# Advancing Phishing Attack Detection with A Novel Dataset and Deep Learning Solution

Quoc-Khanh Le[1,2], Quoc-An Nguyen[1,2], Dat-Thinh Nguyen[1,2], Xuan-Ha Nguyen[1,2], and Kim-Hung Le[1,2]

[1] University of Information Technology, Ho Chi Minh City, Vietnam
[2] Vietnam National University, Ho Chi Minh City, Vietnam
`{21520978, 21521809, 19520982}@gm.uit.edu.vn, {hanx, hunglk}@uit.edu.vn`

**Abstract.** Phishing attacks, increasingly complex and accessible due to low cost and technical requirements, demand advanced detection methods. While recent machine learning-based approaches show promising results in preventing these threats, they still face limitations in terms of outdated training datasets and the number of extracted features. Therefore, in this paper, we introduce a novel phishing attack dataset with a high number of samples and dimensionality. We also propose a transformer-based deep learning model to detect phishing attacks accurately. Our experimental results on our dataset show a significant performance gain, achieving 98.13% accuracy, surpassing popular machine learning models and SAINT, a state-of-the-art deep learning model for tabular data.

**Keywords:** Phishing detection · deep learning · Transformer architecture

## 1 Introduction

Phishing attacks stand out as a popular cyber threat because of their minimal cost and technical requirements. A phishing attack is typically initiated with a spam email that contains malicious URLs leading to a phishing website. This website is developed to mimic the user interface of well-known and widely trusted websites [1], where users are likely to provide their personal information, especially ID cards and credit cards. Due to the high similarity, it is challenging for users to recognize the differences, making them vulnerable to these phishing attacks. In response to these attacks, a defense measure must be developed to prevent this threat and protect the users.

One of the most popular phishing detection approaches is signature-based systems. A signature-based detection system often consists of a blacklist of denied URLs, in which the URL of an incoming web request is compared with all items in the list. If an exact match is found, the incoming URL is marked as malicious, and the request is dropped, or an alert is initiated for the user. Although this method delivers a low false positive rate and is easy to deploy, it has some limitations. Firstly, it is challenging to maintain an extensive database of denied

URLs as the larger the database, the more storage it occupies and the more time it takes to process a single URL. Secondly, signature-based detection systems are renowned for their inability to detect novel attacks (i.e., unseen phishing URLs). Particularly, attackers could make a minor change to their malicious URLs, resulting in URLs not existing in the blacklist and bypassing the detector easily. Unfortunately, it is challenging to update the blacklist promptly with respect to new phishing attacks since it takes time and requires expert knowledge to analyze and construct new rules.

To overcome these limitations, researchers have geared towards anomaly-based approaches, which are well-known for their ability to identify unseen phishing attacks while not requiring an extensive database of malicious URLs. In the training mode, a deep learning model embedded in the anomaly-based detection system is trained on a fully annotated phishing attack dataset, including features of phishing attacks. In the operation mode (or inference mode), the model is fed with incoming web requests and then produces the probability of these requests being malicious. Although several works [2] [3] [4] have proven this approach effective, several challenges remain. Firstly, current datasets contain outdated data, which degrades the ability to detect recent phishing attacks. Also, we observe that these datasets were constructed with a limited number of features and samples, which are insufficient to train deep learning models. One typical example is the UCI data repository, which was last updated in 2016 [5]. Notably, the Website Phishing dataset [5] in the UCI repository has only 1350 samples and 10 features, while the Phishing Websites dataset [6] only has 30 features and 2460 samples. Secondly, recent works have not extensively explored the capability of tabular deep learning models (e.g., TabTransformer [7]) in phishing attack detection despite the potential of these models on tabular data.

Recognizing the limitations of current anomaly-based detection systems, we propose our self-collected dataset with extensive features and samples and a tabular deep-learning model for detecting phishing attacks. In more detail, the dataset was constructed from state-of-the-art sources of phishing attacks, such as OpenPhish [8], PhishTank [9], PhishStats [10], and Alexa [11], all of which include active phishing attacks in 2023. The data collection and feature extraction pipeline are also detailed in this paper. In addition, we propose a TabTransformer-based deep-learning model to detect phishing attacks and evaluate the performance of the proposed model on our self-collected dataset. Finally, we compare its performance with popular machine learning and deep learning models, such as Linear Model, KNN, SVM, Decision Tree, Random Forest, and SAINT.

