ScaleNet: Scalable and Hybrid Framework for Cyber Threat Situational Awareness Based on DNS, URL, and Email Data Analysis

R. Vinayakumar*, K. P. Soman, Prabaharan Poornachandran, Vysakh S. Mohan and Amara Dinesh Kumar

Center for Computational Engineering and Networking (CEN), Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India
E-mail: vinayakumarr77@gmail.com
**Corresponding Author*

Received 13 October 2018; Accepted 15 October 2018;
Publication 06 November 2018

Abstract

A computer virus or malware is a computer program, but with the purpose of causing harm to the system. This year has witnessed the rise of malware and the loss caused by them is high. Cyber criminals have continually advancing their methods of attack. The existing methodologies to detect the existence of such malicious programs and to prevent them from executing are static, dynamic and hybrid analysis. These approaches are adopted by anti-malware products. The conventional methods of were only efficient till a certain extent. They are incompetent in labeling the malware because of the time taken to reverse engineer the malware to generate a signature. When the signature becomes available, there is a high chance that a significant amount of damage might have occurred. However, there is a chance of detecting the malicious activities quickly by analyzing the events of DNS logs, Emails, and URLs. As these unstructured raw data contains rich source of information, we explore how the large volume of data can be leveraged to create cyber intelligent situational awareness to mitigate advanced cyber threats.

Journal of Cyber Security and Mobility, Vol. 8_2, 189–240. *River Publishers*
doi: 10.13052/jcsm2245-1439.823

Deep learning is a machine learning technique largely used by researchers in recent days. It avoids feature engineering which served as a critical step for conventional machine learning algorithms. It can be used along with the existing automation methods such as rule and heuristics based and machine learning techniques. This work takes the advantage of deep learning architectures to classify and correlate malicious activities that are perceived from the various sources such as DNS, Email, and URLs. Unlike conventional machine learning approaches, deep learning architectures don't follow any feature engineering and feature representation methods. They can extract optimal features by themselves. Still, additional domain level features can be defined for deep learning methods in NLP tasks to enhance the performance. The cyber security events considered in this study are surrounded by texts. To convert text to real valued vectors, various natural language processing and text mining methods are incorporated. To our knowledge, this is the first attempt, a framework that can analyze and correlate the events of DNS, Email, and URLs at scale to provide situational awareness against malicious activities. The developed framework is highly scalable and capable of detecting the malicious activities in near real time. Moreover, the framework can be easily extended to handle large volume of other cyber security events by adding additional resources. These characteristics have made the proposed framework stand out from any other system of similar kind.

Keywords: cyber security, natural language processing, text mining, machine learning, neural networks, deep learning, big data, cognitive security, distributed and semantic word representation, domain generation algorithms, uniform resource locator, spam, ransomware.

1 Introduction

With the rapid advancement of Internet, its services and applications, human life has moderately transformed into a cyber space. This is virtual for both individuals and organizations. Cyber security plays a major role in the ongoing development of technology, applications and Internet services. Threat actors have been continually advancing their methods to cope up with firewalls, antimalware systems, and intrusion detection and prevention mechanisms etc. It has been reported that the private companies in US have faced cybercrimes cost of around $100 billion and will be upsurge to over $2 trillion world-wide [1]. Cyber security is composed of a set of tools and techniques that can be used to preserve computers and its networks. It has been evolving, expanding

steadily and swiftly due to the ever increasing demands of information and communications technologies (ICT) systems day by day. It is one of the fastest growing and a highly complex cross-disciplinary field including a manifold set of domains such as natural language processing, image processing, complex systems, mathematics and other domains. The quantity of data produced by security artifacts from different sensors has been increasing at an exponential rate in the last years and will retain to achieve this in the foreseeable future. Thus, the cyber security has invaded the era of big data to handle large amount of data. To provide cyber situational awareness, the large volume of data has to be processed.

