Baltic J. Modern Computing, Vol. 8 (2020), No. 3, 471-483
https://doi.org/10.22364/bjmc.2020.8.3.06

Detection of Phishing URLs by Using Deep Learning Approach and Multiple Features Combinations

Tomas RASYMAS, Laurynas DOVYDAITIS

Vilnius University, Kaunas Faculty, Institute of Social Sciences and Applied Informatics,
Muitinės Street 8, Kaunas, Lithuania

tomas.rasymas@gmail.com, laurynas.dovydaitis@knf.vu.lt

Abstract. Phishing detection is mostly performed through the usage of blacklists. However, blacklists cannot be exhaustive and lack the ability to detect newly generated phishing URLs. In recent years, increased attention has been given to exploring machine learning techniques in order to improve the universality of phishing URL detectors. This article aims at presenting our results on phishing URLs classification where three different features: lexical features, character level embeddings, and word level embeddings were compared with the view to find an approach that maximizes the ratio of phishing URL detection. In addition, a new deep neural network architecture for that problem was suggested. The said deep neural network consists of combined multiple CNN and LSTM layers. The 94.4% accuracy was achieved by combining character and word level embeddings.

Keywords: social engineering, phishing, deep learning, phishing URL classification.

1. Introduction

In recent years, the use of internet technologies has expanded in the areas of online social networking, online banking, and other services that collect and store sensitive personal data. This has made people's lives easier. On the other hand, computer networks pose many different security threats worldwide, such as sensitive personal data thefts. One of these serious threats is Phishing, which tries to deceive its victims into giving their private information. It is imperative to act on such threats in a timely manner.

Cyber-security threats appear in our well-connected computer networks and their number is constantly growing. Individuals and organizations are targeted by cyber attackers every day. Social engineering is one of the techniques used in these types of attacks, yet the issue is not fully solved due to different challenges that need to be addressed by security experts and human behaviour analysts.

Phishing is a form of cyber-attack typically performed by sending false correspondence that seems to originate from a legitimate source. The objective of such

an attack is to gain access to sensitive information such as credit card numbers, credential data, or even to download and activate malware applications and viruses on the target machines. Most of these attacking techniques are implemented through spreading spoofed URLs (Uniform Resource Locators).

The analysis of recent studies (4Q2019) conducted by the Anti-Phishing Working Group, Inc. (APWG), has shown that the number of phishing sites is growing. The total number of phishing sites detected by the APWG in the fourth quarter was 162,155. This number is lower than 266,387 recorded in Q3 and 182,465 recorded in Q2, and higher than 138,328 recorded in Q4 2018. The APWG identified that the most targeted sectors are SaaS/Webmail (30.80%), payment (19.80%), financial institutions (19.40%), social media (6.80%), telecom, e-commerce, retail, and cloud storage (12.10%). As reported by the APWG, phishing activity has caused aggregated losses in the billions of dollars in large and small companies (Anti-Phishing Working Group, 2020).

Basically, the phishing website detection can be performed via reactive or proactive means. Reactive approaches use blacklists of malicious URLs. These lists are constructed by manually reporting or by crawling the web and searching for such malicious URLs. On the other hand, proactive methods mitigate the problem by analysing different characteristics of a website in real time to evaluate a potential risk of a website (Correa Bahnsen et al., 2017). The drawback of a reactive approach is that it is difficult to detect newly created malicious websites because they have not been reported and added to blacklists. The process of creating and publishing a website is quite easy, it can be done quickly and requires very little resources. It is highly unrealistic to track all possible phishing websites and add them to a blacklist. Therefore, this paper focuses on the proactive approach.

Proactive approaches rely on the analysis of a website URL or its content. To use such methods, one needs to obtain relevant information about an URL or the corresponding site content, such as obtaining site keywords and site forms, and other features, such as website ranking and IP address, which are obtained with the help of a search engine service or a DNS (Domain name service). These methods, based on URL and web content features, require local computing resources, network access, and third-party services. The detection efficiency is low, and when phishing attacks continue to change and escalate, the effectiveness of these features is waning (Wang et al., 2019).

This paper proposes a method of detection of phishing websites by using only their URLs. This method does not require any third-party services such as search engine or DNS services. Different combinations of URL features such as lexical, char level and word level embeddings are assessed.

