URL-based phishing detection

ZIHUI LI, MRIDUL BANIK, SHREYA ANILKUMAR, HOSSEIN SHIRAZI, and INDRAKSHI RAY, Colorado State University, USA

1. ABSTRACT

Recently online surfing security has become a significant topic. More and more attackers try to deceive online users by mimicking a uniform resource locator (URL) and a webpage. In order to detect phishing URLs from legitimate URLs, researchers began to explore machine learning and natural language processing techniques.

This paper proposes implementing and improvising the performance of a lightweight phishing detection algorithm that distinguishes phishing from legitimate websites. Phishing detection through a machine-learning and natural language processing approach has become quite effective now[11]. The proposed framework is based on several linguistic and URL based features. Experiments are performed on an already established model, using numerous Huggingface [14] Fill-Mask and Text Classification based transformers such as BERT, Roberta, Albert etc with their results being analyzed and debugged in the case of unsatisfactory results. BERT is a bidirectional transformer, pre-trained using a combination of masked language modeling objective and next sentence prediction on a large corpus comprising the Toronto Book Corpus and Wikipedia [3]. Roberta and Albert is an extension of Bert, i.e. A Robustly Optimized BERT Pretraining Approach and A Lite BERT for Self-supervised Learning of Language Representations . Owing to the effects of phishing attacks, it is crucial for the development of a compatible URL based phishing detector that can be used conveniently among common people.

The innovations of this work are as follows:
We make a comparison about the performance of various NLP transformer models on the URL phishing dataset.This is the first time analyzing the reasons that cause performance difference of models BERT, ROBERTA and ALBERT.

2. INTRODUCTION

One of modern cybersecurity issues is phishing. Scammers mainly use emails, text messages and attractive hyperlinks to trap the recipients into revealing information such as passwords, account

numbers, and credit card details. In order to address this issue, researchers have explored various methods to detect illegal websites. Previous work shows that the classification outcome has high accuracy for most of the main ML and NLP classifiers. But the processing procedure cost much training time due to sophisticated processing flow series. On the basis of that, [5] proposed an effective lightweight phishing detection technique based on BERT - A NLP transformer model. Since URL has unique tokens that different the task from normal NLP task, although BERT and ELECTRA have been proved to perform good on URL legitimate-phishing dataset, the reason why they perform well and the comparison between other transformer models have not been discovered yet. In this paper, we compared the fine-tuned results based on BERT, ROBERTA and ALBERT using the URL phishing dataset. Further, it compares the performance and explores the reasons that caused the difference.

3. MOTIVATION

In today's world, life is unthinkable without the usage of the internet. With the onset of Covid-19, the world connects through the internet irrespective of work or for leisure. With human dependence so heavily on the internet, data can often be subjected to malicious and illegal attacks. Scammers mainly use emails, text messages and attractive hyperlinks to trap the recipients into revealing information such as passwords, account numbers, and credit card details[4]. Statistics show that almost 50% of emails and text messages are spam. The attackers try to deceive online users by mimicking a uniform resource locator (URL) and a webpage[13].These attacks pose a threat on reputations, loss of data, direct monetary loss, reduction in productivity and value. Cautious victims can detect phishing on visiting the website, however detecting an attack just from the URL isn't common among the general public, although these URLs have several characteristics which make them distinguishable from the usual website links. Hence there is a need for accurate classification of URLs based on their structure into authentic and malicious websites.

The common structure of a legitimate URL is protocol://domain-name.top-level-domain/path
Quoting an example, A well-written URLs can serve as their own anchor text when copied and pasted as links
Example Anchor Text . When the destination is clear, it can be concluded as a legitimate URL.
Hence, this brings a need to make the phishing detection scalable. Thus paving the way for a need for NLP techniques and machine learning strategies.

4. RELATED WORK

URL-based phishing detection is lightweight and in large demand, there are multiple approaches to detect malicious websites. According to the previous research, the author [6] checks the similarity/distance between new URL and known legitimate URLs, or take the websites URLs that the user logged in before as reference [2] to determine if a website is phishing, both of them have largely demand on previous knowledge to form a URL white list. However, the number of websites has risen sharply and phishing techniques have become more and more sophisticated. It is nearly impossible to collect all the URLs from the internet, which results in high false positives. The black URL list based detection methods have similar drawbacks.

