Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 9, September : 2023

USING MACHINE LEARNING TO DETERMINE BLOCKED URLS

[#1]ANDRI BHAVANI,

[#2]Dr.V.BAPUJI, ***Associate Professor& HOD,***

Department of Master of Computer Applications,

VAAGESWARI COLLEGE OF ENGINEERING, KARIMNAGAR, TELANGANA.

ABSTRACT: A malicious URL, often known as a malicious website, is a common mechanism for storing unwelcome content such as spam, malicious ads, phishing, and drive-by vulnerabilities, to mention a few. It is critical to identify harmful URLs as soon as possible. Techniques such as blacklisting, regular expression, and signature matching have already been employed in research. These methods are useless for detecting versions of existing malicious URLs or wholly new URLs. To address this issue, a machine learning-based solution might be proposed. This type of solution necessitates extensive research in the fields of feature engineering and feature representation of security objects such as URLs. Furthermore, feature engineering and feature representation resources must be regularly updated to handle both versions of current URLs and whole new URLs. Deep learning has enabled AI systems to achieve human-level performance in a range of disciplines, even outperforming human vision in a number of computer vision applications. They can automatically extract the best feature representation from raw inputs. Deep URL Detect (DUD) encrypts raw URLs using character level embedding for use and translation in the cyber security arena. Character level embedding is a cutting-edge method for encoding characters in numeric form in natural language processing (NLP). Hidden layers in deep learning architectures take properties from character level embedding and then apply a non-linear activation function to determine whether the URL is potentially dangerous. We compare and contrast different cutting-edge deep learning-based character level embedding approaches for detecting counterfeit URLs in this paper. Several trials are carried out in order to determine the most effective deep learning-based character level em- bedding model. All experiments employing various deep learning-based character level embedding models are carried out for 500 epochs at a learning rate of 0.001. In all test instances, DUD beats all comparable deep learning-based character level embedding algorithms in terms of performance and computing cost. Furthermore, character level embedding models-based deep learning architectures outperformed n-gram representation. This is because the embedding records the sequence and interrelationship of all the characters in the URL.

Keywords: Cyber security, Cybercrime, Malicious URL, Machine learning, Deep learning, Character embedding

1. INTRODUCTION

Malicious Uniform Resource Locators (URLs) are frequently used by threat actors to host and transmit malicious content. They are crucial to the success of many cyberattacks. Harmful URLs are commonly spread through email and social media, and are often shared using Facebook, Twitter, WhatsApp, Orkut, and other social media apps. Users understand that the information found here has not been vetted or approved. Hackers can gain access to the hosting platform if an unsuspecting user accesses the website via the URL. The user is then vulnerable to frauds that attempt to steal

information from or even impersonate them. Every year, billions of dollars are lost due to malicious URLs. Now more than ever, we need a means to detect potentially harmful URLs and report them to the administrator of the network in a timely fashion.

Blacklisting is used in many commercially available products today. The foundation of this strategy is a comprehensive collection of potentially harmful URLs. The anti-virus team ensures the accuracy of the blacklists by regularly verifying them and collecting new information from the general public. If a URL is already in the database and it is attempting malicious activity, the blacklisting technique can be used to identify it. However, they are absolutely unable to discover novel harmful URLs or variants of preexisting harmful URLs. Today's cybercriminals utilize a process called "mutation" to create numerous variants of the same malware. This issue is addressed by applying machine learning techniques.

The most popular method in recent years for extracting lexical information from URLs has been to build upon existing domain knowledge using machine learning models. When it comes to machine learning, the Support Vector Machine (SVM) model is frequently employed, whereas the Bag-of-Words (BoW) approach is frequently used when developing features. There are certain issues with utilizing machine learning algorithms, but they may be able to replace blacklists as a solution. One issue is that typical representations of URLs don't adequately convey the structure of the characters included within them.

In conventional machine learning models, feature construction is performed manually. A thorough understanding of the cyber security sector is essential for success in this position, which is widely acknowledged to be extremely challenging.

The system has no way of remembering secret characteristics and is confused by test results. The sheer variety of words also hinders the machine learning system's ability to acquire knowledge, as the system simply cannot store all of them in its limited memory.

