Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 212 (2022) 74–82

Procedia Computer Science

www.elsevier.com/locate/procedia

11th International Young Scientist Conference on Computational Science

Detection of Phishing URLs Using Temporal Convolutional Network

Mohamed Abdelkarim Remmide[a,*], Fatima Boumahdi[a], Narhimene Boustia[a], Chalabia Lilia Feknous[a], Romaissa Della[a]

[a] *Université BLIDA1, Laboratoire LRDSI, Faculté Sciences, B.P 270, Route de Soumaa, BLIDA, ALGERIE.*

Abstract

Throughout the past few years, phishing attacks have become an increasingly substantial problem for individuals and organizations. In this non-technical attack, the victim is deceived into accessing a malicious URL that downloads a malicious program to access the network or redirects the victim to a page that requests sensitive information. The literature is filled with many research proposals to mitigate this problem however, the dynamic nature and the creativity of the attacker have made it difficult and reoccurring. In this paper, we proposed to use novel deep learning techniques, namely the Temporal convolutional network (TCN) with word embedding, to detect phishing URLs. As a result of our experiments, we found that our model can detect phishing URLs with 98.95% accuracy, 98% precision, 98% recall, and 98% f1-score. This result indicates that our model is effective against phishing attacks.

Keywords: Social engineering; Phishing attack; Phishing URL detection; URL classification; Temporal Convolutional Network .

1. Introduction

Phishing is a fraud that involves using social engineering and/or technical subterfuge to steal consumers' personal information and financial account credentials[1]. Social engineering is used to lure the victim into clicking the link and eventually giving his sensitive information. While technical subterfuge, the cybercriminal plants malware onto the victim's computer so that he can steal all his sensitive information.

In recent years, phishing attacks have become a significant issue for individuals and organizations because they require no technical knowledge and are easy to perform. This makes them the most commune type of attack. A report released by the Anti-Phishing Working Group (APWG)[1] an international organization for eliminating phishing and related accidents found that phishing attacks doubled from early 2020 to the third quarter of 2021. At the same time,

* Corresponding author.
E-mail address: abdelkarimremmide@gmail.com

10.1016/j.procs.2022.10.209

software-as-a-service and webmail are the most frequently attacked sectors. On the other hand, financial institutions and payment providers remained extensively targeted with numerous attacks and should not be ignored.

Since phishing attacks significantly impact businesses, researchers have taken a closer look at how to detect them. Several approaches to detect phishing attacks have been proposed in the literature. These proposals can be categorized into list-based approaches [2, 3] where the known phishing URLs are stored in a list and are tested against the list whenever a new URL arrives. Due to the difficulties of maintaining phishing URL lists, a heuristic-based approach [4, 5] is proposed to generate the characteristic of phishing websites. In order to add intelligence and learnability to the solution, a machine learning-based approach [6, 7] is used to perform a classification of the URLs.However, the extraction of the features requires expertise. Finally the deep learning approach is used to overcome the last limitation, such as the work of Wei et al.[8], Huang et al.[9] and Le et al.[10]. However, the attacker's dynamic nature and creativity have made the detection of phishing attacks difficult and reoccurring problem.

In this paper, we focus on detecting phishing URLs using a deep learning approach. We use a Temporal convolutional network (TCN)[11] model. This novel deep learning model shows great potential in NLP tasks because it can extract spatial and temporal features using only one model rather than a combination of CNN and RNN. After the experiments, the result proves our hypothesis when we achieved 98,95% accuracy and 98% precision using the TCN model with word2Vec embedding. Which is better than the similar work of Liang et al. [12, 13]. The difference between our works is that we used word2vec embedding to represent the URL, followed by two TCN layers to perform classification, whereas they used their proposed URL representation and a single TCN layer.

The remainder of the paper is divided into the following sections: In Section 2, we begin by presenting relevant related work. We present a temporal convolutional network in Section 3. The purpose of Section 4 is to present the proposed architecture and experimental datasets. Section 5 details the experimental setup and the findings. Finally, we give some future orientations and conclude in section 6.