There are some researches focused on heuristic approaches to detect malicious sites, [9] achieved the detection by building a signature database using the information collected from attackers, this detecting system cannot be arranged without anti-virus applications on devices. Besides, it performs poorly when encountering novel attack patterns because the predefined signature database misses newly generated features since the updates of the database always lag behind novel attacking techniques. On the basis of that, Nguyen [8] proposed a new heuristic detection method based on URL components and domain rank, the author designed a metric to evaluate the value of each component in of a URL and compare a fixed threshold with metric value. The author tested 9,661 phishing websites and 1,000 legitimate sites and claims accuracy reaches 97%. However, the heuristic PrimaryDomain is still based on white list of already known websites and the heuristic PageRank requires online computing that is connected to Google. Besides, the experiment needs an equal division between phishing and legitimate websites. In 2017, Ebubekir [1] proposed a system that was used for feature extraction using NLP technique. They conducted many tests that using features extracted from NLP models and applied features to the created system, the Random Forest algorithm has been proved to be the best algorithm. It also proved that NLP is an effective tool to extract URL features.

Feature engineering is another significant part in the Machine Learning and NLP area [12]. An appropriate feature list contributes to high accuracy of classification. In 2018, Tan extracted 48 features from HTML context and URLs and from 5000 legitimate sites from Alexa, CommonCrawl.org and 5000 phishing sites from PhishTank.com and OpenPhish.com, these features were used in literatures [10][5][15].

To boost the performance of classifiers, in 2020, Li [7] combined linear transformation using distance metric learning method and non-linear transformation based on Nystrom method then

validated their newly proposed feature extraction approach. Overall the outcome has high accuracy for most of the main ML and NLP classifiers. But the processing procedure cost much training time due to sophisticated processing flow series.

One of the restrictions blocking the current phishing detection research is the amount of available phishing sites and legitimate sites. Enlightened by that, Hossein [10] greatly expanded the amount of phishing websites by mimicking the existing sites using an Adversarial Autoencoder.On the basis of it, Haynes and Hossein [5] proposed a standalone URL based phishing detection approach which utilizes NLP transformers BERT and ELECTRA trained on large URL dataset. It shows that this approach obtains high accuracy while requiring less training time and has no dependency on URL preprocessing. In addition, it only requires URLs without any HTML context or other application on a device.

5 METHODOLOGY

We initially had two CSV files as our dataset which we used as train and test dataset. The CSV file consists of two columns. One is for the URL and another is for the label. We label our dataset as 0 and 1. 0 stands for Legit urls and 1 stands for Phishing URLs. In CSV files there are an unequal number of phishing and legit urls. That inequality can make training and test sets imbalanced. To solve this problem, we introduced a function get5050 which will return an equal number of urls in both legit and phishing categories and in both training and testing dataset. Pandas dataframes are used to store returned data by get5050. We then had two different data frames to store Train and Test set.We used regex tokenizer from nltk library and used as splitter to split all the urls of Train and Test dataframe. The splitted list is then processed and cleaned with multiple functionalities. At first the urls were converted to lowercase and encoded with UTF-8 encoding. All the space of URls then removed with strip() function. After applying all this processing, we were having a train dataframe with length 24310 and test data frame with length 6078 with 50-50 phishing-legit url.

To classify whether an URL is a phishing site or Legit, we have used Text Classification and Fill Mask based Transformer models. We used a bunch of transformer based pre-trained classifiers. We then compared the result of each pre-trained classifier. There are some popular transformer based language models such as BERT, RoBERTa, Albert and their variations. In our classifier we used seven different versions of BERT and RoBERTa, Albert and their variations to classify our task.

We will be going to explore different variations of these transformer based models and document our results.