This research offers the Deep URL Detect (DUD) model, which addresses the aforementioned issues by fusing character embedding with the cutting-edge machine learning technique known as "deep learning." In order to learn models at varying levels of abstraction, deep learning uses a large number of hidden layers to perform nonlinear projection. This approach has wide-ranging applications in cyber defense. The significant benefits of the proposed project are as follows:

The purpose of this research is to identify malicious URLs by doing a comprehensive analysis and review of several benchmark deep learning architectures.

During the experimental investigation, multiple data sets are employed to assess the versatility of the models. The experimental analysis highlights the distinction between the time-split and random-split approaches to data partitioning.

In this research, we experiment with a variety of deep learning models for both character embedding and n-gram encoding.

2. RELATEDWORK

This paper examines the literature on the use of machine learning techniques to the problem of identifying potentially harmful URLs. This section provides an overview of the primary methods used to identify potentially dangerous URLs. Blacklisting, regular expressions, and signature matching are all common early-stage techniques for locating malicious URLs. So far, we've only discussed techniques that can locate unchanged URLs. The signature library should be continuously updated to account for emerging threats to URL patterns. Then, we applied machine learning strategies to identify fresh clusters of potentially malicious uniform

resource locators (URLs). In conventional machine learning approaches, gathering all of the relevant URL attributes sometimes necessitates the feature engineering process. Feature engineering in cyber security requires a thorough familiarity with the Uniform Resource Locator (URL) and a well-structured collection of high-quality features selected with great care. There have been numerous published studies that employ various criteria to identify and characterize malicious URLs. Content-, popularity-, and context-based qualities, as well as blacklist-, vocabulary-, and host-based traits, are all examples of this category. We can learn more about a piece of brick by looking up its Uniform Resource Locator (URL). This tool has the potential to be an efficient means of tracking down and identifying rogue URLs. Lexical characteristics are evaluated using information about URL strings, such as the quantity of special characters and the length of the URL. The hostname parameters of the URL are employed to facilitate access to the host's native features. This might be anything from the location of the computer to its IP address and WHOIS information. Website content is created using HTML and JavaScript when an unsuspecting user hits a malicious URL. Some of the characteristics of the content are its status, its popularity, and its origin. Experts in the area have hand-picked the trait categories and trait combinations used in the past studies.

Because of the inherent security issues, feature development is arduous work. Obtaining characteristics that depend on context, such as those mentioned above, can be challenging and even dangerous. Domain-specific knowledge and expertise are also needed throughout the feature selection process. The URL's underlying structure was often parsed for this information. Published research shows that learning the lexical property is less challenging than learning other qualities. There were additional term frequency-based techniques, such as the term document matrix (TDM) and the term frequency and inverse document frequency (TF-IDF), and n-grams. The order of URLs or their meaning cannot be determined based on any of these characteristics. In this scenario, we don't pay any attention to what the nameless characters tell us. It's also worth noting that a technique for detecting malicious URLs that relies on feature engineering and regular machine learning can be readily circumvented by an adversary.

These days, we use deep learning techniques that encode a value into each character to identify potentially harmful URLs. The effectiveness of deep learning, character-level embedding, regular machine learning, and feature engineering in detecting malicious and counterfeit URLs was investigated. When compared to more conventional approaches, the results from deep learning systems were superior. Long short-term memory (LSTM) and recurrent neural networks (RNNs) are employed to accomplish the task of identifying forged URLs. A random forest classifier was used to make comparisons based on lexical features and statistical URL analysis. The LSTM model performed better than the standard machine learning approach. In order to identify potentially harmful web addresses, file locations, and registry keys, a convolutional neural network (CNN) with a character-level Keras embedding was employed. This research demonstrated how a novel deep learning framework may be utilized to address numerous issues in the realm of cyber security. Currently available deep learning benchmark frameworks are numerous. The purpose of this research is to compare the performance of several deep learning models in locating URLs.

3. SYSTEM DESIGN

An Overview of Uniform Resource Locator (URL)

A Uniform Resource Locator (URL) is the

component of a Uniform Resource Identifier (URI) that specifies the specific resource that should be located and obtained from an online service. Uniform Resource Locators (URLs) have the three components shown in Figure 1. The first section identifies the protocol used, which can be "http" or "https." In the following section, you'll choose a domain name or an IP address. The final section details the route to the target website and any additional requirements. The protocol can be seen after the domain name, which is preceded by a pair of forward slashes. The path is delineated from its constituent parts by a single forward slash. A sample Uniform Resource Locator (URL) is shown in Figure 1.