2. Related work

Due to the persistence of the phishing attack detection problem and the ease with which it can be implemented, researchers have proposed distinct solutions to the problem, which can be grouped into list-based, heuristic-based, machine learning-based, and deep learning-based[9]. In the literature, the most straightforward and widely used technique is blacklist[3] which is a list-based approach. However, this approach has shown some limitations, including the inability to detect newly created URLs that are not included in the list and the requirement of an expert to maintain the list. Due to the limitations mentioned previously, researchers have attempted to improve it, which led to the creation of heuristic-based methods that utilize handcrafted features to generate the characteristics of phishing websites. A famous model in this class is CANTINA[4] and the enhanced version CANTINA+[5]. Albeit they improved results, they continue to require time-consuming manual expert work.

To address the limitations of the preceding method, researchers have developed several machine learning-based techniques for detecting phishing URLs such as Naive Bayes[6], Decision Trees[14] and SVM[7]. While this approach has shown some satisfactory results, it has some limitations due to the presence of unseen features and the loss of semantic information. Following the success of deep learning techniques in the NLP task, after it overcomes the aforementioned limitations, several phishing URL detection solutions have emerged. Bahnsen et al.[15] propose using the LSTM model while also comparing it to a baseline machine learning model, namely random forest, which the LSTM model outperforms, thus showing the potential of the deep learning model. The work of Le et al[10]. further shows the superiority of the deep learning techniques after their proposed model, which combines word-level and character-level CNNs, outperformed the SVM model using various feature extraction techniques. While Tajaddodianfar et al. [16] propose an architecture similar to [10] combining character level and word level CNN. However, They use contextual word embeddings such as fast-Text.

Wei et al[8]. propose to use one-hot character-level encoding of the URL before performing the classification using CNN. While Rasymas and Dovydaitis[17] use combination of word and character level embedding as an input to a network consisting of CNN and GRU layers. Huang et al[9]. take a similar approach, embedding characters and words and using them in a network composed of CNNs and attention-based hierarchical Bi-LSTMs. Additionally, They use a cost-sensitive optimization cross-entropy objective function. Our work also uses a deep learning approach,

the TCN[11] model which shows great potential specially after the work of [12] and [13]. Where they propose a new URL representation while doing the classification using the TCN.

3. Temporal convolutional network (TCN)

Artificial neural networks are a rapidly developing field of science at the moment. Every day, we learn about new neural network models and applications.

TCN, or Temporal Convolutional Networks, are a type of Convolutional Neural Network that combines aspects of RNN and CNN architectures to perform sequence modelling tasks. This architecture was proposed by [11] and shows that it outperforms recurrent networks such as LSTMs over a variety of tasks and datasets. It is characterized by:

- The causal convolutions prevent information from being leaked between the past and the future.
- In the same way as RNNs, it maps any input sequence to a matching output sequence. Moreover, the residual connection and dilated convolutions combine to achieve long effective history sizes.

3.1. Causal Convolutions

TCN has two significant constraints: its output and input lengths must be equal, and its useable data must come from previous time steps. For the first constraint, TCN employs a 1-D fully convolutional network architecture [18]. A zero padded padding is applied to each hidden layer so that it is the same length as its corresponding input layer. In terms of the second constraint, TCN also uses causal convolutions where the output value is solely dependent on past and present inputs.

3.2. Dilated Convolutions

A dilated convolution filter is one that is applied to a region that is larger than its size by skipping a specified number of input values. This method, like pooling or stride convolutions, enlarges the receptive field. The output, however, is equal to the input. When using dilated convolutions, it is common to exponentially increase the dilation factor d with the array depth. This ensures that the receptive field covers all historical inputs and enables deep networks to achieve an extremely large receptive field as effective history. See Figure 1.

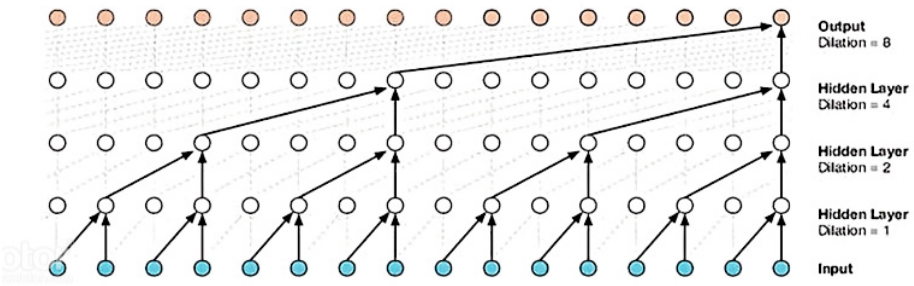


Fig. 1: Dilated causal convolutional layers [19]

