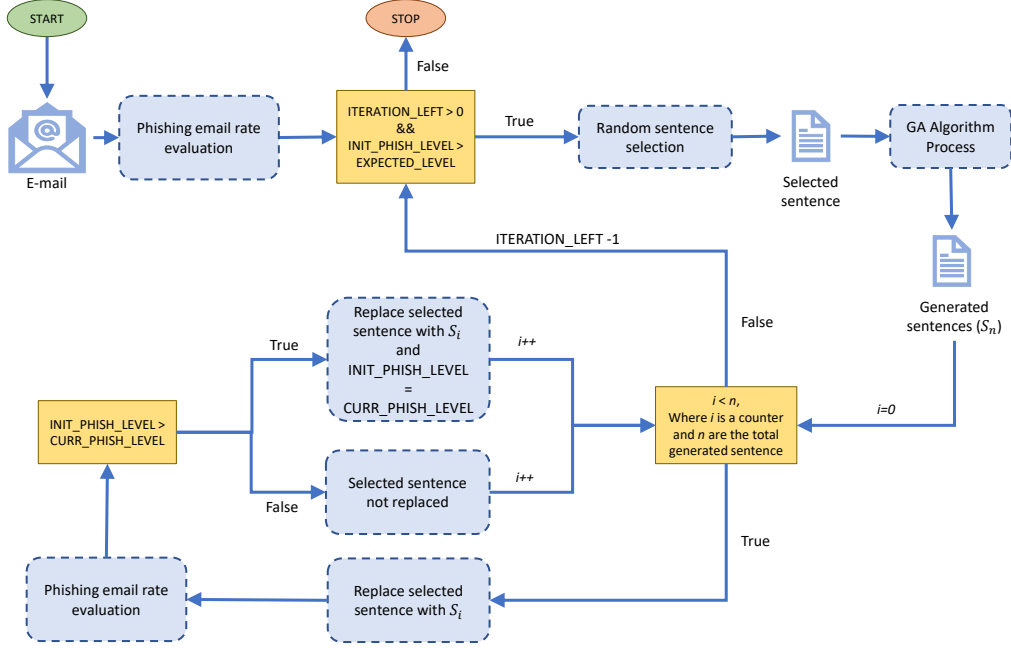


Figure 4.1: Email generation architecture overview

#### 4.2.1 Email Filtering and Sentence Selection

The first step in the implementation starts with checking an individual email against the phishing classifier. In this architecture, Themis model [26] is used to decide the phishing rate. When the email does not satisfy the expected phishing level, the model saves the phishing rate as the initial phishing rate value and the email is passed through the next step, which is the random sentence selection. During the random sentence selection, a defined portion of the email onset is not included in the random selection. This also applies to the email closure as the model is not intended to modify the greeting and the closure of the email body. After the sentence is selected, it will be forwarded to the next process which is the sentence generation.

#### 4.2.2 Sentence Generation

For sentence generation, we utilize Genetic algorithm and GloVe semantic analysis as its fitness function. In the process, suppose that we have a randomly selected sentence from the email. First, the sentence is split into an array of words ( $W_n$ ). For example, ['Thank', 'you', 'for', 'using', 'the', 'hsbc', 'bank', 'online',

'transfer']'. For each word  $W_i \in W_n$ , we extract the synonyms and put each word's synonym into an array as shown in Figure 4.2.

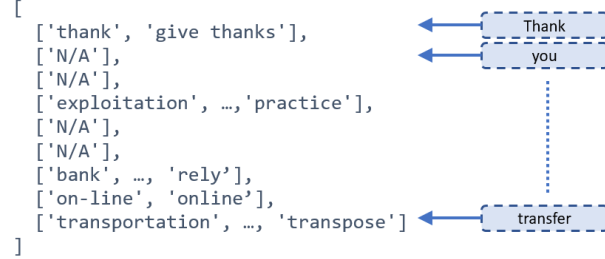


Figure 4.2: List of synonyms corresponding to each word

While extracting the synonyms, we tried to avoid modification on stopwords and known entity such as organization name, time, and people's name. Thus, instead of providing synonyms, we put 'N/A' to inform the model not to modify those words. This list of synonyms acts as a possible replacement of the corresponding words.

Achieving random possibilities in replacing each words with its synonyms using genetic algorithm requires several steps. Firstly, the genetic algorithm is assigned to generate a fixed individual length, which is equals to the total number of words within the sentence. Each individual is represented as randomly generated number. The range of the randomly generated number is based on the maximum number of synonyms over all of the words' synonyms. For example, from Figure 4.2, assume that the maximum synonyms is from word transfer, which has 32 synonyms, so the range of the randomly generated number is from 0 to 31. [31, 2, 3, 1, 14, 2, 3, 5, 22] is one sample of the generated individual. Each of the number in the individual corresponds to each words of the original sentence. From this numbers, to get the modified sentence, each of the number will act as the index for the corresponding words' synonyms. However, since the range of the randomly generated number is from zero to the maximum number of synonyms, it will cause an index out of range for the words with lesser synonyms. To overcome this, we modulus each randomly generated number with the total number of synonyms for each corresponding words. The generation of individuals happens as much as  $n$  times, where  $n$  is the required number of individual within a population.