Using the URL as the primary source opens the door for an adversary to potentially aid and carry out destructive operations. Email and social media are common vectors for the propagation of malicious URLs. A system containing a malicious Uniform Resource Locator (URL) can be compromised if an unsuspecting user clicks on the URL. Understanding what a URL is and where it should go is therefore considered crucial.Convolutional neural networks (CNN) and long short-term memory (LSTM) come together in the CNN-LSTM model, a deep learning architecture. Keras's letter-by-letter embedding approach.

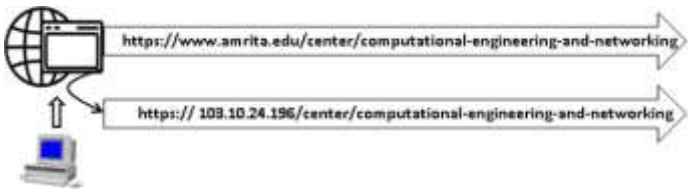


Figure 1: Uniform resource locator (URL) components

Background details of Deep learning Models

Hybrid architecture –

In order to extract features from input, the Convolution Neural Network (CNN) employs the convolutional approach, just like the Deep Neural Network (DNN).

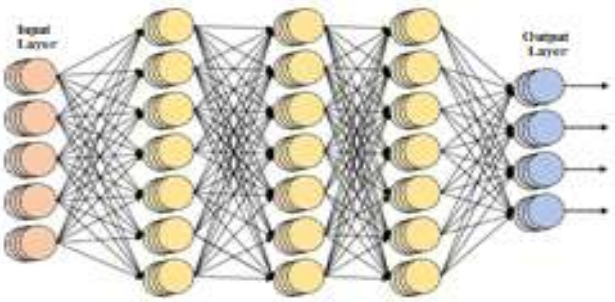


Figure 2: Figures 2 and 3 show some examples of DNN and CNN networks, respectively. CNN's initials shortened to CNN-C. One-dimensional convolution and pooling algorithms, commonly known as temporal convolution and temporal pooling, are frequently used in CNN-C. The CNN-C model selectively applies the best features of character representations of URLs. Keras embedding representation at the character level is employed for this purpose. For this, you'll need to know the length of the character vector, the maximum length of the dictionary, and the length of the embedding vector. The initial values for the character-level embedding weights in Keras can be set via a hyperparameter. The weight is fine-tuned during backpropagation. The LSTM is trained using information from the CNN. This facilitates education in character-level sequence description.

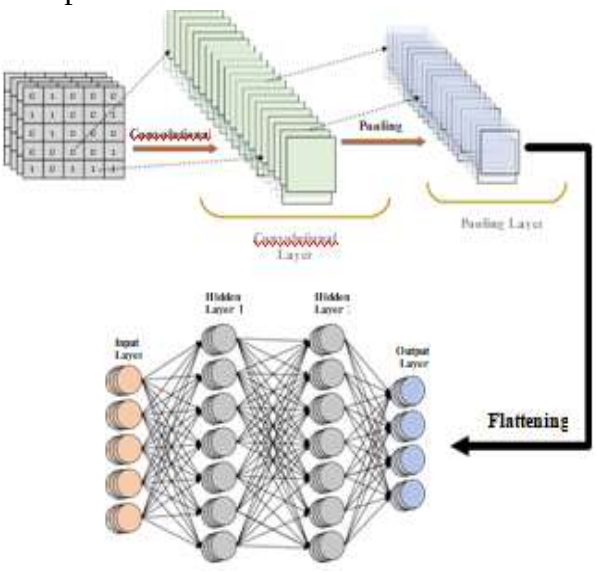


Figure 3: Convolution Neural Network architecture

Character based models

To extract features, character-based models need only be fed sequences of characters. One application of the aforementioned characteristics is text classification. In this research, several character-based NLP models are put to the test in the context of cyber security, with the goal of identifying harmful URLs. Embedding is utilized in the first layer of each model to convert URLs into numerical vectors.