3.3. Residual Connections

Because the network depth, filter size and dilation factor, determines the size of the receptive field of the TCN model. To generate a large enough receptive field, the TCN must be made deeper and larger. Residual connections are particularly effective in deep network training. Where it has been repeatedly shown that residual blocks enables very deep networks to learn changes in identity mapping instead of having to go through the entire transformation process.

4. Phishing detection using TCN

As shown in Figure 2 we propose in this paper to analyze URLs using temporal convolutional neural networks (TCN). The TCN model excels at natural language processing (NLP) and is well-suited for sequence-to-sequence tasks. As a result of their construction, They have: a larger memory capacity than recurrent architectures of comparable capacity, parallelism (convolutional layers), flexible receptive field size (possible to specify how far the model can see), and stable gradients (back-propagation through time, vanishing gradients).

In our case, we want to sensitize the network to detect the address distortions. We aim to train our TCN model to predict whether a given URL is phishing or legitimate correctly.

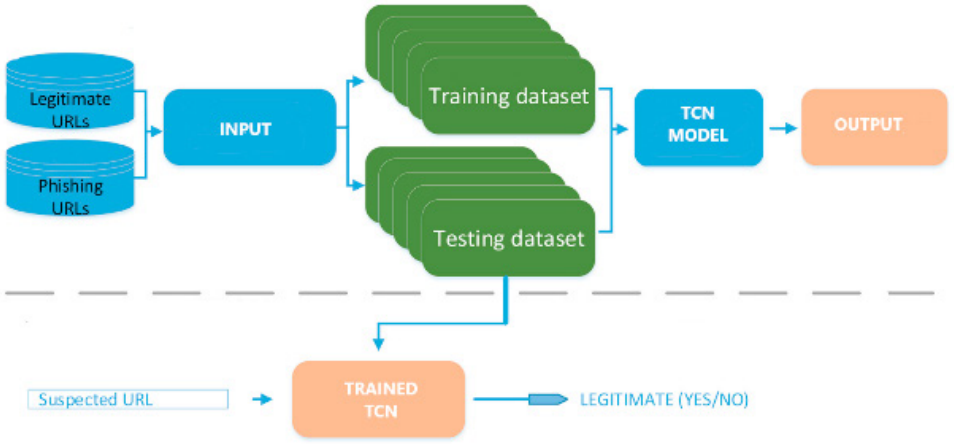


Fig. 2: A high-level diagram of our proposed architecture

4.1. Uniform Resource Locator (URL) structure

A Uniform Resource Locator (URL) addresses web pages. The figure 3 shows an example URL structure. The URL can be broken down into four parts: protocol, domain name, path, and file. First, there is the protocol that is used to access the web page. A domain name consists of two parts, a top-level domain (TLD) and the main level domain (mld) or subdomain, which is the part that the user (phisher) can modify. We can see in Figure 3 that example is the subdomain and .com is the top-level domain. Subdomains and TLDs are combined to create the fully qualified domain name that is used to visit the website. URLs often contain path and file, where path refers to a list of subdirectories leading from the server root to the file. Lastly, the file part refers to the name of the resource and may include parameters.

Fig. 3: URL structure[20]

4.2. Datasets

The performance of a phishing URL detection system is highly dependent on the data used to develop it. Therefore, data collection is an essential step in the development process. The data collected must include a sufficient number of representative examples of all the necessary classes to allow the machine learning models to learn parameters. We used

three different datasets for this review to study the performance of the deep learning algorithms and the importance of the attributes in the three datasets.

Dataset 1 is divided into the full dataset and the small dataset [20]. The data consists of legitimate and phishing URL instances, where phishing URLs are from PhishTank registries that are verified by multiple users. Legitimate URLs include publicly available, community-labelled and organized lists[21], and Alexa top ranking websites. From each URL, 111 features are extracted, as well as whether the URL is legitimate or not. There are a total of 88647 instances in full_dataset, out of which 58000 are legitimate, and 30647 are phishing URLs, which are designed to simulate real-world scenarios where there legitimate URLs outnumber the phishing URLs. As for the dataset_small, it contains 58645 instances, 27998 legitimate, and 30647 phishing URLs which are more or less balanced.