2. Related Work

The latest research (Wanda and Jie, 2019; Wei et al., 2019; Yang, Zhao et al., 2019; Yang, Zuo et al., 2019; Kiruthiga, 2019; Kulkarni and Brown, 2019; Wang et al., 2019) shows that a deep learning approach for classifying URLs yields better results. One of the challenges of using this approach is that in order to train a neural network a vast

amount of data is needed. Since there are no publicly available datasets, researchers use their own private datasets, which brings us to another challenging task of comparing different methods of work.

With the popularity of deep learning algorithms, researchers were focusing on extracting meaningful features from an URL and then using standard machine learning methods to classify URLs by using those features. Some examples of the features used are: length of an URL, number of dots in an URL, number of slashes in an URL, whether an URL is secure or not, does an URL contain words from pre-defined suspicious words dictionary, and so on (Patil and Patil, 2018; Correa Bahnsen et al, 2017; Yang, Zhao et al., 2019; Kiruthiga and Akila, 2019; Kulkarni and Brown, 2019; Aung et al., 2019; Sahoo et al., 2019). Other research focused on third-party features such as WHOIS and DNS information (Kuyama et al., 2016; Aung et al., 2019; Sahoo et al., 2019). All these features were transferred to a machine learning algorithm which made a final prediction about a website. Support vector machines achieved a 97.8% detection accuracy when classifying WHOIS and DNS features (Kuyama et al., 2016). Other trending algorithms for lexical features classification are logistic regression, decision trees, and naïve Bayes (Kiruthiga and Akila, 2019, Aung et al., 2019; Sahoo et al., 2019). Manual selection of URL features is time-consuming and requires domain knowledge; and it is harder to adapt such features to the diversity of new URLs.

Deep learning is part of machine learning, where a set of algorithms help imitate the structure and function of the human brain. These algorithms operate on raw input signals and automate the process of feature extraction (Wanda and Jie, 2019). Deep learning-based research may be split into three categories by neural network architecture: recurrent neural networks, convolutional neural networks, or a mix of both. Features used in research may also be split into three categories: character level embeddings, word level embeddings, or a mix of both. Some authors (Wanda and Jie, 2019) propose a method for classifying phishing URLs by using both character and word level embeddings to feed convolutional neural networks. They gathered a dataset of 16,000,000 different URLs. The researchers of (Correa Bahnsen et al., 2017) used a recurrent neural network with character level embeddings and a 2,000,000 URLs dataset. They achieved a 93,40% F1 score. The authors of (Wei et al., 2019) used word level embeddings with a convolutional neural network and a dataset of 1,523,966 samples. They achieved an 86.63% accuracy of classifying URLs into the phishing and legitimate ones. In the (Yang, Zhao et al., 2019) research on character level embeddings and a network with a mix of convolutional and recurrent layers were used. The authors gathered a dataset of 1,021,758 samples and achieved a 98.61% classification accuracy. The researchers of (Yang, Zuo et al., 2019) used similar network architecture with word level embeddings. The authors of (Wang et al., 2019) experimented with character level embeddings and a network of convolutional and recurrent layers. They used a dataset of 5,118,727 URLs samples and achieved an F1 score of 95.52%. In the research of (Saxe and Berlin, 2017) the authors used character level embeddings and a convolutional neural network to classify URLs and achieved an AUC (Area under the curve) of 99,30%. A dataset that was used contained 19,067,879 samples of URLs. The authors of (Le, Pham et al., 2018) used a combination of character and word level embeddings with a convolutional neural network. They used a dataset of 15,000,000 samples and achieved an AUC of 99,29%. Other authors of (Le, Markopoulou et al., 2010; Abdi and Wenjuan,

2017; Aung et al., 2019; Sahoo et al., 2019; Sahingoz et al., 2018) used similar neural network architectures with character or word level embeddings.