Self-learning system is an important component for an organization to analyze large amount of data, respond to attacks and security incidents and to improve from an experience. This generally aims to identify the patterns of malicious and non-malicious activities. Conventional security solutions which are existing in markets are based on symbolic, signature driven and heuristic based security systems where much of intelligence was not expected from the system. Today's security systems which are existing in markets are capable of detecting the yesterday's malware. Both the methods are relying on the historical events and follows static learning approaches that hinder the self-learning capability of the system. Symbolic and signature based detection system depends on the pre-known signatures which exists in signature data base, data base has to be updated continuously by domain experts to cope up with the new attacks. Heuristic approaches follow behavioral analysis to extract optimal features from various high dimensional data sets which can be used for classifying a scenario as an attack or normal with optimal resource consumption. These solutions completely fail at detecting variants of existing known attack or completely a new attack itself. Moreover, these solutions are not typically considered as a self-learning system. These are no longer enough to meet the demand of today's evolving cyber threat landscape. Application of machine learning techniques can detect these existing or entirely morphed and mutated attacks. These techniques have found enormous application in the context of various cyber security use cases. In the last decade, machine learning applications are leveraged towards various cyber security use cases such as botnet detection, malicious URL detection, spam detection, intrusion detection, malware detection and many others [2]. These solutions have the capability to meet the demands of advanced cyber security objectives, cybercrime detection and prevention. With recent technological advancement, deep neural networks a.k.a deep learning typically complex model of conventional machine learning have advanced. Deep learning architectures

showed significant improvement in comparison to the conventional machine learning techniques in various long standing artificial intelligence tasks exist in various domains such as natural language processing, speech processing, image processing and many others [3]. Addition to that, industries have started to deliver deep learning based solutions for highly cognitive tasks[1,2,3]. Despite the fact that the automatic feature learning, as opposed to manual feature engineering has become the de facto standard methodological framework. A security system which make use of conventional machine learning and deep learning architectures are typically called as cognitive security system. A cognitive security system facilitates to automate the process involved in detecting and categorizing the malicious activities [4]. Conventional neural networks such as multi-layer perceptron (MLP) are limited by fixed-size inputs and static input-output relationships, which can be difficult in dynamic system modeling tasks. The conventional neural network assumes that all inputs are independent of each other, and the neurons of each layer of the directed acyclic neural network do not interact with each other, and it is difficult to deal with the problems associated with the input before and after. But many problems in real life are presented in a dynamic system. The present state is often depends on its previous state. Although it is also possible to calculate the relevant content in the time window by dividing a long period of time into a plurality of time windows of the same length, the time window has too many dependencies and changes, and the size is not good. Convolutional neural network (CNN) and recurrent neural networks (RNNs) are two main types of deep learning architecture. CNN has turn out to be the state of the art method in image processing and surpassing human level overall performance. Recurrent structures which include recurrent neural network (RNN), long short-term memory (LSTM), gated recurrent unit (GRU) and bidirectional recurrent structures are used in sequential data modeling problems. One of the commonly used RNNs is LSTM. The difference from the standard RNN is that the calculations function of the hidden layer unit is more complicated, making the memory ability of the RNN stronger. In recent years, we have seen a wave of deep learning techniques for various security use cases [5–20]. These systems can also helpful to solve complex problems related to cyber security in which humans incapable to solve it.

[1] https://www.sophos.com/en-us.aspx

[2] https://www.endgame.com/

[3] http://technicacorp.com/wp-content/uploads/2017/01/WP_Deep-Learning-for-Cybersecurity_111716.pdf

Generally, cyber security system in an enterprise composed of several layers of security. Each layers of security is a separate system which generates numerous log files. These log files contain activities of each user and devices on the network. There is no well-established system which can collect data from various log files, correlate them and identify the patterns of attacks that will be helpful in early detection of advanced persistent threats or any malicious activities. In [21] discussed the challenges involved in applying big data analytics. In [22] showed that the performance in detecting and preventing threats using big data analytics is good in comparison to the conventional methods. Big data technologies can be used to support the integration and mining of security related data sets. Data is collected from many internal and external sources. Big data analytics is employed to analyze DNS log information, network traffic, malicious activities etc. Various forms of data and its larger size facilitate to derive greater value from the data. Big data analytics has the capability to correlate data from different sources and apply analytics on this data to provide classification and prediction options to the end users. Thus the big data based systems are being part of cyber security to handle the complex threats.

In recent days, organization employs multiple layers of security to enlarge the probability of seizing and hindering the malevolent activity. However, the attackers are still able to invade the network and continuously stay undetected in the target system. This is due to the fact that each layer of security system is relying on the rule and heuristics based system. This work introduces the concept of machine learning "recipes" for cyber security use cases. Recipes state that how to configure machine learning tasks, so that we can use automated attack detection to expose elementary attack characteristics that can be difficult to detect using other means. Elementary attack characteristics include DGA, malicious URL and Email. Recipe steps composed of feature engineering, modeling method, comparison, and analysis and result interpretation.