Dataset 2 contains a total of 11,000 websites, of which 5,500 are legitimate, and 5,500 are phishing sites. The dataset can be found in Kaggle[1]. Each row includes 14 unique features that distinguish phishing from legitimate websites, as well as a binary label indicating whether the website is phishing or legitimate. The dataset is populated with extracted features from URLs gathered from the PhishTank archive and the Google search engine.

Dataset 3 is downloaded from Kaggle[2] and contains the URL string and label. There are 549346 URLs in this dataset, of which 392924 are legitimate URLs and the other is phishing.

4.3. Our model

In this paper, we propose to use the TCN model in two different ways. The first one consists of creating a simple sequential model containing a TCN layer, a fully connected layer, and a softmax activation function. This network was tested on the first and the second.

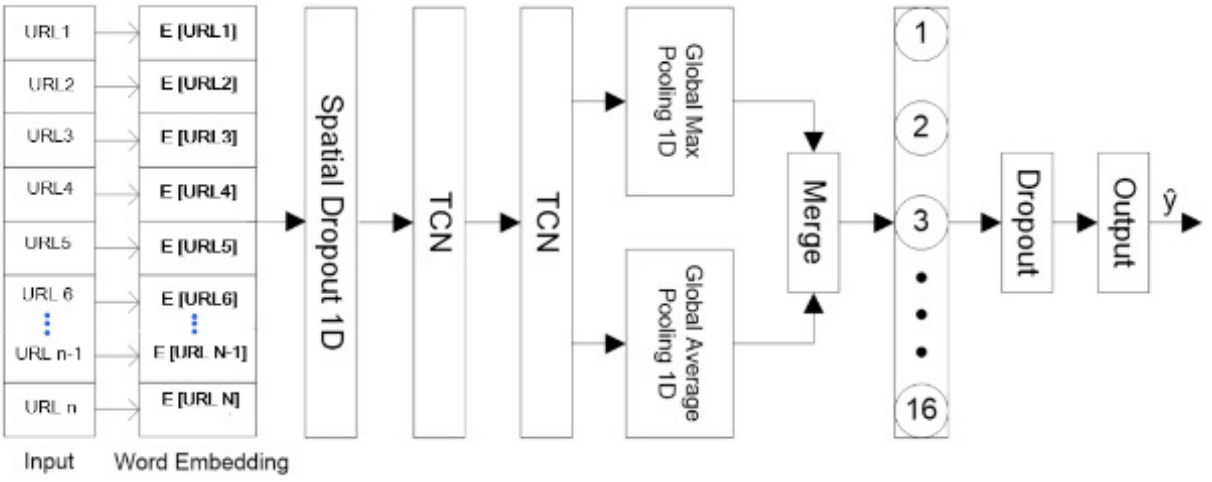


Fig. 4: Our propose phishing detection model

The other proposed model starts from a raw URL input passed through an embedding layer (GloVe, Word2Vec) to produce a dense vector representation used later. The remainder of the model was built by stacking two TCN blocks with a kernel size of 3 and expansion factors of 1, 2, and 4. The first one has 128 filters, while the other contains 64 filters. The result of each block will be a sequence.

The final sequence returned by the last block is passed to two different global pooling layers. Then the two results are concatenated and passed to a dense layer of 16 neurons. The result of the last layer is passed to the output layer with a softmax activation function, as shown in Figure 4. This network was tested on the third dataset.

[1] https://www.kaggle.com/datasets/sagarbanik/phishing-url-binary-datatset
[2] https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls

5. Experiments

In this paper, two TCN models have been proposed and compared with baseline classifiers such as k-Nearest Neighbors(KNN) and Logistic Regression(LR). All were implemented using Tensorflow [22] and Keras. Following are the experiments we conducted according to the dataset used:

Dataset 1: .