After the individual is generated, each of them will be passed through four main phases, namely, evaluation, selection, mating, and mutation. In the evaluation phase, each individual is assessed based on the fitness function. To determine

the fitness of the individuals within the population, each individual is checked against the initial selected sentence in terms of its semantics using GloVe semantic analysis. This fitness score will determine if they will be eliminated from the population when the new offspring is generated. The lower the score of the semantic similarity is, the higher the chance it is to be eliminated. In the selection phase, a tournament based selection is used to choose the best-fit for the next reproduction. This technique requires a fixed number of tournament size which determines the number of individuals that will be included in the competition. Suppose that a population has a size of 500 individuals and a tournament size of 20. In the population, 20 individuals are selected randomly and compared with each other in term of its fitness score. The best individual with the highest fitness score is then selected and appended to the new population. This process is repeated as much as the number of individuals within the previous population as we use the 1:1 replacement ratio. After the new population is obtained, the process continues

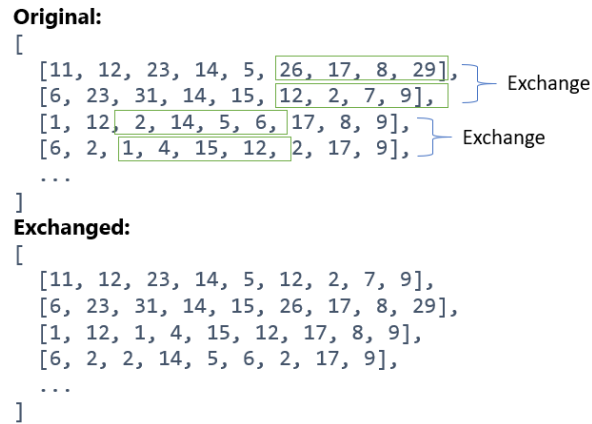


Figure 4.3: Two-point cross-over genetic algorithm

to the mating and the mutation phase. We considered using two-points crossover in the mating process, in which two individuals will exchanges their component while keeping their original length as shown in Figure 4.3. This process will occur based on the defined probability throughout the whole individuals within the population. This also applies for the mutation, except in the mutation, instead of exchanging components between individuals, it shuffles the component within each individual. Finally, the new generation will replace the old generation with the least-fit individual. The process will keep iterating until it meets the termination condition.

### 4.2.3 Evaluation and Sentence Replacement

In the final stage, the top  $n$  generated sentence from the previous step are tested against the phishing email classifier by replacing the selected sentence with each of the generated sentence. For each iteration of sentence replacement, the newly reconstructed email is checked against the Themis model. Its score is then compared with the existing phishing rate. The email with lower phishing rate is considered to replace the previous email if its phishing rate is lower than the previous phishing level. The whole process will keep iterating if the maximum number of iteration has not been met or if the final generated email has not satisfy the expected phishing level.

## 5 Results and Discussion

In this section, we provide discussions on the testing environment, experimental setup, and the results of the proposed work.

### 5.1 Testing Environment

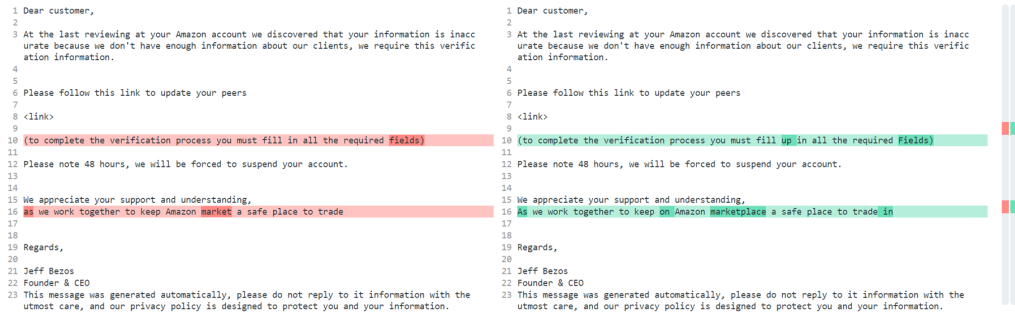
For the experiments, we used a Windows 10 Home computer with Intel(R) Core(TM) i7-7500U CPU (Processor Base Frequency 2.70GHz and Max Turbo Frequency 2.90GHZ) and 16 GB RAM. The experiments on the proposed approach were carried out using the Visual Studio Code with Python version 3.7.