While all the research is focused on the detection of attacks, each work has not demonstrated the ability to scale the detection methods at the scale of Internet. Detection of these attacks at the scale of Internet in real-time is very important to mitigate the attacks before large-scale damage could be done at country or global level [8]. In order to perform the analysis at the scale of Internet in near real-time, there must be scalable algorithm and architecture. The absence of scalable and distributed architectures in solving cyber security related problems motivated the current research to investigate the algorithms and develop a scalable architecture. The algorithms

investigated in this research are deep learning architectures. Many domains have been getting advantageous through the use of big data technologies and deep learning techniques, cyber security is one such domain which is recently leveraging big data technologies to solve many of the real time cyber security use cases.

With the aim to classify the data located at different servers in distributed way, distributed deep learning architectures are used. The objective of this work is set as follows

1. Develop a highly scalable framework, an extended work of [8, 23] which is capable of collecting various security artifacts data of Internet connected hosts of large network and perform web scale data analysis in near real time that provide situational awareness. It has the capability to process billions of events per second in both real time and on demand basis.
2. Develop a cognitive architecture which does a single scale analysis of all various security use cases including but not limited to DNS logs, URLs, and Emails to detect malicious activities and provide an alert notification on detecting malicious activities to the network administrator.
3. In this work, the applications of various advanced NLP techniques are used to significantly amplify the malicious detection rate.

The rest of the parts of the paper are organized as follows. Section 2 discusses scalable architectures; Section 3 provides background knowledge on text representation and deep learning architectures and benchmark character based models on CNN, RNN and CNN-LSTM. Section 4 discusses the shortcomings in cyber security. Section 5 includes problem formulation. Section 6 discusses different cyber security use cases and outlines how conventional machine learning and deep learning can be employed towards various use cases of cyber security. Section 7 discusses the proposed architecture and its components for organizational security. At last, conclusion, future work directions and discussions is placed in Section 8.

2 Scalable Architecture

For the purpose of security analysis, millions of event data can be easily collected from various servers. However, analysis of large volume of data is generally time consuming and storage required become impractical. In order to deal with the large volume of data and the complexity of the event data, the classification of such data must be solved with large number of features,

training examples, or both. Since, these event data are collected from various geographically separated servers, a highly scalable and distributed approach is necessary to solve the classification problem. To handle this, two types of cluster computing framework is developed.

1. Distributed framework on GPUs (NVidia GK110BGL Tesla k40)
2. Distributed framework on CPUs

Due to the confidential nature of research, the detailed configuration details cannot be revealed. Each system has configuration of (32 GB RAM, 2 TB hard disk, Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz) running over 1 Gbps Ethernet network. Apache Spark cluster set up is developed on top of existing Apache Hadoop. The framework provides a scalable design and acts as a distributed monitoring and reporting system.

In real-time, distributed deep learning architectures are used to process extremely large volume of data that result from the system and network events. The architectures are computationally efficient and that can be distributed across multiple systems. These architectures have been studied for various cyber security use cases in the current research work. There are different frameworks exists publically and being used in developing computer vision, speech recognition and natural language processing applications, but are too inefficient for cyber security applications. This is mainly due to that fact that the area of deep learning is practiced by very few researchers and also very few companies are working to deploy real time applications in cyber security. The main reason why cyber security companies don't wish to switch their methods because of the unique challenges has to be met by them. For example, scanning thousands of files per second etc. All conventional machine learning algorithms are developed using Scikit-learn [24] and deep learning architectures are developed using TensorFlow [25].

3 Background

In real-time, it is required to process extremely large volume of data that result from the system and network events. The algorithms that are computationally efficient and that can be distributed across multiple systems have been thoroughly studied for various cyber security use cases in the current research work. The following section describes text representation methods followed to extract features of texts and the algorithms that have been considered for this research work to detect the malicious activities.

Natural language processing (NLP) is a concept-inspired variety of computational strategies for an automated analysis and representation of human language. Be it machine learning or deep learning, raw texts cannot be given as such to them. Before that, somehow those texts have to be converted into numerical representations. Conventional approaches in NLP and text mining suffers from high dimensionality and sparsity issue. Most of the representation disregards the contextual, semantic and syntactic similarity between words. Moreover, they completely fail to preserve the word order. These factors would contribute towards achieving the good performance. Word embedding is a continuous vector representation of words which helps to preserve the contextual, semantic and syntactic similarity and sensitive to word order. It appears clean that no cutting-edge NLP system delivers on the promise of computers understanding language at a human stage.

3.1 Text Representation

Text representation is a process of characterizing the text into numeric form using different forms such as words, characters, sentences etc. There are two types of text representation most commonly used (1) character level (2) word level.