6. EXPERIMENTATION

The experimentation part was all conducted using python programming language to train various Machine learning models and subsequently use ideal NLP techniques. The experimentation was divided into three major sections namely, data preprocessing, modeling and evaluation of results. The trained data consists of various URLS that were tokenized to be ideal for the models. On completion of preprocessing, various transformer models were imported from the transformer package. Common packages used for almost all the experimentations are DataCollatorWithPadding,TFAutoModelForSequenceClassification, model respective TokenizerFast, SequenceClassification, Trainer and rainingArguments. The experimentation consisted of testing seven models and computing their accuracy, precision, recall and F1 score as evaluation parameters.

A confusion matrix was used to compute the evaluation parameters. This matrix is a table that is used to describe the performance of a classification model. The components of a confusion matrix are TruePositives , FalsePositives,TrueNegatives and FalseNegatives as shown in the figure below.

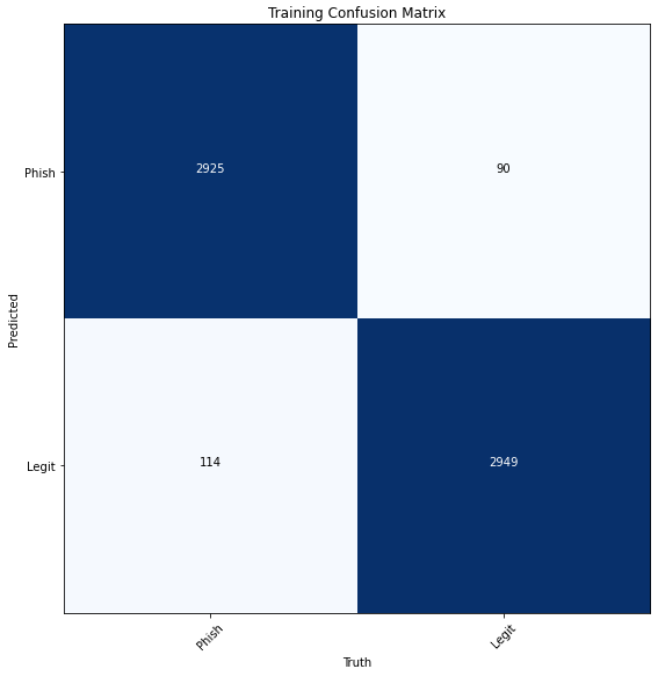

Figure 1. Confusion Matrix representing Phish and Legit URLs

In the case of the above mentioned experimentation, these values represent the following details:
true positives (TP): These are cases in which the model predicts a legitimate URL, and the URL is actually safe/legitimate.
true negatives (TN): model predicts an unsafe URL, and the URL is actually spam.
false positives (FP): the model predicts a legitimate URL, but the URL is actually a spam one. (Also known as a "Type I error.")
false negatives (FN): the model predicts a spam URL, but the URL is actually legitimate. (Also known as a "Type II error.")

Precision = TruePositives / (TruePositives + FalsePositives)
Recall = TruePositives / (TruePositives + FalseNegatives)
F1 score = (2 * Precision * Recall) / (Precision + Recall)
Accuracy = TruePositives+TrueNegatives/(TruePositives + FalsePositives+ TrueNegatives+ FalseNegatives)

All models were initiated with the checkpoint and number of labels as 2. The following table shows the results of the seven tested models with their inference and conclusions. The graphical representation of the results provides a comparison of the metrics for the seven tested models

Transformer	**Accuracy**	**Precision**	**Recall**	**F1 score**
Roberta-base	0.958	0.957	0.958	0.958
Albert-base-v2 Second Run(change start learning rate)	0.5 0.96	0.5 0.955	1 0.965	0.6 0.96
Albert-base-v1	0.99	0.96	0.96	0.96
Distilbert-base-uncased-finetuned-sst-2-english	0.97	0.98	0.96	0.97
distilbert-base-uncased	0.97	0.970	0.96	0.96
distilroberta-base	0.96	0.97	0.95	0.96
BERT-base-uncased	0.98	0.97	0.98	0.97