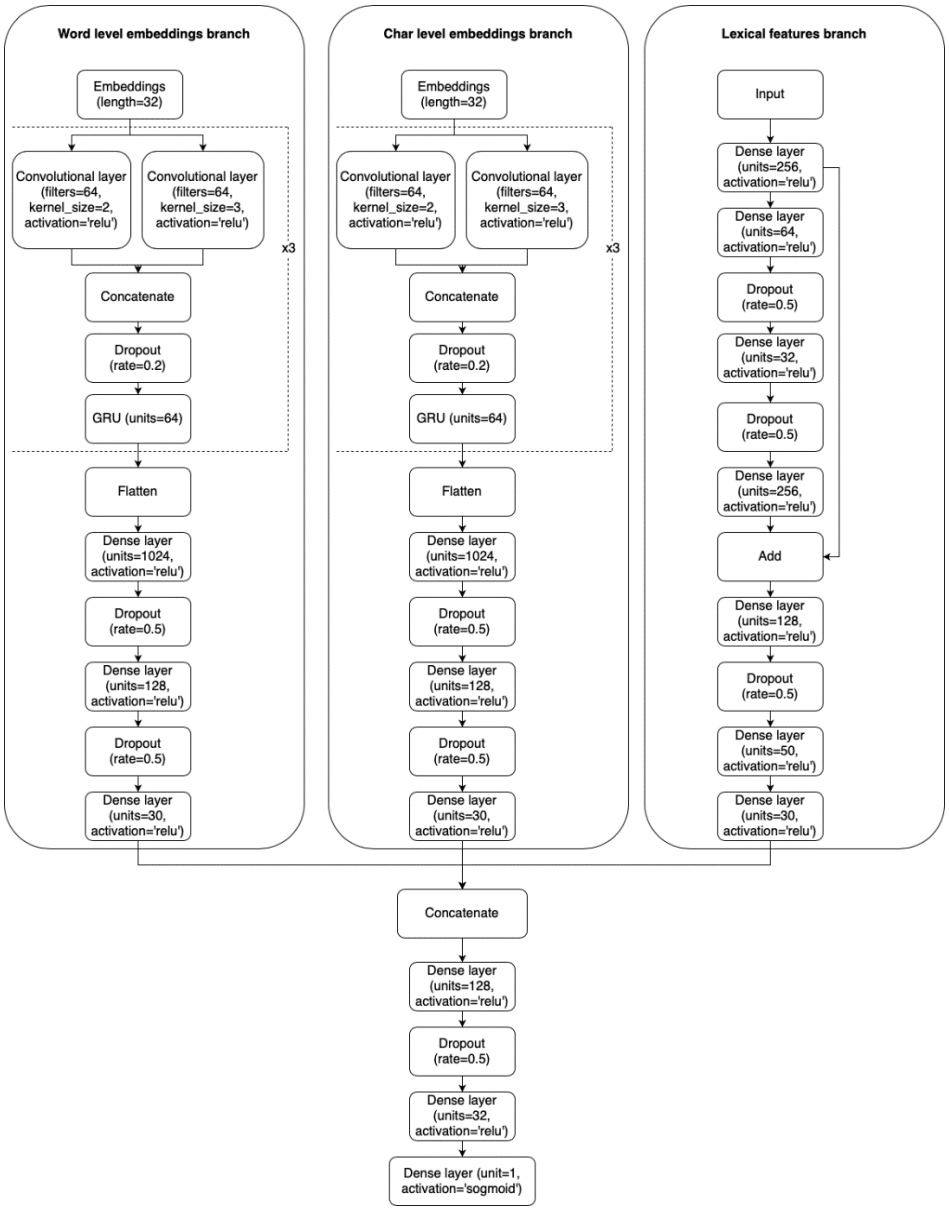


Figure 1: Proposed neural network architecture

3. Proposed Method

In some studies, for detecting phishing URLs (Aung et al., 2019), the authors use character or word level embeddings. But it is unclear, which features are better and why lexical features were not used as an additional feature for neural networks. In this paper we proposed to use neural network architecture with more hidden layers for the purposes of classification and share the results on the impact of different features in the context of final accuracy. A pipeline used for embedding features is standard: a URL was split into tokens, characters or words depending of the features used. Then vectors of tokens were padded to meet the same length.

The proposed neural network architecture is made of various hidden layer branches, depending on the features used. We were using neural network architecture ideas taken from the authors from a related work section, only going a few steps deeper. A full neural network architecture with all features (character and word level embeddings and lexical features) is demonstrated below.