In summary, our contributions are:

- We propose a new phishing attack dataset with a large number of samples and a high dimensionality. We also provide the pipeline to build the dataset from reputable sources such as OpenPhish [8], PhishTank [9], PhishStats [10], and Alexa [11].

- We propose a Tabtransformer-based model to detect phishing attacks, which is specifically designed to suit tabular data and deliver a high detection accuracy.
- We conduct experiments on our dataset and compare the performance of our proposed model with popular baseline models such as KNN, Decision Tree, SVM, and SAINT [12].

The remainder of the paper is organized as follows: Section 2 provides a literature review of anomaly-based phishing attack detection methods. Section 3 gives details of our dataset and the proposed model. Section 4 presents the evaluation results. Finally, we draw our conclusion in Section 5.

## 2 Related works

Several datasets have been proposed for training and evaluating anomaly-based phishing detection systems. One popular dataset is ISCX-URL-2016 [13], which consists of 79 features and 114,400 samples categorized into five categories: Benign, Spam, Malware, Phishing, and Defacement. Notably, this dataset is a collection of around 10,000 phishing samples from Openphish and over 35,300 benign samples from Alexa. Chiew et al., [14] proposed a feature extraction framework, Hybrid Ensemble Feature Selection, which extracted 48 features from multiple data sources (e.g., OpenPhish, Alexa, PhishTank), resulting in a dataset of 10,000 samples categorized into two classes (phishing and legitimate).

Several literature reviews [15,16] have shown that many AI-enabled methods, such as machine learning, deep learning, and hybrid learning, have been effectively proposed in phishing attack detection methods. For instance, [17] proposed using SVM, MLP, Naive Bayes, and Decision Tree to detect phishing attempts based on URLs. These models were trained on a dataset combining Alexa, Openphish, and PhishTank, and achieved the best accuracy with MLP, at 90%. Likewise, [18] proposed combining data from Alexa, UCI, and PhishTank to train K-NN, SVM, Decision Tree, Random Forest, Rotation Forest (RoF), and MLP. Their experimental result demonstrated a high accuracy ranging from 91.86% to 96.65%. Another work [19] used Information Gain and Shapley Additive Explanation (SHAP) to select salient features from their combined dataset. This dataset was then used to train Naive Bayes, XGBoost, and Random Forest, achieving the highest accuracy of 98.561%.

Regarding deep learning models, [20] used RNN, DNN, and CNN to detect phishing attacks. They experimented on a combined dataset of Phishtank, Openphish, Alexa, Common-Crawl [21], and Yandex [22], delivering accuracy ranging from 95.22% to 98.58%. Another work at [23] also trained DNN and BiLSTM on datasets from Ebbu2017 [24] and Phishtank, showing an accuracy of 98.79% and an F1-score of 98.81%. A recent study [25] applied an autoencoders-DNN model on the ISCX-URL-2016 [13] dataset and achieved an accuracy of 97.45%, an F1-score of 97.54%, a precision of 97.89% and a recall of 97.20%.

## 3 Proposed method

### 3.1 Overview

In this section, we are going to introduce our phishing-detecting model, from the overview to the detailed functionalities. The overview of our system is illustrated by Figure 1, including four components: input, features processing, phishing detecting, and output.

**Input**: The system receives suspicious URLs collected from internet users.

**Features processing component**: Responsible for extracting URL features based on their structure, syntax, and content, in addition to querying external services. After the progress, the features are formed as a vector and then fed into the detection model for detecting phishing URLs.

**Phishing detecting component**: Tasked with detecting phishing URLs by using the TabTransformer model. Initially, the model checks the structure and the syntax of the URL to detect anomalies. After that, the HTML content is checked by the model to find malicious scripts considered to contain phishing elements. Finally, the query of the external services is analyzed and concluded by the proposed model.

**Output**: Following the analyzing process, the model determines whether the URL is legitimate or phishing and sends the results to the administrator for further action.