Character level models based on RNN

Endgame Architecture:

In this research, domain names generated by DGAs are searched for and categorized using character-level Keras embedding with LSTM. Keras embedding at the character level makes it simpler to remember and type out domain names in their correct case. Furthermore, it completely ignores feature engineering, a vital stage in more conventional machine learning approaches. The strategy outlined above is superior to the hidden Markov model, feature engineering, and the use of bigrams in conjunction with conventional machine learning classifiers. The LSTM network in this research consists of an embedding layer representing URLs, an LSTM layer for extracting salient features, and a logistic regression layer for categorizing data. To extract features, an LSTM is fed 128-dimensional images of each character generated by the embedding layer. Then, using logistic regression, a probability score is assigned to each domain name.

CMU Architecture:

The architectural framework developed at CMU is known as Tweet2vec. Mainly, it is concerned with the problem of encoding and categorizing social media data, particularly tweets. A bidirectional gated recurrent unit (BGRU) is used to extract feature models from Twitter data. In this technique, tweets are parsed into "tokens," which are collections of characters, and then each token is encoded using a one-hot character encoding approach. The one-time models are converted to a character space and then integrated into the BGRU framework. By incorporating both forward and backward Gated Recurrent Units (GRUs) into the model, the model makes it simpler to comprehend the order of the characters in the domain name. Tweet hashtag predictions are made using the forward and reverse GRU layers of the tweet categorization model. This can be accomplished with a fully linked layer and the softmax non-linear activation function. Word representations on Twitter are used to evaluate tweet2vec for comparative research.

Character level models based on CNN

NYU Architecture:

Convolutional neural networks (CNNs) are currently the standard for this task. In the field of text classification, 1D convolutional neural networks (CNNs) have recently been applied. The specialists in this field used a Convolutional Neural Network (CNN) trained on words along with Long Short-Term Memory (LSTM). The CNN program at NYU has a lot of support there. Pre-trained embedding, embedding, and lookup tables are only a few of CNN's many methods for describing text. Pre-trained word embedding is used to represent text alongside LSTM. Several massive data sets were analyzed. The results showed that compared to both regular and deep learning models, the character level CNN model performed the best. The effectiveness of deep learning models is typically evaluated using more conventional text representation techniques, such as Bag-of-Words (BoW) and n-grams using Term Frequency-Inverse Document Frequency (TF-IDF).

Invincea Architecture:

Short strings of characters, such as Uniform Resource Locators (URLs), file paths, and

registry entries, are ideal candidates for encoding by the CNN network. There are three interconnected levels to the CNN network. There is a Keras embedding layer and a parallel CNN layer for character-level processing. Three fully connected layers, each with 1,024 units, and an activation function based on the Rectified Linear Unit (ReLU). Dropout regularization and batch normalization are employed during model development to accelerate the training process and mitigate overfitting concerns. To determine whether a given sequence of characters is benign or malicious, the Convolutional Neural Network (CNN) employs a fully connected layer consisting of a single unit and a sigmoid non-linear activation function.

Character level models based on hybrid CNN and RNN

MIT Architecture:

The current approach builds upon the NYU model for classifying tweets. The architecture is composed of a long short-term memory (LSTM) layer and layered convolutional neural network (CNN) layers. The deployment of a stacked convolutional neural network (CNN) has been observed to result in overfitting. There are just a handful of tools that aid in this procedure.

Problem Formulation

The purpose of this work is to categorize a given URL as safe or unsafe. Organizing information into meaningful categories is a fundamental challenge. These are some URLs that you should keep in mind: U is a set of tuples, each of which consists of a URL (u) and a label (y) indicating whether or not the URL is dangerous. The value '0' indicates that the URL is good in this case. Finding the best features representation and prediction function are the first and second steps of classification. The feature representation, denoted by xn, can be represented as an n-dimensional vector, and the prediction function can be written as yn = sign(f (xn)). The primary objective is to reduce the frequency with which individuals are inappropriately classified. The loss function can be reduced to accomplish this task. To possibly enhance this loss function, the addition of a "regularization term" may be considered. In this research, we use deep learning models to illustrate the variable f.