Character and word level branches are of similar architecture. The embedding layer is usually used as a first layer of the deep neural network structure for a Natural Language Processing (NLP) problem. This layer maps integers of tokens into a fixed multi-dimensional space. A vector returned from the embedding layer stores relations among tokens (single characters for character embeddings, single word for word embeddings). Following the embedding layer, two convolutional layers were used. Each convolutional layer was applied with the view to extract the most useful features and remove unnecessary information. Each layer assesses one window-sized cluster of consecutive characters and extracts features from them. A rectified linear unit (ReLU) activation function was also used following each convolutional layer. Outputs of each convolutional layer were concatenated into a single vector and passed to a recurrent GRU (Gated Recurrent Unit) layer. This process was repeated three times. The lexical features branch consists of simple fully connected layers. We were using 60 different lexical features. Those features were selected based on (Patil and Patil, 2018; Sahoo et al., 2019) papers with a few additional custom features. A full list of lexical features is displayed in Table 1.

Table 1: Lexical feature list

URL length	number of digits in a path
contains a port URL	number of upper-case characters in a path
has a protocol URL	number of lower-case characters in a path
number of special characters in a URL	upper and lower-case characters ratio in a path
entropy URL	number of special characters in a path
number of repeating symbols in a URL	number of suspicious words in a path

count of repeating symbols in a URL	maximum word length in a path
number of %20 in a URL	minimum word length in a path
number of . in a domain name	average word length in a path
number of @ in a domain name	length of a query
number of – in a domain name	number of & in a query
contains ip in a domain name	number of = in a query
contains www in a domain name	number of + in a query
domain name length	number of - in a query
number of digits in a domain name	number of , in a query
number of upper-case characters in a domain name	number of _ in a query
number of lower-case characters in a domain name	number of (in a query
upper and lower-case characters ratio in a domain name	number of) in a query
number of special characters in a domain name	number of [in a query
number of suspicious words in a domain name	number of] in a query
maximum word length in a domain name	number of . in a query
min word length in a domain name	number of digits in a query
average word length in a domain name	number of upper-case characters in a query
path length	number of lower-case characters in a query
number of / in a path	upper and lower case characters ratio a query
number of . in a path	number of special characters in a query
number of , in a path	number of suspicious words in a query
number of @ in a path	maximum word length in a query
number of - in a path	minimum word length in a query
number of in a path	average word length in a query

The final prediction is made by concatenating all outputs of those three branches. Dropout layers are used to control network overfitting.

4. Experimental evaluation

A dataset was collected in order to perform experiments. The dataset was obtained by collecting publicly available data plus collecting data from phishtank.com. Overall, we collected 2,585,146 data samples. The distribution of samples is shown below.