Table 1. Evaluation metrics of the tested models

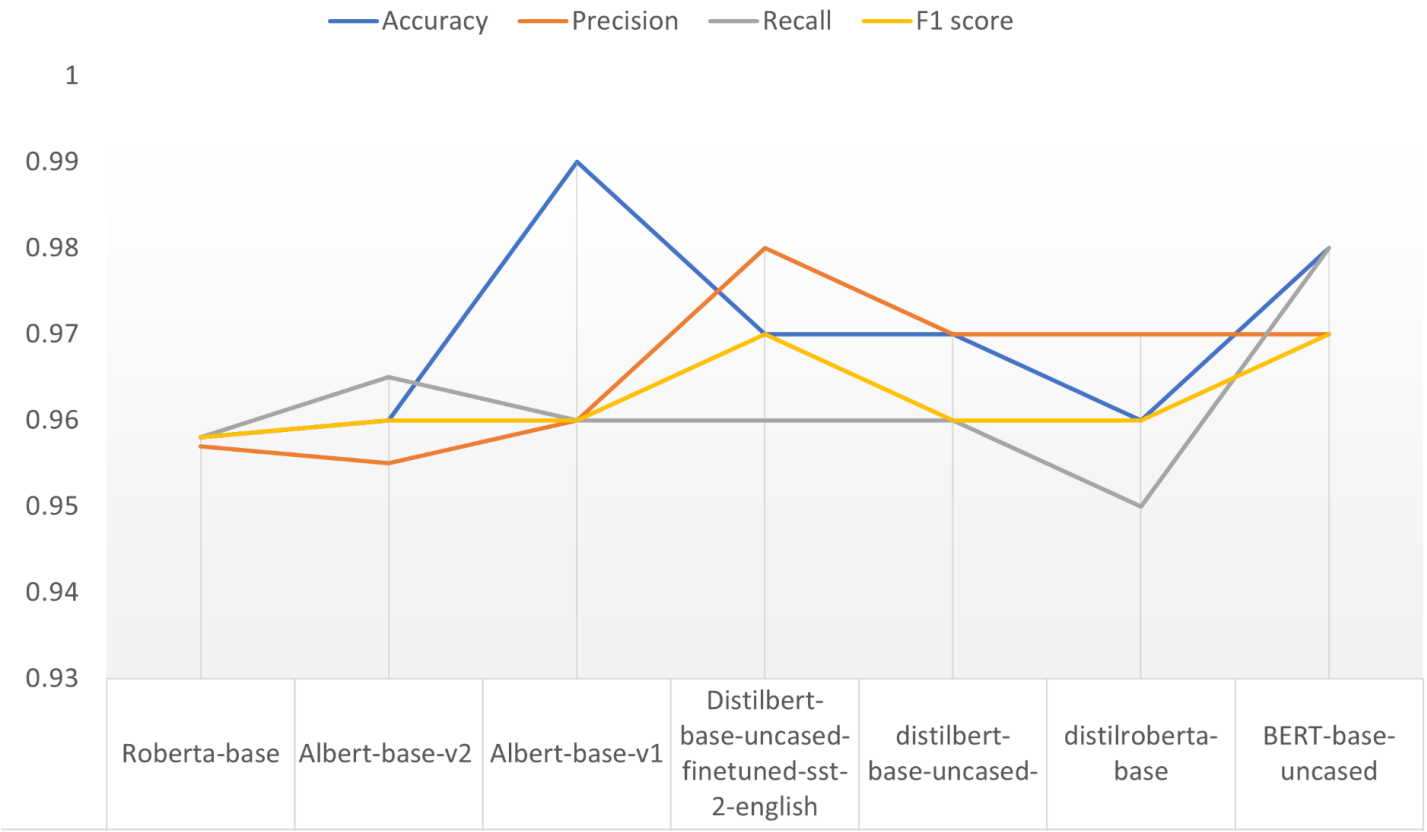

Figure 2. Graphical comparison of evaluation metrics of the models

On evaluating the model, it's seen both from Table 1 and Figure 2 that all models have an accuracy above 95%. The graphical representation shows a comparison of the performance metrics of the seven tested models. The detailed reasoning for the performance of all the models is listed below with its tested model parameters.