- As the class was unbalanced, we used undersampling. A technique used is data analysis and data mining to reduce the number of samples to equal the number of samples for the minority class. There are 30647 samples in this dataset representing phishing URLs (class 1), so we extracted 30647 samples from the other class 0, which represent legitimate URLs.
- As for the second experiment, oversampling was used to augement the classes in dataset. The dataset contains 58645 legitemate URLs (class 0), which we augemented to match the legitimate URLs.

For both cases, we applied 10-cross validation with 10 training epochs.

Dataset 2:. The dataset described was divided into 75% training set and 25% testing set. We trained our TCN model with 100 epochs.

Dataset 3:. TCN with word embedding.

The URLs were transformed to numbers, with each URL represented by an array of integers of a specific length. We set up the Tokenizer object, which was imported as a token from the Keras library. The tokenizer is then applied to the total URL, with each URL being allocated a unique number and each word in the URL being represented by a number. Then each sentence is converted into a number sequence that is allocated to it. By padding the sequences, we will be able to make all of the phrases the same length. The maxlen parameter determines the length. We set the vocab size to the amount of words in the token vector, plus one, so that if any word enters the model that hasn't been seen before, it will be allocated to that spot.

Table 1: Comparison of the proposed model in the three datasets.

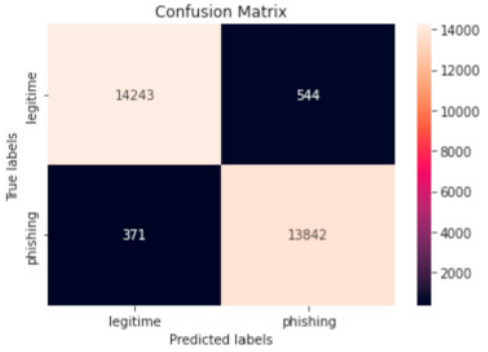
Dataset	Model	Accuracy	Precision	Recall	F-1 score
2	**TCN**	**99,96%**	**99%**	**99%**	**99%**
1 (dataset_small)	TCN	98,76%	98%	98%	98%
1 (dataset_full)	TCN	96%	97%	96%	96%
1 (dataset_full) with undersampling	TCN	96%	96%	96%	96%
1 (dataset_full) with oversampling	TCN	98%	97%	97%	97%
3	Word2vec embedding with TCN	98,95%	98%	98%	98%
3	GloVe embedding with TCN	97,76%	98%	98%	98%

For the glove embedding, we created an embedding vector dictionary with keys defined as words extracted from the glove embedding file (840B,300d) and values defined as the embeddings extracted from the file. This dictionary will contain all of the words contained in the glove-embedding file. We created a matrix containing only our vocabulary words and their embedding vectors. After that, we trained our TCN model over a period of 20 epochs.

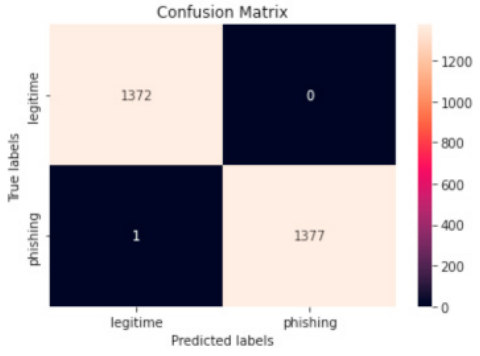
The above experiments and results are all publicly available through GitHub[3]. Table 1 summarises the results of the model in the three datasets.

After evaluating the performance of the different experiments, the best results were obtained using a simple TCN in the second dataset, which have 14 features. However, comparable results could be achived by using word embedding, which automatically extract features. In terms of results, word2Vec gives slightly better results than GloVe with an accuracy of about 1%.

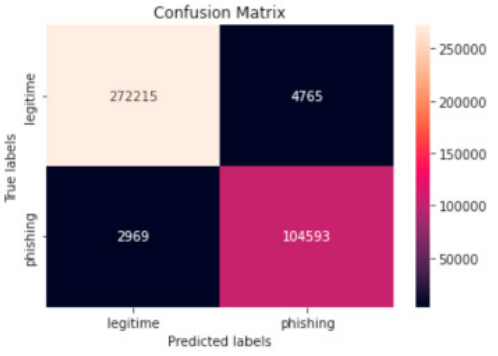
The confusion matrix in figures 5a, 5b, 5c shows that we successfully reduced the number of false negatives and false positives throughout all the experiments, especially in the second dataset, which demonstrates the effectiveness of the proposed model.