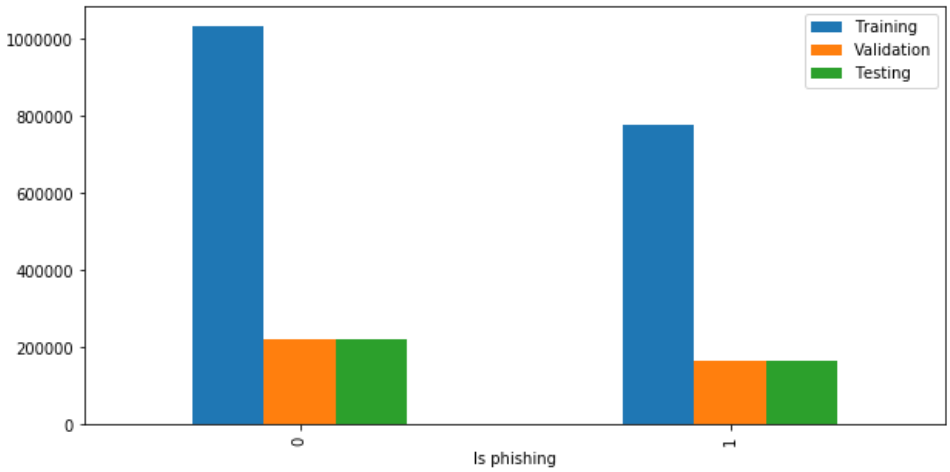
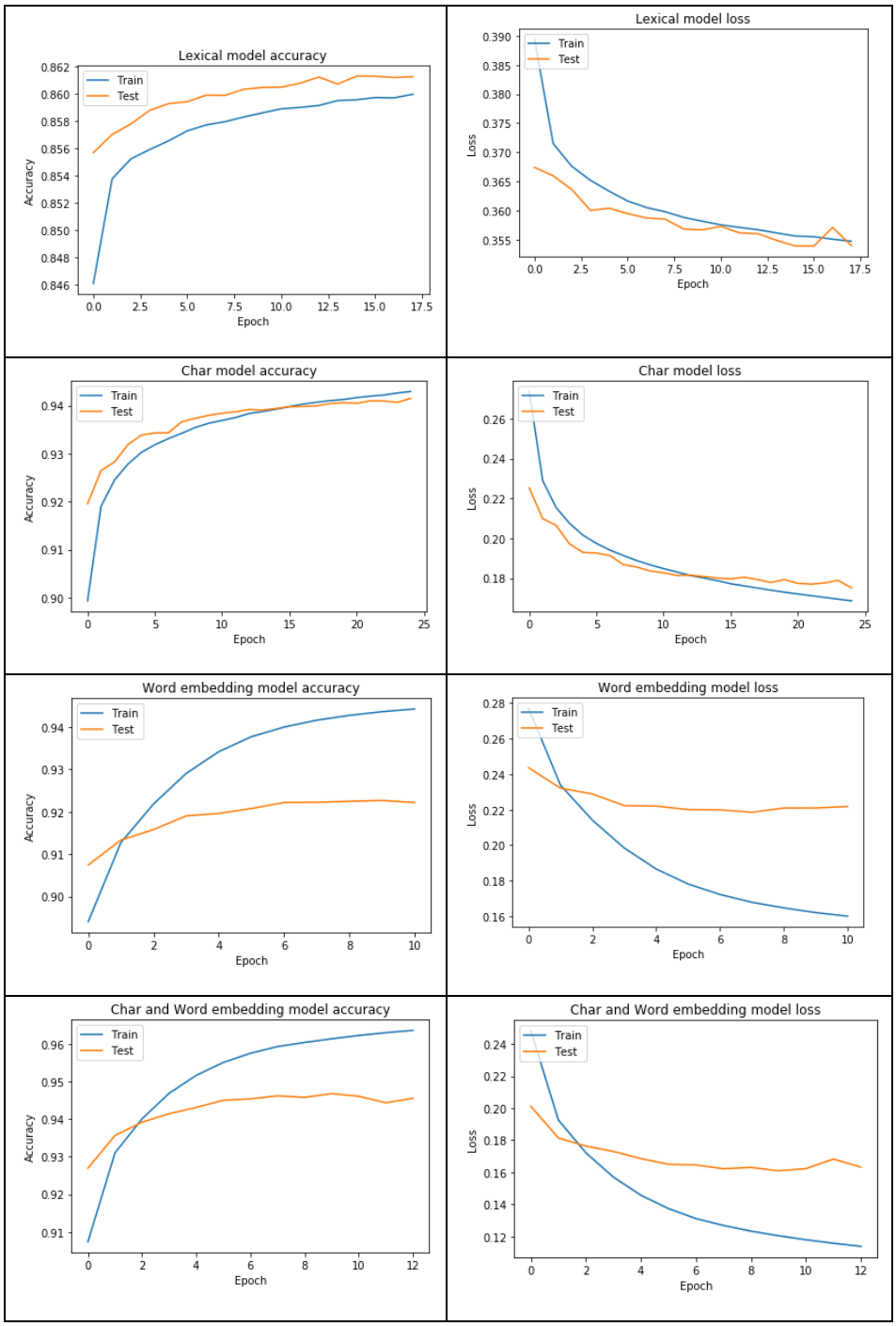