### 5.2 Experimental Setup

In analysing our proposed approach, we setup several parameters and used several existing models. For the expected phishing score, we set the maximum threshold of 50 percent for the email to be classified as a benign email. So, when the email's phishing score is below 50 percent, the process will be terminated. For the Themis classifier, we used a Themis pre-trained model to classify the email. The model itself is trained with 32,000 emails. As for the Global Vector model, we used a pre-trained model from wikipedia web crawling with 200d vectors.

## 5.3 Model Performance

From the experiment, we modified over 300 phishing emails. Figure 5.1 shows the original (left) and the modified (right) phishing email. From the modification, the original email is detected as phishing email with a phishing score of 87% while the modified email is detected as a benign email with a phishing score of 11%.



<pre> 1 Dear customer, 2 3 At the last reviewing at your Amazon account we discovered that your information is inacc urate because we don't have enough information about our clients, we require this verific ation information. 4 5 6 Please follow this link to update your peers 7 8 &lt;link&gt; 9 10 (to complete the verification process you must fill in all the required fields) 11 12 Please note 48 hours, we will be forced to suspend your account. 13 14 15 We appreciate your support and understanding. 16 As we work together to keep Amazon market a safe place to trade 17 18 19 Regards, 20 21 Jeff Bezos 22 Founder &amp; CEO 23 This message was generated automatically, please do not reply to it information with the utmost care, and our privacy policy is designed to protect you and your information. </pre>	<pre> 1 Dear customer, 2 3 At the last reviewing at your Amazon account we discovered that your information is inacc urate because we don't have enough information about our clients, we require this verific ation information. 4 5 6 Please follow this link to update your peers 7 8 &lt;link&gt; 9 10 (to complete the verification process you must fill up in all the required fields) 11 12 Please note 48 hours, we will be forced to suspend your account. 13 14 15 We appreciate your support and understanding. 16 As we work together to keep on Amazon marketplace a safe place to trade in 17 18 19 Regards, 20 21 Jeff Bezos 22 Founder &amp; CEO 23 This message was generated automatically, please do not reply to it information with the utmost care, and our privacy policy is designed to protect you and your information. </pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 5.1: Phishing email sample generated when utilizing genetic algorithm

Figure 5.2 shows the sample phishing email generated with GPT-2 transformer. The generated sample can bypass the phishing classifier with a score of 25%

We recently reviewed your e-gold account, and suspect that it has been accessed by an unauthorized third party features.

To restore your account to the security level you have set it to , please send your account data to us and we will immediately repair your account.

To get started, please follow these steps :

Sign in to the e-gold account.

<link>

Sincerely,

The e-gold online team.

We apologize for any inconvenience this may cause, and appreciate your assistance in helping us maintain the integrity of the entire e-gold system matter.

Figure 5.2: Phishing email sample generated when utilizing GPT-2

Figure 5.3 shows the success rate of two different text generation models when integrated to the proposed framework. It is apparent from the figure that the success rate of Genetic algorithm in bypassing the existing Themis phishing classifier is 10 out of 300, while the GPT-2 has a success rate of 4 out of 15.

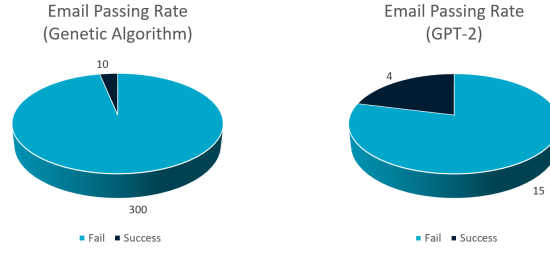


Figure 5.3: Success rate comparison between genetic algorithm and GPT-2 text generation model

## 6 Conclusion and Future Work

In this project, we presented a phishing email body modification technique. The proposed approach adapted a text generation technique based on Genetic algorithm and a semantic similarity analysis model based on Global Vector (GloVe). With the proposed approach, we were able to generate variations of phishing email which maintain the structure and the semantics of its original input email while also able to deceive the phishing email classifier to a certain extent. However, to further improve the proposed approach, an in depth investigation on the text generation used and exchanging the method used (e.g., replacing text generation phase technique with grammatical evolution) in the framework is encouraged to compare the efficacy of different state of arts on the current proposed framework.