Character level text representation takes an input text as a string of characters and automatically extracts features using machine learning models. These features can be used for performing different tasks for e.g. text classification. There are different character level text representation exist, in this work the efficacy of them are evaluated for cyber security use cases namely DGA domain name detection and malicious and phishing URL detection, otherwise the best character level model will not be known for both DGA domain detection and malicious and phishing URL detection. Word level text representation takes an input text as a string of words and automatically extracts features using machine learning models. These features can be used for performing different tasks for e.g. text classification. There are different word level text representation exist, in this work the efficacy of them are evaluated for cyber security use case namely spam Email detection, otherwise the best word level model will not be known for Email spam detection.

There are two types of text representation namely non-sequential and sequential. These can be applied in both character and word level text representation. Sequential text representation method facilitates to preserve the sequence information while non-sequential representation methods don't preserve the sequence information.

3.1.1 Vector space model

Bag of words (BoW): It is most commonly used method which takes frequency of word/character as a feature. It doesn't preserve the word/character order and syntactic and semantic relationship among word/character representations. To fuse statistics based on external source of information, Term document matrix (TDM) and term frequency-inverse document frequency (tf-idf) were introduced [26]. TDM gives over weights to the frequently occurring words/characters. Most common words like a, an, the, etc. can't be considered as important words towards final stage classification. To avoid this technique tf-idf was introduced. This has the ability to give importance over the rarely occurring words/characters. Both of these methods fail at preserving the context of the words/character. To capture the context of the words/character, the n-gram technique was introduced and it looks the co-occurring word/character sequence of a particular length. This helps to learn the local word/character ordering, for example bigram consider that the words/character that are adjacent to each other. This type representation produces sparsity in text representation and ends up in producing the worse results. To alleviate this, minimum document frequency is considered that use threshold to keep the term in the vocabulary.

3.1.2 Vector space model of semantics

The dimension of the matrix produced for text corpus by vector space model is large. This is generally depends on the vocabulary size. Thus, the classifier has to completely deal with the large number of features for classification. Moreover, all features may not contribute towards classification. To solve this, vector space model of semantics was introduced. This method basically applies matrix factorization approaches on the TDM or tf-idf to decompose the matrix into lower dimension. This type of representation methods typically called as distributional representation [27].

3.1.3 Distributed representation

The representation method of vector space model and vector space model of semantics fails to preserve the word order. Both methods can be together called as non-sequential input representation [5–7]. Thus it ignores the spatial correlation between words/characters. These methods consider the words as discrete and unrelated symbols. The simplest word representation is one-hot representation, which expresses each word/character as a very long vector. The dimension of this vector is the size of the vocabulary, where only one

dimension has a value of 1, and the rest are 0. The dimension represents the current word/character. This representation is very succinct, but it is easy to cause dimensionality disasters and can't describe the relationship between words/characters. Another way to represent it is distributed representation, such as Word2vec. This method represents the word as a dense and low-dimensional real vector. The vector can represent the position of a word in an n-dimensional space, and similar words have similar positions in space. Since the context of the word is used during training, the word vector trained by Word2vec naturally has some syntactic and semantic features. Each dimension of it represents a potential feature of a word and the spatial distance can be used to describe the similarity between words.

1. **Keras embedding (KE):** It is a sequential representation method [5–7]. It forms a dictionary by assigning a unique key for each word/character for the training texts corpus. The size of the dictionary denotes the size of the feature vector. The words/characters are placed in an ascending order in a dictionary *D* based on the frequently occurred word/character statistics. Each word/character of texts is assigned to an index of a dictionary *D*. Word and character vectors are transformed to the same length by choosing the fixed length. The word and character vectors that are too long than the fixed length vector are discarded and zero padding is done to vectors that are short than the fixed length.
2. **FastText (FT):** It is a library developed by the Facebook research community, used primarily to capture the word representations and text classification [28–30]. FastText has the capability to give vector representation for uncommon words. The uncommon words can be broken down into character n-grams and character n-grams are shared with the common words. For example, a pretrained model is trained on news data set, the security terms e.g. phishing can be the rare words. FastText typically character n-grams embeddings method is more efficient than the Word2vec and glove on smaller corpus.
3. **Word embedding (WE):** Word to vector a.k.a Word2vec is a most commonly used tool in natural language processing (NLP) community [31–33]. It learns the context behind words in an unsupervised way. In general, it is similar to single layer neural network. Word embedding represents the continuous representation of words which facilitates to keep syntactic and semantic similarity and more sensitive to word order. Word2vec forms a vector representation for each and every single word. Fast text infers that the word has to be formed by an n-gram

of character. This can also construct the vector representation for the words that do not exist in the vocabulary. Both Word2vec and glove fails to provide vector representation for words that do not exist in the dictionary. Most commonly used methods to create word vectors are skipgram and continuous bag of words (CBOW).