- **BERT -base uncased:**

  BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding is a bidirectional transformer pre-trained using a combination of masked language modeling objective and next sentence prediction on a large corpus comprising the Toronto Book Corpus and Wikipedia.

  Before the implementation of BERT, language models generally would have looked at the text sequence during training from left to right or combined right to left and left to right. For generating sequences this one directional approach works well. However, BERT is

bidirectionally trained architecture. That means this model will have a deeper sense of language context and flow compared to a unidirectional approach.

BERT does not predict the next word in a sequence; rather it makes use of a technique called Masked LM (MLM) which randomly masks words of a sentence and then tries to predict them. That masking helps the model to look in both directions and utilizes the full context of the sentence. BERT uses transformer architecture, basically the attention mechanism . Normally a transformer consists of an encoder for reading text input and a decoder to produce prediction tasks. BERT only requires the encoder part as it only generates language representation models [20].

In BERT-Base, the architecture consists of 12 layers, 768 hidden units, 12 attention heads with 110M parameters. .To get the high accuracy, the parameters were initialized to the below mentioned values:
batch_size = 32
warmup = 600
max_seq_length = 128
num_train_epochs = 3.0

- **RoBERTa-base:**

Though BERT outperformed the state of the art across a wide variety of tasks under general language understanding, there was still room for improvement. One thing was, the masking in BERT was done only once during data processing which results in a single static mask. As a result the same input masks were fed to the model on every single epoch. Another thing was the original format used in BERT is segment-pair+nsp (next sentence prediction) loss. Here, each input has a pair of segments which can contain multiple natural sentences but total combined length must be less than 512 tokens. Moreover, the original BERT was trained for 1M steps with a batch size of 256 sequence which shows room for improvement.

RoBERTa , released by Facebook, uses dynamic masking which allows the data to be repeated 10 times and every next time. The masked word would also be different within the same sentence. The Roberta implemented larger batch sizes which were suitable to parallelize via distributed systems and improved accuracy. The training data was also increased in Roberta to 160 GB [18].

So, for the next experimentation we considered the transformer RoBERTa which is pre-trained on a large corpus of English data in a self-supervised fashion. It is pre-trained with Masked language modeling which is different from traditional recurrent neural networks**.**

The major packages imported in addition were RobertaTokenizer and TFRobertaModel. On running the model and analyzing the results, it is seen that the Roberta base transformer produces excellent results with an accuracy of 96%. The efficiency of Roberta is due to the fact that it is trained in a dynamic way where masking is generated every time a sequence is fed to the model. This means that the hyperparameter tuning is incorporated automatically during the training.

To get the high accuracy, the parameters were initialized to the below mentioned values:
batch_size = 8
num_epochs = 3
initial_learning_rate=5e-5
end_learning_rate=0.0

- **ALBERT**

ALBERT is a lite BERT system which uses less memory space. It uses two parameter reduction techniques for using memory space in an efficient way. The first is a factorized embeddings parameterization. By decomposing the large vocabulary embedding matrix into two small matrices, the size of the hidden layers is separated from the size of vocabulary embedding. This makes it easier to grow the hidden size without significantly increasing the parameter size of the vocabulary embeddings. The second technique is cross-layer parameter sharing. This technique prevents the parameter from growing with the depth of the network[19].

Additionally, ALBERT utilizes a self-supervised loss for sentence-order prediction (SOP). SOP primary focuses on inter-sentence coherence and is designed to address the ineffectiveness of the next sentence prediction (NSP) loss proposed in the original BERT [16]. In addition, ALBERT also uses a pretraining loss based on predicting the ordering of two consecutive segments of text which is known as Sentence Ordering Prediction.
The experimentation looks into the performance of two versions, namely Albert-base-v1 and Albert-base-v2.

Albert-base-v1

The first version of Albert has the following configuration:

12 repeating layers

128 embedding dimension

768 hidden dimension

12 attention heads

11M parameters

Owing to the experimentation, the major packages imported in addition were AlbertTokenizer. On running the model and analyzing the results, its seen that Albert-base-v1 transformer produces excellent results with an accuracy of 99%.