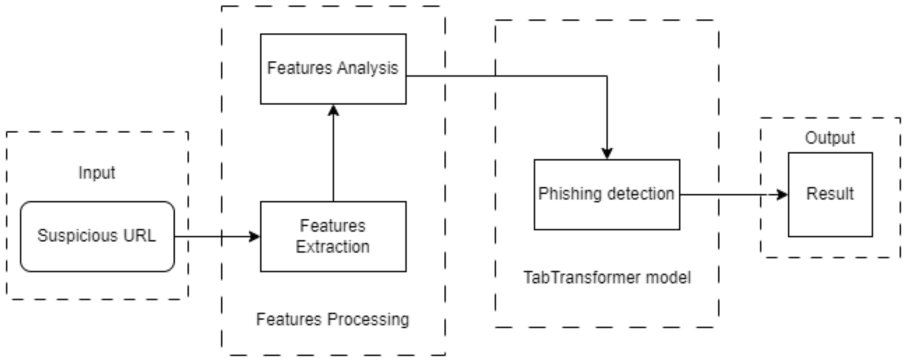

Fig. 1: The overview of our proposed system.

### 3.2 Dataset

Datasets play an important role in training machine learning models. However, within the domain of phishing attack detection, there is a lack of standard datasets for training models. This is due to the fact that phishing sites are short-lived and dead URLs cannot be used for content analysis. Additionally, most available datasets only contain the values of the tested features without information about the URLs used. To address this challenge, our research collects raw resources from various reputable sources, similar to previous studies, and applies our proposed feature extraction to gain valuable information. The popular sources to collect raw phishing websites are listed below:

- OpenPhish is a comprehensive platform that provides a database of 18 million phishing URLs with metadata such as page content and IP addresses. This platform provides valuable insights for understanding and analyzing phishing threats. [8]
- PhishTank is a community-driven platform where anyone can freely submit, verify, track, and share phishing data. It provides accurate, useful information for people to identify malicious online activities. [9]
- PhishStats offers real-time phishing monitoring from various sources, including VirusTotal and Google Safe Search. [10]
- Alexa, hosted by Amazon, provides various analytics tools to analyze and monitor website performance. One of its popular tools is SiteRank, which ranks millions of benign website URLs based on their probability. [11]

Figure 2 depicts our process, including two steps G1 and G2, to collect raw sources for training our detection modes.

**G1**: Start by collecting URLs from different sources. Detailed analysis shows that many studies have focused on gathering information from the top-ranked websites according to Alexa statistics. However, Alexa only recommends top-ranked domains without mentioning subdomains or paths. Therefore, to ensure uniformity of the dataset, one cannot directly use Alexa listings, especially when features related to subdomains and paths are applied. To solve this problem, the study proposes a new method, using the list of top domains that Alexa provides as a seed. This seed is then used to expand the crawl from more general URLs, including subdomains and detailed paths.

**G2**: Crawled URLs need to be processed first. The preprocessing step includes:

- Removing duplicate URLs.
- avoiding using multiple URLs from the same domain because those URLs may have similar feature values.

### 3.3 Features Extraction

We focus on combining lexical features (Lexical Features), content features (Content Features) and host features (Host Features) to create a powerful and accurate recognition model [26]. We have integrated 82 features from the above three feature types to build a system capable of effectively evaluating and identifying phishing URLs. This helps us tackle the increasingly sophisticated tactics cybercriminals use to defraud users :

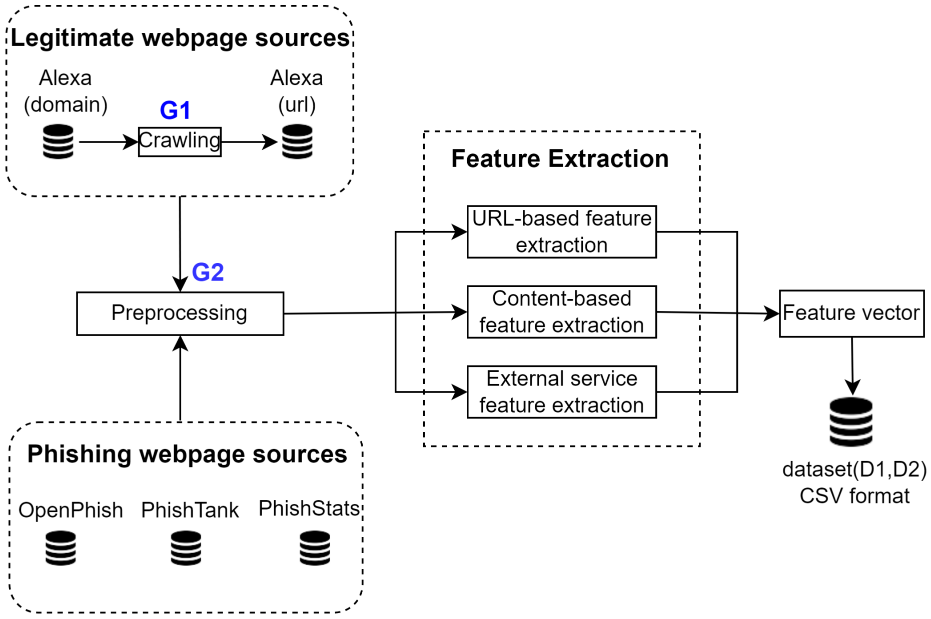
Legitimate webpage sources
Alexa (domain)
G1
Crawling
Alexa (url)
G2
Preprocessing
Phishing webpage sources
OpenPhish
PhishTank
PhishStats
Feature Extraction
URL-based feature extraction
Content-based feature extraction
External service feature extraction
Feature vector
dataset(D1,D2) CSV format