4. **Neural-Bag-of-words (NBOW):** It is a fully connected network which maps an input sequence of words *X* to one of *K* output labels [34, 35]. For each word $w \in X$, a composition function *c* is applied to the sequence of word vectors v_w. The composition function *c* is a vector that is passed to fully connected layer to find out the probabilities for the target label as:

$$z = \frac{1}{X} \sum_{w \in X} v_w \tag{1}$$

$$y1 = soft\ \max(W_l z + b) \tag{2}$$

where W_l is $k \times d$ matrix, *b* is a bias vector and $softmax$ is defined as follows

$$soft\ \max(q) = \frac{\exp q}{\sum_{j=1}^{k} \exp q_j} \tag{3}$$

where *k* is target labels.

3.2 Artificial Neural Networks (ANNs)

An artificial neural network (ANN) is a computational model influenced by the characteristics of biological neural networks. Feed forward neural network (FFN), recurrent neural network (RNN) and convolutional neural network (CNN) are belongs to a family of ANN.

3.2.1 Feed forward neural network (FFN)

Feed forward neural network (FFN) creates a directed graph in which a graph is composed of nodes and edges. FFN passes information along edges from one node to another without formation of a cycle. Multi-layer perceptron (MLP) is a type of FFN that contains 3 or more layers, specifically one input layer, one or more hidden layer and an output layer in which each layer has many neurons, called as units in mathematical notation (as shown in Figure 1). The number of hidden layer is selected through following hyper parameter fine-tuning approach. The information is transformed from one layer to another layer in forward direction without considering the past values. Moreover, neurons in each layer are fully connected. MLP is defined mathematically as $O : R^m \times R^n$

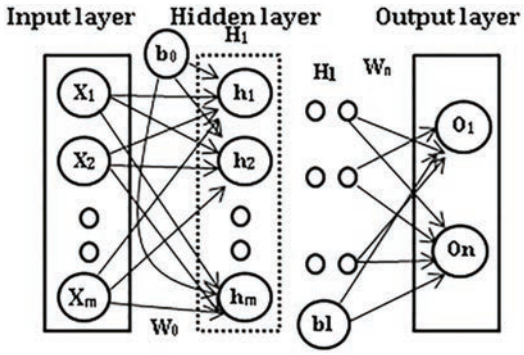

Figure 1 Architecture of Multi-layer perceptron with one hidden layer, all connections are not shown.

where *m* is the size of the input vector $x = x_1, x_2, \cdots, x_{m-1}, x_m$ and *n* is the size of the output vector *O*(*x*) respectively. Each hidden layer h_i computation is mathematically defined as

$$h_i(x) = f(w_i^T x + b_i) \tag{4}$$

where $h_i : R^{d_i - 1} \rightarrow R^{d_i}$, $f : R \rightarrow R$, $w_i \in R^{d \times d_{i-1}}$, $b \in R^{d_i}$, d_i denotes the size of the input, f is non-linear activation function, as either $sigmoid$ (values in the range [0, 1]) or *tangent* function (values in the range [1, –1]). In case of multi-class classification, MLP uses $softmax$ function as non-linear activation function. *Softmax* function outputs the probabilities of each class and selection of largest value among probabilities gives a more crisp value. They are defined mathematically as follows

$$\sigma(y) = \frac{1}{1 + e^{-y}} \tag{5}$$

$$tanh(y) = \frac{e^{2y} - 1}{e^{2y} + 1} \tag{6}$$

$$SM(y)_i = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_i}} \tag{7}$$

Equations (5), (6) and (7) for $sigmoid$, $tanh$ and $softmax$ nonlinear activation function respectively. MLP can be mathematically formulated for *n* hidden layers as given below

$$H(x) = H_n(H_{n-1}(H_{n-2}(\cdots(H_1(x))))) \tag{8}$$

Loss functions, steepest descent or gradient descent and back propagation

As MLP has parameterized functions, finding an optimal parameter is an essential towards achieving good performance. This includes loss function as an initial step. Loss function is used to calculate the amount of difference between the predicted and target values. This is defined mathematically as;

$$d(t,p) = \| \ t - p \ \|_2^2 \tag{9}$$

where *t* denotes the target value and *p* denotes the predicted value.