Shortcomings in Malicious URL Detection

There are currently no publicly available benchmark datasets for research on malicious URL detection. Most existing literature examines the efficacy of various classic machine learning algorithms and deep learning frameworks in detecting fraudulent URLs; these studies typically make use of proprietary datasets. Data from numerous sources, including Alexa, DMOZ, Phishtank, Open Phish, Malware Domains, Malware Domain-List, and others, were combined to create these secret databases. However, it is essential to keep in mind that these techniques cannot be referred to as "generic procedures" due to the fact that they utilise varying sets of data. Most of the reported outcomes don't explain how the time-split mechanism is employed to partition data into train and test sets. The significance of using a technique called "temporal split" to separate data into training and testing sets was investigated in a recent study. Time-split data splitting is a crucial technique for detecting zero-day malware. Recently, researchers have been trying to understand why machine learning-based security solutions aren't more widely adopted. The various test scenarios that need to be considered in test research are all described in detail. The robustness of machine learning-based solutions may be evaluated with greater ease when a large number of test cases are available. The challenges of using data science techniques to cyber security were also discussed.

4. SYSTEM ANALYSIS

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 9, September : 2023

Model Configuration of Malicious URL Detection Engine

Algorithm 1: Malicious URL Detection Engine

Input: A set of URLs $U_1, U_2, ..., U_n$.

Output: Deep URL Detect Model Labels $y_1, y_2, ..., y_n$ (0: legitimate or 1: Malicious) and BenchMarkModels prediction p.

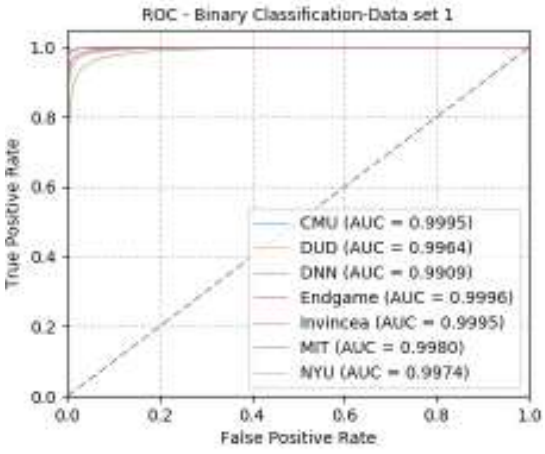
```
VectorizedURLs = Dataprocessing(Ui)  // URLs into numerical vectors using text representation method
Predicions p = BenchMarkModels(VectorizedURL)  // Invincea, NYU, MIT, CMU, Endgame
for each URLs Ui do
    lcURL= lowerCase(Ui)
    ZeroPaddedURL Zi = Padding(lcURL)
    E = Character level Keras Embedding (Zi)
    C = CNN(E)
    L = LSTM(C)
    Compute yi = Sigmoid(L)
end for
```

Table 1: Detailed configuration parameter information of DUD for Data set 1

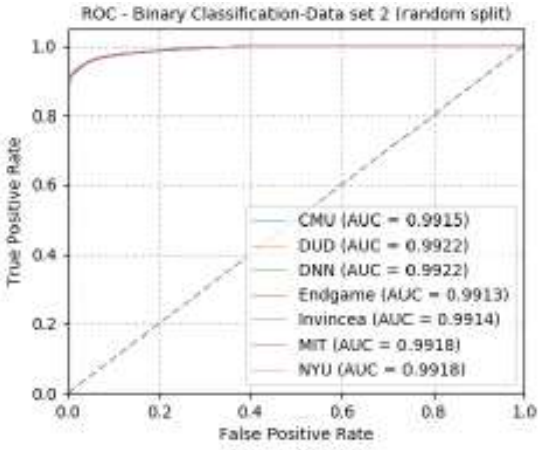
Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 2307, 128)	19200
conv1d_1 (Conv1D)	(None, 2306, 128)	32896
max_pooling1d_1 (MaxPooling1	(None, 1153, 128)	0
lstm_1 (LSTM)	(None, 70)	55720
dense_1 (Dense)	(None, 1)	71
activation_1 (Activation)	(None, 1)	0
Total params: 107,887 Trainable params: 107,887 Non-trainable params: 0		

Table 2: Detailed configuration parameter information of DUD for Data set 2 random-split and time-split