Fig. 2: Dataset construction process.

**Lexical Features:** URLs are often considered strings of characters that don't have much meaning, but they still contain valuable information about the content, purpose, and even potential risks. Lexical features rely on the fact that malicious URLs have a different structure than legitimate URLs. We extracted the lexical characteristics of the URL, including:

– Determine the length of the URL and its components such as domain name and path.
– Check for the presence of an IP address in the URL.
– Count the number of marks such as dots and dashes in the URL.
– Check for the presence of strings like ”www” or ”com” in the URL.
– Count the number of tokens like ”http” in the URL and check for the presence of ”https” in the scheme.
– Count the number of digits in the URL and calculate the ratio of digits in the hostname part.
– Check for signs of a URL that could be a phishing or fraudulent site.
– Check to see if the domain name contains any trademarks.
– Generate statistical reports on URL characteristics and associated domains.

**Content Features:** Information about a website's HTML content aids in the process of determining whether a website is a phishing site or not. We extracted various characteristics from the website's HTML content including:

– Nature of links (internal, external).

**Algorithm 1:** Extract features algorithm

```
Input: URL of suspicious website
Output: 82 features of URL
Function LexicalFeatures($url$):
    Step 1: Lexical parsing of URLs in paths, domains,...
    Step 2: Extract 55 lexical features of the URL.
    return List 55 features

Function LexicalFeatures($url$):
    Step 1: Parse HTML content from URL.
    Step 2: Extract 19 features from HTML content.
    return List 19 features

Function ContentFeatures($url$):
    Step 1: Get URL server information from external services.
    Step 2: Extract 8 features of server information.
    return List 8 features

Function Main($LexicalFeatures$, $ContentFeatures$,$HostFeatures$):
    Combine features of Lexical, Content, Host .
    return List 82 features

```

- CSS files and multimedia files.
- Login forms and features.
- Internal and external errors.
- Components like iframes, popup windows, and secure links.
- Title and domain information.

**Host Features:** Calculate and check the characteristics of the server or domain to evaluate the reliability of the website. We extract the URL's host features including:

- Determine whether the domain name is registered or not.
- Information about the registration period of the domain name.
- Determine the age of the domain name.
- Extract domain DNS records.
- Check if the URL has been indexed by Google.
- Evaluate the page rank of the URL.
- Information about domain name expiration.

### 3.4 Detection model

Following the features extraction phase, the dataset is structured as a tabular of vector, where each vector represents a set of features extracted from a URL.

Due to the heterogeneity problem of a tabular dataset, some other deep learning architectures, such as convolutional neural networks or long short-term memory, may not be optimal for processing tabular data. Therefore, we design our detection model based on TabTransformer architecture [7], a novel architecture proposed for handling tabular data. Its unique designs enable TabTransformer to process both categorical and continuous features; hence, it is more robust in processing tabular data than other architectures. Figure 3 depicted our model design, which consists of an Embedding part, followed by a Transformer part and an MLP part for final classification.