(a) Confusion matrix obtained from oversampling test set of dataset 1.



(b) Confusion matrix obtained from the test set of dataset 2.



(c) Confusion matrix obtained from oversampling the test set of dataset 3.

Fig. 5: Confusion matrices of our TCN model on the 3 dataset

Table 2 compares the results obtained by our model using the first and the dataset against other baseline approaches. In which we can observe that our model has outperform the others baseline approaches.

Table 2: Comparison of the proposed model with other baselines in the first and the second dataset.

Dataset	Model	Accuracy	Precision	Recall	F-measure
D2	KNN	94.25%	94.2%	94%	94.10%
	TCN	99.96%	99%	99%	99%
D1 small	KNN	85.64%	85.4%	87.2%	86.31%
	TCN	98,76%	98%	98%	98%

[3] https://github.com/abdelkarim-remmide/phishing-url-detection-using-TCN

Table 3 compares the results obtained by our model using the third dataset against other baseline approaches. We can see that our proposed model has outperformed all the baseline approaches.

Table 3: Comparison of the proposed model with other baselines in the third dataset.

Method	Accuracy	Precision	Recall	F1-score
KNN	91%	90%	86%	88%
LR + CountVectorizer	96%	93%	96%	95%
LR + RegexpTokenizer	96%	94%	96%	96%
Word2Vec + TCN	**98,95%**	**98%**	**98%**	**98%**
GloVe + TCN	97,76%	98%	98%	98%

Our model has outperform the baseline model across all metrics in the third dataset. Among the two propose TCN model with word embedding word2Vec achieved a better results. Our best proposed model is then compared with the state of the art model in the table 4.

Table 4: Comparison of the proposed model with state of the art proposal.

Refrence	Accuracy	Precision	Recall	F1-score
[23]	95%	-	-	96%
[24]	95.80%	96%	94%	95%
[25]	95.6%	97.33%	93.78%	95%
[12]	-	95.97%	95.96%	95.96%
[13]	99.27%	97.42%	95.44%	96.38%
Our model	**98,95%**	**98%**	**98%**	**98%**

Our model achieved slightly better results in terms of precision, recall, and f1-score than the other proposal. At the same time, maintaining an approximate accuracy of Liang et al. [13] work.

6. Conclusion

In this paper, we presented and evaluated a Temporal convolutional network (TCN) with word embedding, a deep learning-based model for detecting phishing URLs. Experiments show that our model is better than the other baseline model, where our proposed model has achieved 98% in terms of accuracy, precision, recall, and f1-score. As for future work, we aim to improve this model detection against modern phishing attacks and test it on real-world systems.

References

[1] Anti-Phishing Working Group. (2021). Phishing Activity Trends Report 3st Quarter 2021. [online] Available at: https://docs.apwg.org/reports/apwg_trends_report_q3_2021.pdf [Accessed 15 Janv 2022].

[2] S. Sheng, M. Holbrook, P. Kumaraguru, L. F. Cranor, J. Downs, Who falls for phish? a demographic analysis of phishing susceptibility and effectiveness of interventions, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '10, Association for Computing Machinery, New York, NY, USA, 2010, p. 373–382. doi:10.1145/1753326.1753383.
URL https://doi.org/10.1145/1753326.1753383

[3] R. S. Rao, A. R. Pais, An enhanced blacklist method to detect phishing websites, in: International Conference on Information Systems Security, Springer, 2017, pp. 323–333.