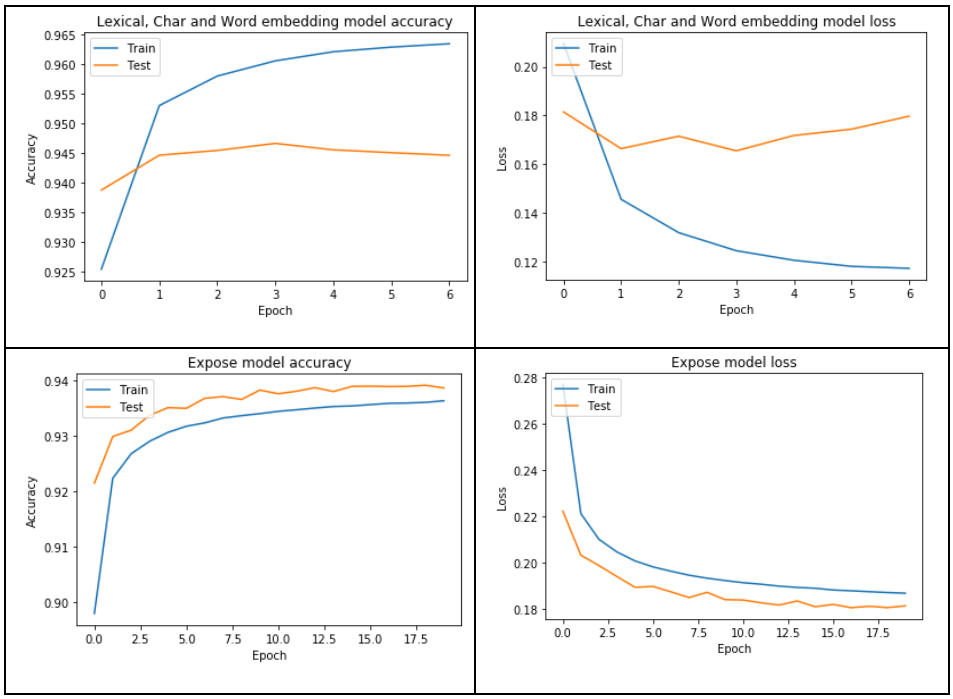
Figure 2: Phishing dataset distribution.

The suggested model was assessed in accordance with the research made in (Saxe and Berlin, 2017). There its authors suggested to use an eXpose network architecture which is based on character level embeddings and a convolutional neural network.

While performing the assessment, early stopping technique was used to stop the model from overfitting. Its loss was being monitored and if no reduction was being observed for 3 epochs, the training process would be stopped. In addition, before the training, we had searched for optimal learning rate as proposed in (Smith, 2017). A batch size of 64 was used.

Table 2: Model accuracy and loss





As can be seen in the graphics above, the results show that lexical model loss stops decreasing rapidly and after reaching 0.35 it stops decreasing altogether. The networks of word level embeddings, character, and word level embeddings, as well as combined lexical, character and word level embeddings tend to overfit. From the graphics above, we can also see that the difference between training and testing accuracies and loss is growing. This overfitting may be influenced by word level embeddings. If the model used only these features, it would overfit; however, reducing the number of features would not prevent it from overfitting. The eXpose network and network with character level embeddings generalizes well.

After assessing the performance of different models, the best performance was achieved by using a combination of character, and word level embeddings. Comparable results were achieved by using lexical, character, and word level embeddings. The results show that the overall model accuracy does not increase by adding lexical features only. The results are shown in Table 3.

The results show that lexical features have no impact on model accuracy. Despite lexical features having many different text properties, such information is insufficient in order to classify URLs into the phishing or benign ones. The usage of embeddings captures phishing URL patterns better and allows achieving better classification accuracy.

Having compared the eXpose data versus the character and word level embedding model, we managed to increase an F1 score by 0.5%, accuracy by 0.6% and precision by 1%. The comparison of ROC-AUC (Receiver operating characteristic – Area under the curve) curves of different models is provided in Figure 3.

Table 3: Comparison of model performance

Models	accuracy	precision	recall	f1	roc_auc
Lexical model	0.861354	0.935280	0.726785	0.817956	0.844533
Char model	0.941215	0.972225	0.888211	0.928321	0.934590
Word model	0.918192	0.956640	0.847528	0.898785	0.909359
Lexical, Char and Word model	0.942801	0.972089	0.892152	0.930407	0.936470
Char and Word model	0.944001	0.971699	0.895414	0.931998	0.937927
Expose model	0.938098	0.960799	0.891954	0.925097	0.932330

The results show that lexical features have no impact on model accuracy. Despite lexical features having many different text properties, such information is insufficient in order to classify URLs into the phishing or benign ones. The usage of embeddings captures phishing URL patterns better and allows achieving better classification accuracy.

Having compared the eXpose data versus the character and word level embedding model, we managed to increase an F1 score by 0.5%, accuracy by 0.6% and precision by 1%. The comparison of ROC-AUC (Receiver operating characteristic – Area under the curve) curves of different models is provided in Figure 3.