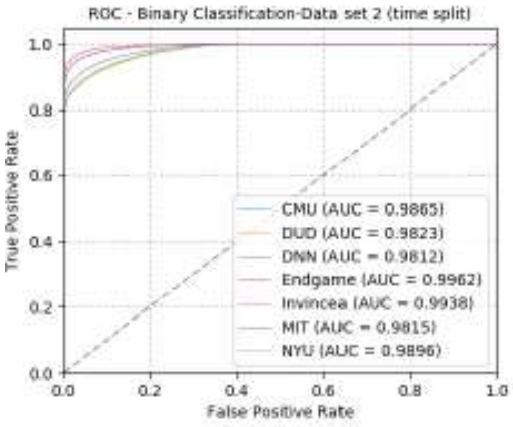
Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 246, 128)	5376
conv1d_1 (Conv1D)	(None, 245, 128)	32896
max_pooling1d_1 (MaxPooling1	(None, 122, 128)	0
lstm_1 (LSTM)	(None, 70)	55720
dense_1 (Dense)	(None, 1)	71
activation_1 (Activation)	(None, 1)	0
Total params: 94,063 Trainable params: 94,063 Non-trainable params: 0		



(a)



(b)



(c)

Figure 4: ROC curve for (a) Data set 1, (b) Data set 2 (random-split) (c) Data set 2 (time-split)

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 9, September : 2023

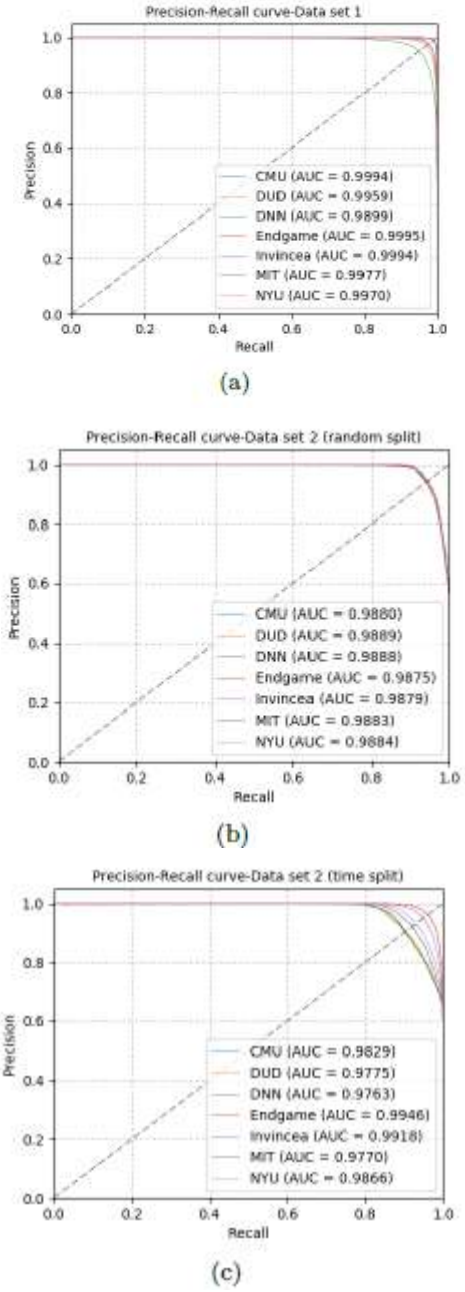


Figure 5: ROC curve for (a) Data set 1, (b) Data set 2 (random-split) (c) Data set 2 (time-split)

5. CONCLUSION

In order to identify phony URLs, this research examines and contrasts character-level embedding models built with deep learning. The effectiveness of various deep learning architectures is very consistent. Out of the five models, two are based on Recurrent Neural Networks (RNN), two on Convolutional Neural Networks (CNN), and one is a hybrid that combines elements of RNN and CNN architectures. All of the models performed admirably, properly identifying harmful URLs between 93% and 98% of the time with a false positive rate of only 0.001. A comparison study is conducted using deep neural networks (DNN) and n-grams. Deep learning-based character models have consistently outperformed other models across all evaluation metrics. Using character-level embedding, deep learning techniques may effectively deal with a broad variety of variations in malicious URLs. Despite the progress made by deep learning, there are situations in which more conventional approaches might be preferable to character-level embedding models trained with deep learning. Blacklists based on regular expressions, matching signatures, and conventional machine learning methods are examples of such conventional approaches. The DeepURLDetect (DUD) paradigm can be improved by include other components such as registrar services, website content, network reputation, file paths, and registry keys. In my opinion, this road is crucial to the success of any future endeavors.