**Embedding feature:** The Feature Embedding module is responsible for converting input feature values into learnable embeddings. Specifically, it takes as input categorical features $(x_1, x_2, x_3...x_m)$ separately. Categorical features are embedded in various ways, explained in detail as Equation 1, where $Embed()$ and $b_i^{cat}$ are the categorical embedding function and bias.

$$E_i^{cat} = Embed(x_i^{cat}) + b_i^{cat} \tag{1}$$

**Transformer blocks.** The output of Feature Embedding, $E^{cat}$, is fed into Transformer encoders, which is a stack of N transformer blocks. Each block is constructed from a multi-head attention layer and a fully connected feed-forward network, each of which is followed by the residual link and normalization layer. In particular, our design model adopted in this work has N = 4, two heads of multi-head attention layers, and four embedding dimensions in each block, with a 0.4 post-attention dropout ratio and a 0.4 feed-forward dropout ratio. Equation 2 provides a detailed representation of the calculation for the attention level $Attention(\cdot)$. Based on the input embedding $E_{head_i}^{cat}$, three learned vectors are computed: query vector $(Q)$, key vector $(K)$, and value vector $(V)$. The weighted sum of the $V$ vectors is then obtained by computing the dot product of the $Q$ vector with the $K$ vector and subsequently applying the softmax function to the result to get the weights.

$$A(E_{head_i}^{cat}) = softmax\left(\frac{Q(E_{head_i}^{cat})K^T(E_{head_i}^{cat})}{\sqrt{d}}\right)V(E_{head_i}^{cat}) \tag{2}$$

After being processed by transformer block, the output vector $E^{trans\_out}$ is concatenated with the output vector $E^{continuos\_out}$ and then fed into the MLP block.

$$E^{out} = Concat([E^{trans\_out}, E^{continuos\_out}]) \tag{3}$$

**MLP Block:** A Multi-Layer Perceptron (MLP) block is a basic building block used in deep learning architectures for classification tasks. In this work, we design an MLP block with four layers to classify vector $E^{out}$ received from the transformer block into two labels: legitimate or Phishing. The $E^{out}$ is initially normalized to reduce internal covariate shift, help the model to stabilize, learn faster, and be less sensitive to the scale of input features. The Dense layer performs a transformation by using $Relu$, an activation function that applies non-linearity to the model. Following this, we apply a Dropout layer to prevent the model from being overfitting. Finally, we use the Sigmoid layer to produce final binary predictions.

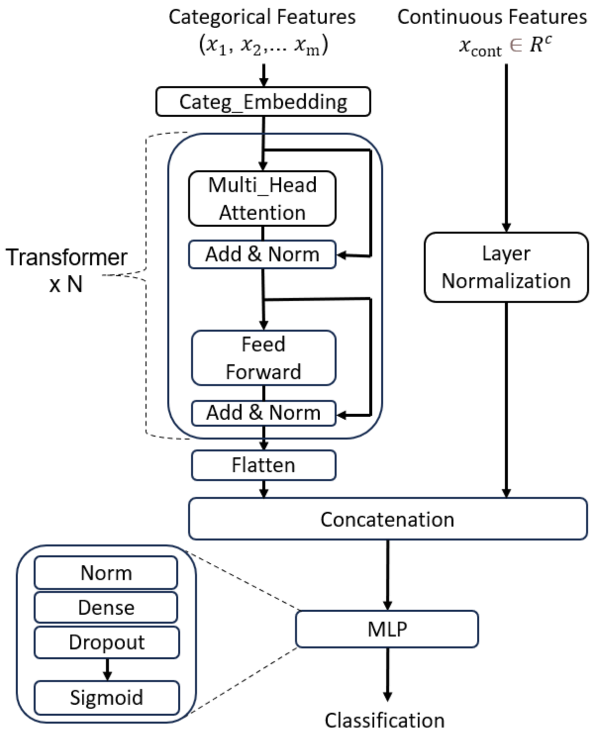

Fig. 3: The architecture of proposed model.

## 4 Experimental results

### 4.1 Metrics

The metrics used to evaluate the model's efficiency include: Accuracy, F1-score, Precision, Recall, and AUC

– Accuracy: The metric used to compute the ratio of Correctly-predicted samples to the total number of samples. The accuracy of a model is measured using the following formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision: The metric shows the number of URLs detected as phishing in the dataset. The precision of a model is measured using the following formula:

$$Precision = \frac{TP}{TP + FP}$$

- Recall: illustrates the total number of URLs classified as legitimate or phishing. The recall of a model is measured using the following formula:

$$Recall = \frac{TP}{TP + FN}$$

- F1 score: The metric indicates the harmonic mean of Precision and Recall. The F1 score is measured using the following formula:

$$F1 - score = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

- AUC: the area under the ROC curve.