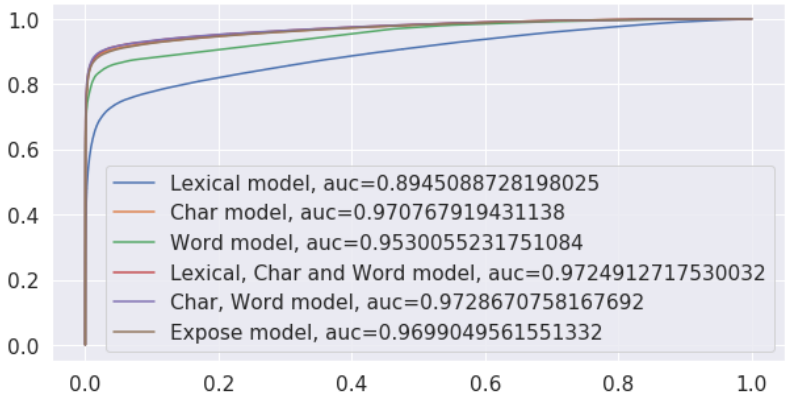


Figure 3: Model AUC

ROC-AUC curves show that lexical and word level embedding networks are outperformed by others. A confusion matrix comparison between the eXpose versus character and word (CW) level embedding networks is shown in Figure 4.

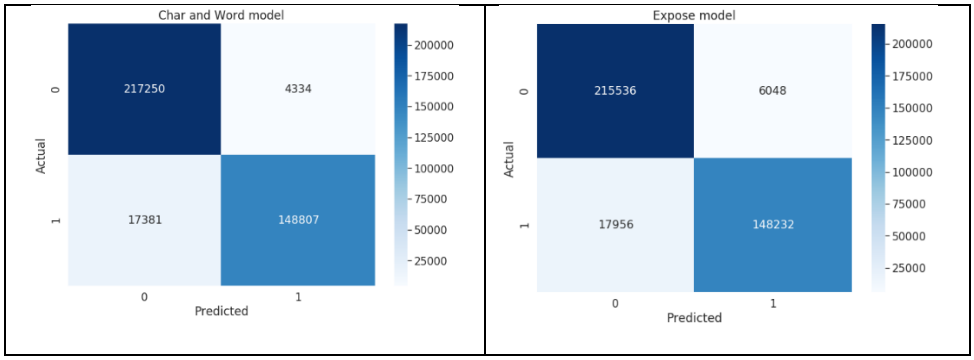


Figure 4: Confusion matrices for Character and Word versus the eXpose model

As seen from a confusion matrix, we managed to decrease the number of false negatives and false positives.

In order to check whether accuracy differences between the eXpose and the proposed character and word model are statistically significant, the McNemar's test was used. Both p-tail values are under 0.00000%.

Table 4: McNemar statistic values

Parameter	**Value**
#1 – CW model and eXpose model positive prediction	359,321
#2 - CW model and eXpose model prediction wrong	17,241
#3 – CW model wrong prediction, eXpose model positive prediction	4,474
#4 – CW model positive prediction, eXpose model wrong prediction	6,736
McNemar test statistics	4.56032
p-value *1 tail*	0.00000
p-value *2 tails*	0.00000

Since, typically, differences are considered statistically significant when p-tail values are under 0.005%, this shows that the character and word model performs better than the eXpose.

5. Conclusions and future work

By using the proposed deep neural network architecture on a given dataset, the classification of phishing URLs and benign URLs may be performed with a 94.4% accuracy.

Different feature combinations (lexical, character level embeddings, word level embeddings, character, and word level embeddings, lexical and character and word level embeddings) were assessed. The results have shown that the best accuracy is achieved by using the character and word level embeddings model. This implies that by combining two embeddings we can achieve better accuracy.

Our model with the best combination of features achieves better accuracy compared to the reproduced eXpose (Saxe and Berlin, 2017) model on the same dataset. We increased an F1 score by 0.5%, accuracy by 0.6% and precision by 1%. The McNemar's test showed that these results may be considered statistically significant.

Future work shall include replicating more methods in a related work on the same dataset with the view to get a more accurate comparison on the accuracy metrics.