Multi-class classification uses the negative log probability with the *t* as the target class and *p*(*pd*) as the probability distributions

$$d(t, p(pd)) = -\log p(pd)_t \tag{10}$$

The loss function is defined as below

$$loss(i_n, o_n) = \frac{1}{n}\sum_{i=1}^{n} d(o_i, f(i_i)) \tag{11}$$

This type of loss function will be used in the following discussed deep learning architectures. Loss function has to be minimized to get better results in neural network. A loss functions is defined as

$$Train_{i\_o}(\theta) \equiv L_{i\_o}(\theta) = \frac{1}{n}\sum_{i=1}^{n} d(o_i, f_\theta(i_i)) \tag{12}$$

where $\theta = (w_1, b_1, \cdots, w_n, b_n)$

Loss function minimization $L_{i\_o}(\theta)$ is done by following a right selection of value $\theta \in R^d$ and inherently includes the estimation of $f_\theta(p_i)$ and $\nabla f_\theta(i_i)$ at the cost $|i\_o|$.

$$\min_{\theta} \ L(\theta) \tag{13}$$

There are various optimization techniques exists, Gradient descent is one most commonly used. Gradient descent uses the following rule to calculate and update parameter repeatedly.

$$\theta^{new} = \theta_{old} - \alpha \nabla_\theta L(\theta) \tag{14}$$

where α denotes learning rate and it is selected based on parameter tuning. To find a derivative of *L*, backpropogation or Backward propogation of errors

algorithm is adopted. Backpropogation uses chain rule to compute $\theta \in R^d$ with the aim to minimize the loss function $L_{i\_o}(\theta)$. However, most of the neural network uses an extension of backpropogation called as stochastic gradient descent (SGD) to find minimum θ. SGD uses a mini batch of training samples im_om, in which training samples i_o are chosen randomly instead of using the entire training set ($im\_om \subseteq i\_m$). SGD update rule is given as

$$\theta^{new} = \theta_{old} - \alpha \nabla_\theta J(\theta; im^{(i)}, jm^{(i)}) \tag{15}$$

where $im^{(i)}, jm^{(i)}$ denotes input-output pair training samples.

3.2.2 Recurrent neural network (RNN)

Recurrent neural network (RNN) is an improved architecture of conventional multi-layer perceptron (MLP) [36]. It contains a self-recurrent connection which facilitates to capture and transform the time information across time-steps. This nature of characteristics helps RNN to learn the temporal information. This has obtained good performance in various tasks related to NLP, speech recognition, image processing and others [3]. Generally, RNN takes an input $x = (x_1, x_2, ..., x_T)$ (where $x_t \in R^d$) and transforms to hidden input sequence $h = (h_1, h_2, ..., h_T)$ recurrently with the help of a transition function tf. The hidden state vectors at each time step *t* are estimated as a transition function tf of current input sequence x_t and previous hidden state vector h_{t-1}, as shown below

$$h_t = \left\{ \begin{array}{cc} 0 & t = 0 \\ tf(h_{t-1}, x_t) & otherwise \end{array} \right\} \tag{16}$$

where tf is an affine transformation of x_t and h_{t-1}. This kind of transition function generates vanishing and exploding gradient issue when we propagate the error back in many time steps through back propogation through time (BPTT) in the deep unrolled RNNs network models. To minimize the vanishing and exploding gradient issue, [37–39] introduced LSTM. It contains a memory block that is composed of memory cell and a set of gating functions. A memory cell acts like a container which carries information from one time step to another. The information of a memory cell is controlled by gating functions such as input gate, output gate and forget gate. Additionally, it contains a peephole connection that helps to learn the precise timing of output. Input and output gate controls the flow of input and output and forget facilitates to forget the information of a memory cell. The computation of LSTM units at time step *T* can be generally defined as follows.