## 4.2 Experiment results

Figure 4 and Figure 5 present the confusion matrix of our model on datasets D1 and D2, respectively. The results indicate the proposed model has high accuracy, with most of the phishing samples correctly predicted. Furthermore, our proposed model limits the false positive rates to very low (24 incorrect legitimate samples in D1 and 42 corresponding in D2), helping reduce false alarms to administrators.

Table 1 compares evaluation metrics among models in dataset D1. The proposed model has the greatest AUC index (99.70%), indicating superior classifier performance. Additionally, the model has high precision and recall indexes (97.96% and 95.60%, respectively), indicating its efficiency and accuracy in identifying possibilities. Moreover, the accuracy index is higher than that of the other models, at 97.64%.

Table 1: Comparison between proposed model and other models when executing dataset D1.

| MODEL | AUC | RECALL | PRECISION | F1-SCORE | ACCURACY |
|---|---|---|---|---|---|
| Linear Model | 97.52 | 92.58 | 92.56 | 92.56 | 92.58 |
| KNN | 97.24 | 91.37 | 91.51 | 91.23 | 91.37 |
| SVM | 99.08 | 96.33 | 96.33 | 96.33 | 96.33 |
| Decision Tree | 96.95 | 93.28 | 93.27 | 93.26 | 93.28 |
| Random Forest | 99.13 | 96.05 | 96.04 | 96.05 | 96.05 |
| SAINT | 99.26 | **96.50** | 96.51 | 96.50 | 96.50 |
| Proposed model | **99.70** | 95.60 | **97.96** | **96.77** | **97.64** |

Table 2 shows the comparison of evaluating indices between models in dataset D2. The AUC index of the proposed model is the highest (99.71%), indicating the superiority of the model's classifier. Besides, the proposed model's precision and recall index are high, 98.04% and 98.55% respectively, which proves the efficiency and the accuracy of the model's identifying possibility. Moreover, the accuracy index is higher compared to the remaining (98.13%).

Table 2: Comparison between proposed model and other models when executing dataset D2.

| MODEL | AUC | RECALL | PRECISION | F1-SCORE | ACCURACY |
|---|---|---|---|---|---|
| Linear Model | 98.71 | 95.98 | 95.97 | 95.97 | 95.98 |
| KNN | 98.66 | 94.87 | 94.86 | 94.86 | 94.87 |
| SVM | 97.88 | 0.89.72 | 90.88 | 89.80 | 89.72 |
| Decision Tree | 98.84 | 96.20 | 96.21 | 96.20 | 96.20 |
| Random Forest | 99.34 | 96.15 | 96.15 | 96.15 | 96.15 |
| SAINT | 99.36 | 96.07 | 96.61 | 96.60 | 96.42 |
| Proposed model | **99.71** | **98.55** | **98.04** | **98.30** | **98.13** |

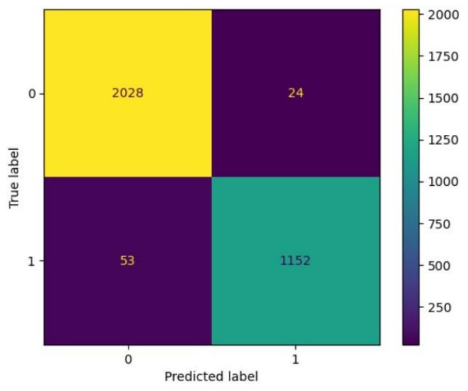

Fig. 4: The confusion matrix of our proposed model on dataset D1.

## 5 Conclusion

Phishing attacks are one of the most common online threats nowadays, posing a substantial threat to internet users. To migrate this attack effect, we present an advanced solution aimed at protecting internet users from phishing websites. To

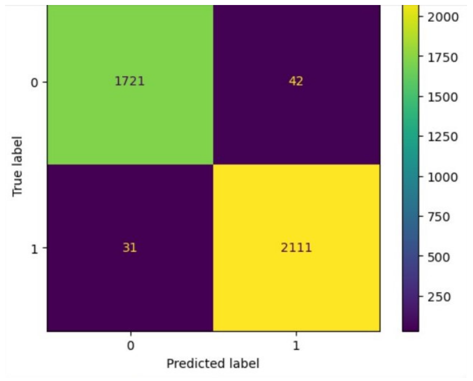

Fig. 5: The confusion matrix of our proposed model on dataset D2.

begin with, we introduce a comprehensive feature extraction algorithm to provide valuable information and provide a new dataset with over 70,000 samples to train detection models. Furthermore, we proposed a cutting-edge phishing detection model powered by a novel deep learning-based architecture - TabTransformer. The proposed model was evaluated with the new datasets and proved its high performance when outperforming other competitors. Finally, our findings and current implementation of the TabTransformer model also provide a basis for further research and even practical application implementation in the field of phished URL prediction.