REFERENCES

[1] Tran, K. N., Alazab, M., & Broadhurst, R. (2013, November). Towards a feature rich model for predicting spam emails containing malicious attachments and urls. In 11th Australasian Data Mining Conference, Canberra.

[2] Alazab, M., & Broadhurst, R. (2015). Spam and criminal activity.

[3] Alazab, M., Layton, R., Broadhurst, R., & Bouhours, B. (2013, November). Malicious spam emails developments and authorship attribution. In Cybercrime and Trustworthy Computing Workshop (CTC), 2013 Fourth (pp. 58-68). IEEE.

[4] Broadhurst, R., Grabosky, P., Alazab, M., Bouhours, B., & Chon, S. (2014). An

analysis of the nature of groups engaged in cyber crime.

[5] Alazab, M., Venkatraman, S., Watters, P., & Alazab, M. (2011, December). Zero-day malware detection based on supervised learning algorithms of API call signatures. In Proceedings of the Ninth Australasian Data Mining Conference-Volume 121 (pp. 171-182). Australian Computer Society, Inc..

[6] Vinayakumar, R., Alazab, M., Srinivasan, S., Pham, Q. V., Padannayil, S. K., & Simran,
K. (2020). A Visualized Botnet Detection System based Deep Learning for the Internetof Things Networks of Smart Cities. IEEE Transactions on Industry Applications.

[7] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. IEEE Access, 7, 41525-41550.

[8] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., & Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. IEEE Access, 7,46717-46738.

[9] Vinayakumar, R., Alazab, M., Jolfaei, A., Soman, K. P., & Poornachandran, P. (2019, May). Ransomware triage using deep learning: twitter as a case study. In 2019 Cybersecurity and Cyberforensics Conference (CCC) (pp. 67-73). IEEE.

[10] Srinivasan, S., Ravi, V., Sowmya, V., Krichen, M., Noureddine, D. B., Anivilla, S., & Kp, S. (2020, March). Deep convolutional neural network based image spam classification. In 2020 6th Conference on Data Science and Machine Learning Applications (CDMA) (pp. 112-117). IEEE.

[11] Sahoo, D., Liu, C., & Hoi, S. C. (2017). Malicious URL detection using machine learning: A survey. arXiv preprint arXiv:1701.07179.

[12] Felegyhazi, M., Kreibich, C., & Paxson, V. (2010). On the Potential of Proactive Domain Blacklisting. LEET, 10, 6-6.

[13] Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009, June). Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1245-1254). ACM.

[14] Kolari, P., Finin, T., & Joshi, A. (2006, March). SVMs for the Blogosphere: Blog Iden- tification and Splog Detection. In AAAI spring symposium: Computational approaches to analyzing weblogs (pp. 92-99).

[15] Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009, June). Identifying suspicious URLs: an application of large-scale online learning. In Proceedings of the 26th annual international conference on machine learning (pp. 681-688). ACM.

[16] Chiba, D., Tobe, K., Mori, T., & Goto, S. (2012, July). Detecting malicious web- sites by learning IP address features. In Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on (pp. 29-39). IEEE.

[17] Chiba, D., Tobe, K., Mori, T., & Goto, S. (2012, July). Detecting malicious web- sites by learning IP address features. In Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on (pp. 29-39). IEEE.

[18] McGrath, D. K., & Gupta, M. (2008). Behind Phishing: An Examination of Phisher Modi Operandi. LEET, 8, 4.

[19] Cao, J., Li, Q., Ji, Y., He, Y., & Guo, D. (2016). Detection of forwarding-based malicious urls in online social networks. International Journal of Parallel Programming, 44(1), 163-180.

[20] Choi, H., Zhu, B. B., & Lee, H. (2011). Detecting Malicious Web Links and Identifying Their Attack Types. WebApps, 11, 11-11.