$$i_t = \sigma(w_i x_t + U_i h_{t-1} + V_i m_{t-1} + b_i) \tag{17}$$

$$f_t = \sigma(w_f x_t + U_f h_{t-1} + V_f m_{t-1} + b_f) \tag{18}$$

$$o_t = \sigma(w_o x_t + U_o h_{t-1} + V_o m_{t-1} + b_o) \tag{19}$$

$$m1_t = tanh(w_m x_t + U_m h_{t-1} + b_m) \tag{20}$$

$$m_t = f_t \odot m_{t-1} + i_t \odot m1 \tag{21}$$

$$h_t = o_t \odot tanh(m_t) \tag{22}$$

where x_t is the input at time step *t*, σ denotes $sigmoid$ activation function. [40, 41] identity-recurrent neural network (IRNN), [42] gated recurrent unit (GRU) and [43] clock-work RNN (CWRNN) are most important types of RNN. IRNN contains minor changes in comparison to RNN. This has significantly performed well in capturing long-range temporal dependencies. The minor changes are related to initialization tricks; to initialize the appropriate RNNs weight matrix using an identity matrix or its scaled version and use *ReLU* as non-linear activation function. Moreover, this method performance is closer to LSTM in 4 important tasks; two toy problems, language modeling and speech recognition. GRU is a minimal set of LSTM units. CWRNN is a variant to standard RNN architecture in which the hidden layer subdivided into parallel *M* modules. Each such *M* module runs at various clock rates T_m and weight matrices in modules are get updated based on the condition *t* mod $T_m = 0$ across time steps *t* otherwise the previous states are retained.

In addition, hidden layer with many time steps of CWRNN network facilitates to learn both the short term and long term dependencies of the temporal patterns in sequence data. As previously discussed RNNs are unidirectional that focused on modeling the temporal patterns with considering the previous time step context information for output at each time step. Bidirectional RNN is an extension to conventional RNN that model the dependence of both present and future states with forward (start to end of the sequence) and backward direction (end to start of the sequence) [44]. The computation of forward (from *t* = 1 to *T*) and backward pass (from *t* = *T* to 1) in hidden layer and updating output layer (*o*) is mathematically formulated as:

$$\overrightarrow{h}_t = \sigma(w_{x\overrightarrow{h}} x_t + w_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \text{ Forward pass} \tag{23}$$

$$\overleftarrow{h}_t = \sigma(w_{x\overleftarrow{h}} x_t + w_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}) \text{ Backward pass} \tag{24}$$

$$o_t = w_{\overrightarrow{h}o} h_t^{\rightarrow} + w_{\overleftarrow{h}o} h_t^{\leftarrow} + b_o \text{ Output layer} \tag{25}$$

Bidirectional LSTM is formed by combining BRNN with LSTM which helps to capture long range context in both the directions [45].

3.2.3 Convolutional neural network (CNN)

The phenomenal success of convolutional neural network (CNN) in the image domain has inspired many researchers to experiment CNN in various domains. In text domain, in order to best understand the meaning, it is necessary to consider the text as a temporal sequence (i.e. each word occurring in the space of time) rather than considering them as an independent feature [46, 47]. The convolution is seen as the operation of blending of features, it is inferred that convoluted representation has information of nearby words. In NLP, operations are performed over representations given to the text. These representations are given either in the word or character level. In word level, each word is given a vector representation, which is either randomly initialized or assigned through the distributed representations of words. Vector dimension or length of the representations is empirically fixed; this corresponds to the amount of information contained in it. In temporal convolution, the convolution is performed over the temporal sequence. Here the filter dimension must be equal to the vector dimension of the word embedding. So during convolution, filters slides over the words. In convolution operation, over a window of *h* words, a filter *w* is applied to produce new features.

$$o_i = f(w.x_{i:i+h-1} + b) \tag{26}$$

Here, o_i is new feature obtained for the words, $x_{i:i+h-1}$, *b* is the bias term and f is non-linear function. Similarly, features are obtained for all possible word windows as

$$o = \{o_1, o_2, \cdots, o_{i:i+h-1}\} \tag{27}$$

Then max-over time pooling is done over every feature map of all the available filter, which gives $o1 = \max\{o\} \cdot o1$ is the feature of the corresponding filter. The idea of max-over time pooling is to capture the most significant feature in each feature map. The max-over time pooling naturally deals with different length sequences and decreases the temporal dimension. The pooling operation is generally connected to a dense layer and output layer. The number of classes is equal the number of neurons in the output layer. Instead of passing the pooling layer features into fully connected layer, it can also be given to recurrent layer i.e. LSTM to capture the sequence related information across the words sequence of the textual data [48].