## References

1. "Top 50 most impersonated brands in phishing attacks and new tools you can use to protect your employees from them." `https://cloudflare.com`.
2. O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from urls," *Expert Systems with Applications*, vol. 117, pp. 345–357, 2019.
3. P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning," *IEEE access*, vol. 7, pp. 15196–15209, 2019.
4. S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, pp. 60–69, 2007.
5. N. Abdelhamid, "Website Phishing." UCI Machine Learning Repository, 2016. DOI: https://doi.org/10.24432/C5B301.
6. R. Mohammad and L. McCluskey, "Phishing Websites." UCI Machine Learning Repository, 2015. DOI: https://doi.org/10.24432/C51W2X.
7. X. Huang, A. Khetan, M. Cvitkovic, and Z. Karnin, "Tabtransformer: Tabular data modeling using contextual embeddings," *arXiv preprint arXiv:2012.06678*, 2020.
8. "Openphish," 2023. `https://openphish.com/`.
9. OpenDNS, "Phishtank," 2023. `http://www.phishtank.com`.
10. "Phishstats," 2023. `https://phishstats.info/`.

11. "Alexa." `https://qse.ifs.tuwien.ac.at/ci/material/pub/ecsa19/documents/survey_alexa_top500.pdf`.
12. G. Somepalli, M. Goldblum, A. Schwarzschild, C. B. Bruss, and T. Goldstein, "Saint: Improved neural networks for tabular data via row attention and contrastive pre-training," *arXiv preprint arXiv:2106.01342*, 2021.
13. M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, "Detecting malicious urls using lexical analysis," in *International Conference on Network and System Security*, 2016.
14. K. L. Chiew, C. L. Tan, K. Wong, K. S. Yong, and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning-based phishing detection system," *Information Sciences*, vol. 484, pp. 153–166, 2019.
15. A. Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil, and K. Kifayat, "A comprehensive survey of ai-enabled phishing attacks detection techniques," *Telecommunication Systems*, vol. 76, pp. 139–154, 2021.
16. L. Tang and Q. H. Mahmoud, "A survey of machine learning-based solutions for phishing website detection," *Machine Learning and Knowledge Extraction*, vol. 3, no. 3, pp. 672–694, 2021.
17. K. Rendall, A. Nisioti, and A. Mylonas, "Towards a multi-layered phishing detection," *Sensors*, vol. 20, no. 16, p. 4540, 2020.
18. S. Al-Ahmadi, "Pdmlp: phishing detection using multilayer perceptron," *International Journal of Network Security & Its Applications (IJNSA) Vol*, vol. 12, 2020.
19. L. M. Rani, C. F. M. Foozy, and S. N. B. Mustafa, "Feature selection to enhance phishing website detection based on url using machine learning techniques," *Journal of Soft Computing and Data Mining*, 2023.
20. A. Aljofey, Q. Jiang, Q. Qu, M. Huang, and J.-P. Niyigena, "An effective phishing detection model based on character level convolutional neural network from url," *Electronics*, 2020.
21. "Commoncrawl," 2023. `https://commoncrawl.org/`.
22. "Yandex," 2023. `https://yandex.com/dev/xml/`.
23. A. Ozcan, C. Catal, E. Donmez, and B. Senturk, "A hybrid dnn–lstm model for detecting phishing urls," *Neural Computing & Applications*, vol. 35, pp. 4957 – 4973, 2021.
24. "Ebbu dataset." `https://github.com/ebubekirbbr/pdd/tree/master/input`.
25. M. K. Prabakaran, P. Meenakshi Sundaram, and A. D. Chandrasekar, "An enhanced deep learning-based phishing detection mechanism to effectively identify malicious urls using variational autoencoders," *IET Information Security*, vol. 17, no. 3, pp. 423–440, 2023.
26. G. Robots, "Extracting feature vectors from url strings for malicious url detection," 2021. `https://towardsdatascience.com`.