To get the high accuracy, the parameters were initialized to the below mentioned values:

batch_size = 8

num_epochs = 3

initial_learning_rate=5e-5

end_learning_rate=0.0

Albert-base-v2

Version 2 is different from version 1 due to different dropout rates, additional training data, and longer training. The rest of the configurations remain the same.

Since Albert-base-v2 is similar however different from it's version 1 model. We first tried it because v2 is the latest version of Albert. We tuned hyperparameters with 2 epochs, polynomial decay steps with 48000 and initial learning rate with 1e-4, the training process showed that the loss value oscillates heavily and accuracy fluctuates just above 50%. Then we tuned polynomial decay steps to 72000 and initial learning rate to 5e-5. The training loss value decreases fast and finally results in a high accuracy value.

The hyperparameter tuning trials manifest that higher learning rate decay steps would reduce oscillation phenomena. However, hyperparameters are a significant factor of fine-tuning the transformer models. It is necessary to explore and try various parameters to get best performance results.

- **DistilBERT base uncased**

  DistilBERT is a small, fast, cheap and light Transformer model based on the BERT architecture. Knowledge distillation is performed during the pre-training phase to reduce the size of a BERT model by 40%. To leverage the inductive biases learned by larger models during pre-training, the authors introduce a triple loss combining language modeling, distillation and cosine-distance losses [17]

  DistilBERT base uncased model is a distilled version of the BERT base model. This transformer model is smaller and faster than BERT, and is pretrained on the same corpus in a self-supervised fashion, using the BERT base model as a teacher. In addition to Masked language modeling, it was also trained to generate Cosine embedding loss.

  Owing to the experimentation, the major packages imported in addition were DistilBertTokenizer, TFDistilBertModel. The experimentation yielded an excellent accuracy of 97% when the parameters were initialized to the below mentioned values:
  initial_learning_rate=5e-5
  end_learning_rate=0.0
  Batch Size : 128
  Num_epcohs : 2

- **DistilBERT base uncased fine tuned SST-2-english**

  DistilBERT base uncased fine tuned SST-2-english is a fine-tune checkpoint of DistilBERT-base-uncased.
  The following are the parameters that can be tuned:
  learning_rate = 1e-5
  batch_size = 32
  warmup = 600
  max_seq_length = 128
  num_train_epochs = 3.0

  The experimentation with DistilBERT base uncased fine tuned SST-2-english yielded an accuracy of 97% with the above mentioned parameters.

- **Distil Roberta-base**

  This model is a distilled version of the RoBERTa-base model and follows the same training procedure as DistilBERT. This model has a total of 82M parameters, consisting of the following features:

  6 layers
  768 dimension
  12 heads

  Compared to lesser parameters than the RoBERTa-base model, DistilRoBERTa is twice as fast as Roberta-base which has a total of 125M parameters.
  The experimentation with Distil Roberta base produced an accuracy of 96% with the following parameters:
  initial_learning_rate=0.00005
  end_learning_rate=0.0
  Batch Size : 128
  Num_epcohs : 2

7. CONCLUSION

In this study, we explored different transformer model's performance on URL legitimate-malicious dataset, we proved that the current widely used models such as BERT, RoBERTa, ALBERT, DistillBERT, including their sub-versions could serve as effective method to predict phishing URLs. Among these models, two versions of ALBERT take a small amount of disk memory space that are less than 60 megabytes, thus becoming ideal models that can be applied on lightweight devices that reduce memory cost without affecting the predicted accuracy.

The average prediction time for one URL sample is less than 3s, which means it only takes a very short time to discover a phishing URL. We utilize the polynomial decay learning rate to accelerate the initial learning speed, thus it is easy to discover whether a hyperparameter fits a certain model or not in a relatively short time. This study also explored hyperparameters that best fit each model. We discovered that even when applying the above models without preprocessing the URLs, the result is good enough to conduct prediction, thus proving that the NLP transformer model is a fast and easy approach to training raw URLs.