References

Abdi, F.D., Wenjuan, L. (2017). Malicious URL Detection Using Convolutional Neural Network. International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), 7(6). https://doi.org/10.5121/ijcseit.2017.7601

Anti-Phishing Working Group, Inc. (2020), https://docs.apwg.org/reports/apwg_trends_report_q4_2019.pdf

Aung, E. S., Zan, T., Yamana, H. (2019). A Survey of URL-based Phishing Detection. https://db-event.jpn.org/deim2019/post/papers/201.pdf

Correa Bahnsen, A., Contreras Bohorquez, E., Villegas, S., Vargas, J., González, F. A. (2017). Classifying Phishing URLs Using Recurrent Neural Networks. http://albahnsen.github.io/files/Classifying Phishing URLs Using Recurrent Neural Networks_cameraready.pdf

Kiruthiga, R., Akila, D. (2019). Phishing Websites Detection Using Machine Learning, 112. International Journal of Recent Technology and Engineering (IJRTE), 8(2S11). https://doi.org/10.35940/ijrte.B1018.0982S1119

Kulkarni, A., Brown, L. L. (2019). Phishing Websites Detection using Machine Learning. In IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, Issue 7. https://thesai.org/Downloads/Volume10No7/Paper_2-Phishing_Websites_Detection_using_Machine_Learning.pdf

Kuyama, M., Kakizaki, Y., Sasaki, R. (2016). Method for Detecting a Malicious Domain by using WHOIS and DNS features. http://d.researchbib.com/f/9nBGVlAmLhpTEz.pdf

Le, A., Markopoulou, A., Faloutsos, M. (2010). PhishDef: URL Names Say It All. https://arxiv.org/pdf/1009.2275.pdf

Le, H., Pham, Q., Sahoo, D., Hoi, S. C. H. (2018). URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection. https://arxiv.org/pdf/1802.03162.pdf

Patil, D. R., Patil, J. B. (2018). Feature-based Malicious URL and Attack Type Detection Using Multi-class Classification. ISC International Journal of Information Security (ISeCure), Vol. 10, No. 2, 141-162. http://www.isecure-journal.com/article_63041 _7477a24f5e3a62ba0bf7a8416d70c51d.pdf

Sahingoz, O.K., Baykal, S.I., Bulut, D. (2018). Phishing Detection from URLs by Using Neural Networks. Computer Science & Information Technology (CS & IT), 41–54. https://doi.org/10.5121/csit.2018.81705

Sahoo, D., Liu, C., Hoi, S. C. H. (2019). Malicious URL Detection using Machine Learning: A Survey. arXiv preprint arXiv:1701.07179

Saxe, J., Berlin, K. (2017). eXpose: A Character-Level Convolutional Neural Network with Embeddings for Detecting Malicious URLs, File Paths and Registry Keys. https://arxiv.org/pdf/1702.08568.pdf

Smith, L. N. (2017). Cyclical Learning Rates for Training Neural Networks. https://arxiv.org/pdf/1506.01186.pdf

Wanda, P., Jie, H.J. (2019). URLDeep: Continuous Prediction of Malicious URL with Dynamic Deep Learning in Social Networks. International Journal of Network Security, 21(6), 971–978. https://doi.org/10.6633/IJNS.201911

Wang, W., Zhang, F., Luo, X., Zhang, S. (2019). PDRCNN: Precise Phishing Detection with Recurrent Convolutional Neural Networks. 2019, 15. https://doi.org/10.1155/2019/2595794

Wei, B., Hamad, R. A., Yang, L., He, X., Wang, H., Gao, B., Woo, W. L. (2019). A Deep-Learning-Driven Light-Weight Phishing Detection Sensor. Sensors, 19(19), 4258. https://doi.org/10.3390/s19194258

Yang, P., Zhao, G., Zeng, P. (2019). Phishing website detection based on multidimensional features driven by deep learning. IEEE Access, 7, 15196–15209. https://doi.org/10.1109/ACCESS.2019.2892066

Yang, W., Zuo, W., Cui, B. (2019). Detecting Malicious URLs via a Keyword-Based Convolutional Gated-Recurrent-Unit Neural Network. IEEE Access, 7, 29891–29900. https://doi.org/10.1109/ACCESS.2019.2895751

Received August 14, 2020, accepted September 19, 2020