[4] Y. Zhang, J. I. Hong, L. F. Cranor, Cantina: A content-based approach to detecting phishing web sites, WWW '07, Association for Computing Machinery, New York, NY, USA, 2007, p. 639–648. doi:10.1145/1242572.1242659.
URL https://doi.org/10.1145/1242572.1242659
[5] G. Xiang, J. Hong, C. P. Rose, L. Cranor, Cantina+: A feature-rich machine learning framework for detecting phishing web sites, ACM Trans. Inf. Syst. Secur. 14 (2) (sep 2011). doi:10.1145/2019599.2019606.
URL https://doi.org/10.1145/2019599.2019606
[6] D. Chiba, K. Tobe, T. Mori, S. Goto, Detecting malicious websites by learning ip address features, in: 2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet, IEEE, 2012, pp. 29–39.
[7] B. Banik, A. Sarma, Phishing url detection system based on url features using svm, International Journal of Electronics and Applied Research 5 (2018) 40–55. doi:10.33665/ijear.2018.v05i02.003.
[8] W. Wei, Q. Ke, J. Nowak, M. Korytkowski, R. Scherer, M. Woźniak, Accurate and fast url phishing detector: A convolutional neural network approach, Computer Networks 178 (2020). doi:10.1016/j.comnet.2020.107275.
[9] Y. Huang, Q. Yang, J. Qin, W. Wen, Phishing url detection via cnn and attention-based hierarchical rnn, Proceedings - 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering, TrustCom/BigDataSE 2019 (2019) 112–119doi:10.1109/TrustCom/BigDataSE.2019.00024.
[10] H. Le, Q. Pham, D. Sahoo, S. C. H. Hoi, Urlnet: Learning a url representation with deep learning for malicious url detection, PODS 2017: Proceedings of the ACM Symposium on Principles of Distributed Computing, Washington, DC, July 25 (2018).
URL http://arxiv.org/abs/1802.03162
[11] S. Bai, J. Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv preprint arXiv:1803.01271 (2018).
[12] Y. Liang, J. Kang, Z. Yu, B. Guo, X. Zheng, S. He, Leverage temporal convolutional network for the representation learning of urls, in: 2019 IEEE International Conference on Intelligence and Security Informatics (ISI), 2019, pp. 74–79. doi:10.1109/ISI.2019.8823362.
[13] Y. Liang, Q. Wang, K. Xiong, X. Zheng, Z. Yu, D. Zeng, Robust detection of malicious urls with self-paced wide amp; deep learning, IEEE Transactions on Dependable and Secure Computing 19 (2) (2022) 717–730. doi:10.1109/TDSC.2021.3121388.
[14] S. Marchal, J. François, R. State, T. Engel, Phishscore: Hacking phishers' minds, in: 10th International Conference on Network and Service Management (CNSM) and Workshop, IEEE, 2014, pp. 46–54.
[15] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, F. A. Gonzalez, Classifying phishing urls using recurrent neural networks, eCrime Researchers Summit, eCrime (2017) 1–8doi:10.1109/ECRIME.2017.7945048.
[16] F. Tajaddodianfar, J. W. Stokes, A. Gururajan, Texception: A character/word-level deep learning model for phishing url detection, in: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 2857–2861. doi:10.1109/ICASSP40776.2020.9053670.
[17] T. Rasymas, L. Dovydaitis, Detection of phishing urls by using deep learning approach and multiple features combinations, Baltic Journal of Modern Computing 8 (2020) 471–483. doi:10.22364/BJMC.2020.8.3.06.
[18] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3431–3440.
[19] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: A generative model for raw audio, arXiv preprint arXiv:1609.03499 (2016).
[20] G. Vrbančič, I. Fister, V. Podgorelec, Datasets for phishing websites detection, Data in Brief 33 (2020) 106438. doi:https://doi.org/10.1016/j.dib.2020.106438.
URL https://www.sciencedirect.com/science/article/pii/S2352340920313202
[21] C. Lab, Others, Url testing lists intended for discovering website censorship, https://github.com/citizenlab/test-lists (2014).
URL https://github.com/citizenlab/test-lists
[22] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, Tensorflow: A system for large-scale machine learning (2016). arXiv:1605.08695.
[23] H. Abutair, A. Belghith, S. AlAhmadi, Cbr-pds: a case-based reasoning phishing detection system, Journal of Ambient Intelligence and Humanized Computing 10 (7) (2019) 2593–2606.
[24] A. Aljofey, Q. Jiang, Q. Qu, M. Huang, J.-P. Niyigena, An effective phishing detection model based on character level convolutional neural network from url, Electronics 9 (9) (2020). doi:10.3390/electronics9091514.
URL https://www.mdpi.com/2079-9292/9/9/1514
[25] W. Wang, F. Zhang, X. Luo, S. Zhang, Pdrcnn: precise phishing detection with recurrent convolutional neural networks, Security and Communication Networks 2019 (2019).