8. FUTURE WORK

The future work of this paper will include testing the models on different URL datasets to see if the results are the same. In this study, the URL dataset was based on the generated dataset which is not based on the most recent real world URLs. Thus the training samples didn't include world wide used websites such as Google, Bing, Baidu, etc. Although the result is promising, it is necessary to conduct extra experiments on an actual large URL dataset.

In addition, the experiments can be extended to the breadth requirements of being compatible with Pytorch, which now is only compatible with Tensorflow. There are various transformer models that possibly could be used to predict phishing URLs and many of them are written only to be used along with Pytorch. In order to make this study more comprehensive, future work will include more tasks on Encoder based models, Decoder based models and several sequence models to see if architecture of transformer models affect URL category classification.

REFERENCES

[1] Ebubekir Buber, Banu Dırı, and Ozgur Koray Sahingoz. 2017. Detecting phishing attacks from URL by using NLP techniques. In 2017 International conference on computer science and Engineering (UBMK). IEEE, 337–342.

[2] Ye Cao, Weili Han, and Yueran Le. 2008. Anti-phishing based on automated individual white-list. In Proceedings of the 4th ACM workshop on Digital identity management. 51–60.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).

[4] Gal Egozi and Rakesh Verma. 2018. Phishing email detection using robust nlp techniques. In 2018 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE, 7–12.

[5] Katherine Haynes, Hossein Shirazi, and Indrakshi Ray. 2021. Lightweight URL-based phishing detection using natural language processing transformers for mobile devices. Procedia Computer Science 191 (2021), 127–134.

[6] JungMin Kang and DoHoon Lee. 2007. Advanced white list approach for preventing access to phishing sites. In 2007 International Conference on Convergence Information Technology (ICCIT 2007). IEEE, 491–496.

[7] Tie Li, Gang Kou, and Yi Peng. 2020. Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods. Information Systems 91 (2020), 101494.

[8] Luong Anh Tuan Nguyen, Ba Lam To, Huu Khuong Nguyen, and Minh Hoang Nguyen. 2013. Detecting phishing web sites: A heuristic URL-based approach. In 2013 International Conference on Advanced Technologies for Communications (ATC 2013). IEEE, 597–602.

[9]ChristianSeifert,IanWelch,andPeterKomisarczuk.2008.Identificationofmaliciouswebpageswith staticheuristics.In2008Australasian Telecommunication Networks and Applications Conference. IEEE, 91–96.

[10] Hossein Shirazi, Shashika R Muramudalige, Indrakshi Ray, and Anura P Jayasumana. 2020. Improved phishing detection algorithms using adversarial autoencoder synthesized data. In 2020 IEEE 45th Conference on Local Computer Networks (LCN). IEEE, 24–32.

[11] Muhammad Taseer Suleman and Shahid Mahmood Awan. 2019. Optimization of URL-based phishing websites detection through genetic algorithms. Automatic Control and Computer Sciences 53, 4 (2019), 333–341.

[12] Choon Lin Tan. 2018. Phishing dataset for machine learning: Feature evaluation. Mendeley Data 1 (2018), 2018.

[13] Bo Wei, Rebeen Ali Hamad, Longzhi Yang, Xuan He, Hao Wang, Bin Gao, and Wai Lok Woo. 2019. A deep-learning-driven light-weight

[14] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019).

[15] Victor Zeng, Shahryar Baki, Ayman El Aassal, Rakesh Verma, Luis Felipe Teixeira De Moraes, and Avisha Das. 2020. Diverse datasets and a customizable benchmarking framework for phishing. In Proceedings of the Sixth International Workshop on Security and Privacy Analytics. 35–41.

[16] Lan, Zhenzhong, et al. "Albert: A lite bert for self-supervised learning of language representations." arXiv preprint arXiv:1909.11942 (2019).

[17] Sanh, Victor, et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." arXiv preprint arXiv:1910.01108 (2019).

[18] Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692 (2019).

[19] Lan, Zhenzhong, et al. "Albert: A lite bert for self-supervised learning of language representations." arXiv preprint arXiv:1909.11942 (2019).

[20] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).