## References

- [1] K. Greene, M. Steves, and M. Theofanos, “No phishing beyond this point,” *Computer*, vol. 51, no. 6, pp. 86–89, 2018.
- [2] A. APWG, “Phishing activity trends report: 3rd quarter 2017,” *Anti-Phishing Working Group*. Retrieved April, vol. 30, p. 2018, 2018.
- [3] Y. Fang, C. Zhang, C. Huang, L. Liu, and Y. Yang, “Phishing email detection using improved rcnn model with multilevel vectors and attention mechanism,” *IEEE Access*, vol. 7, pp. 56 329–56 340, 2019.
- [4] G. Egozi and R. Verma, “Phishing email detection using robust nlp techniques,” in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, IEEE, 2018, pp. 7–12.

- [5] G. Yu, W. Fan, W. Huang, and J. An, “An explainable method of phishing emails generation and its application in machine learning,” in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, IEEE, vol. 1, 2020, pp. 1279–1283.
- [6] J. Nazario, *Phishing corpus*, 2007.
- [7] A. Das and R. Verma, “Automated email generation for targeted attacks using natural language,” *arXiv preprint arXiv:1908.06893*, 2019.
- [8] C. van der Lee, E. Krahmer, and S. Wubben, “Automated learning of templates for data-to-text generation: Comparing rule-based, statistical and neural methods,” in *Proceedings of the 11th International Conference on Natural Language Generation*, 2018, pp. 35–45.
- [9] T. C. Ferreira, C. van der Lee, E. van Miltenburg, and E. Krahmer, “Neural data-to-text generation: A comparison between pipeline and end-to-end architectures,” *arXiv preprint arXiv:1908.09022*, 2019.
- [10] C. van der Lee, E. Krahmer, and S. Wubben, “Pass: A dutch data-to-text system for soccer, targeted towards specific audiences,” in *Proceedings of the 10th International Conference on Natural Language Generation*, 2017, pp. 95–104.
- [11] J. M. Conroy and D. P. O’leary, “Text summarization via hidden markov models,” in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 406–407.
- [12] Y.-z. Liu, Y.-p. Lin, and Z.-p. Chen, “Text information extraction based on hidden markov model [j],” *Acta Simulata Systematica Sinica*, vol. 3, p. 038, 2004.
- [13] K. Yi and J. Beheshti, “A hidden markov model-based text classification of medical documents,” *Journal of Information Science*, vol. 35, no. 1, pp. 67–81, 2009.
- [14] A. Astigarraga, J. M. Martinez-Otzeta, I. Rodriguez, B. Sierra, and E. Lazkano, “Markov text generator for basque poetry,” in *International Conference on Text, Speech, and Dialogue*, Springer, 2017, pp. 228–236.



- [15] K. Benke, S. Norng, N. Robinson, L. Benke, and T. Peterson, “Error propagation in computer models: Analytic approaches, advantages, disadvantages and constraints,” *Stochastic Environmental Research and Risk Assessment*, vol. 32, no. 10, pp. 2971–2985, 2018.
- [16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
- [17] L. Yu, W. Zhang, J. Wang, and Y. Yu, “Seqgan: Sequence generative adversarial nets with policy gradient,” in *Thirty-first AAAI conference on artificial intelligence*, 2017.
- [18] F. Juefei-Xu, R. Dey, V. N. Boddeti, and M. Savvides, “Rankgan: A maximum margin ranking gan for generating faces,” in *Asian Conference on Computer Vision*, Springer, 2018, pp. 3–18.
- [19] J. Guo, S. Lu, H. Cai, W. Zhang, Y. Yu, and J. Wang, “Long text generation via adversarial training with leaked information,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.
- [20] W. Nie, N. Narodytska, and A. Patel, “Relgan: Relational generative adversarial networks for text generation,” in *International conference on learning representations*, 2018.
- [21] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
- [22] A. Tifrea, G. Bécigneul, and O.-E. Ganea, “Poincaré glove: Hyperbolic word embeddings,” *arXiv preprint arXiv:1810.06546*, 2018.
- [23] Y. Sharma, G. Agrawal, P. Jain, and T. Kumar, “Vector representation of words for sentiment analysis using glove,” in *2017 international conference on intelligent communication and computational techniques (icct)*, IEEE, 2017, pp. 279–284.
- [24] IT@Cornell, *Phish bowl*, available: <https://it.cornell.edu/phish-bowl>, 2021.

- [25] IWSPA, *The first security and privacy analytics anti-phishing shared task*, available: [https://dasavisha.github.io/IWSPA-sharedtask/?tdsourcetag=s\\_pctim\\_aiomsg](https://dasavisha.github.io/IWSPA-sharedtask/?tdsourcetag=s_pctim_aiomsg), 2021.
- [26] Y. Fang, C. Zhang, C. Huang, L. Liu, and Y. Yang, “Phishing email detection using improved rcnn model with multilevel vectors and attention mechanism,” *IEEE Access*, vol. 7, pp. 56 329–56 340, 2019. DOI: 10.1109/ACCESS.2019.2913705.