3.3 Syntactic Patterns for Identification of Ominous Online Factors (SPOOFNet)

Syntactic Patterns for identification of Ominous Online Factors (SPOOFNet) was proposed by [20] for online scams and cybercrimes detection. This composed of embedding for converting text into numeric vector representation, combination of convolutional neural network and long short term memory (CNN-LSTM) for feature extraction and fully connected layer for classification. An embedding layer used 128 as embedding size, CNN used 128 as filter size, pool length 2. LSTM layer includes 70 memory blocks. To avoid overfitting, dropout of 0.001 was used in between the CNN and LSTM layer. The fully connected layer includes $sigmoid$ activation function to classify the event as malicious or legitimate. The performance of the proposed method was evaluated for DGA and malicious URL detection. This has performed well in comparison to the classical text representation along with classical machine learning algorithms.

4 Shortcomings in Cyber Security

Though large published machine learning based cyberattack detection solutions exists, enterprise companies are still struggling with a contradictory dilemma between selection of machine learning algorithm and benchmark data set. Finding an adequate data set for cyber security use cases is often difficult. Due to privacy and security concern, the security researchers aren't wishing to proportion their data sets to the public for further research. A few data sets exist but these data sets have their own issues. Most common issues are, (1) most of the data sets are out dated (2) they are not real representative data sets. Thus these data sets are not considered as benchmark to check the efficacy of various conventional machine learning and deep learning algorithms. The vast majority of security companies make their data set private and moreover companies do not want to reveal that they have been attacked. These are all the main factors why use cases of cyber security doesn't have generic method and why industries lagging behind in leveraging the machine learning techniques to the development of real time products to detect and classify the malicious activities. There are various research papers on various use cases of cyber security with supervised, unsupervised and semi super- vised exists. Most of the published results have used their own private data sets in evaluating the efficacy of various conventional machine learning algorithms and deep learning architectures. Though, these approaches cannot be regarded

as generic methods due to the data sets are distinct in the methods. The detailed information on why the existing machine learning based solutions can't be deployed in real time is discussed by [49]. Moreover, the problems existing in adopting the data science to the security analytics is briefly discussed by [50]. Thus, these factors have made cyber security as an evolving area of research, new methods will help to detect and alert for malicious activities and advanced persistent threats more accurately in a timely manner.

5 Problem Formulation

Let $c = \{c_1, c_2, c_3, \cdots c_n, 1\}$, here c_1 denotes domain name, c_2 denotes URL and c_3 denotes Email and l denotes label which is 0 (legitimate) or 1 (malicious). The aim is to label the domain name, URL and Email into either legitimate or malicious. It is a supervised learning problem. For each task, initially labeled data is used to train the machine learning model and these models can be deployed in real time systems to detect the malware activities in a timely manner using various data sources such as DGA, URL and Email.

6 Cyber Security Use Cases

In this work, domain generation algorithm (DGA), uniform resource locator (URL) and electronic mail (Email) cyber security use cases are considered. For each use cases, the performance of classical machine learning algorithms and deep learning architectures are evaluated on both the publically available and private data sets.

6.1 DeepDGANet (DDN): Detecting DGA Generated Domain Names Using SPOOFNet

6.1.1 Introduction

Modern malwares use Domain Generation Algorithms (DGAs) which helps to generate pseudo random domains for resilient communication. Their motive is to avoid the blacklisting and evade the intrusion detection systems (IDS). Malware create a rendezvous channel for communication with Command and Control (C&C) servers. Generally techniques like blacklisting and reverse engineering the malware binary code are employed in the DGA detection. But attackers have evolved from using the conventional domain obfuscation techniques to more advanced techniques out of them fast fluxing is the most prominent technique. There are two types of fast fluxing, one is domain fluxing

and other one is IP fluxing. This work is towards domain fluxing, domain fluxing uses DGAs. DGA detection became harder using the conventional manual analysis they consume more time for detection, labor intensive, error prone and less accurate. More over the conventional techniques completely fails to detect the new DGA generated domain.

Due to rapid growth and variance in DGA sophistication and complexity and to overcome the limitations, Machine learning techniques for DGA has recently received substantial attention. Both the unsupervised learning and supervised learning techniques are used for finding the DGA domains. Grouping of DGA domains into clusters in order to determine the statistical attributes and identify patterns is employed in unsupervised learning. DNS server is queried for the domain name resolution to the IP address and when that resolution fails it leads to Non Existent (NX) domains and by identifying the NX domain traffic and analyzing the NX domains can be used for DGA detection. Pleiades [51] DGA detection system uses this unsuccessful domain name resolutions for detecting and classifying the DGA domains. This technique is time consuming and required lot of data and processing for domain clusters for generalization. One more limitation of unsupervised learning is when one or few bots are associated with same DGA in a network then they fail to extract statistical attributes. In supervised learning manually features have to be extracted for training the model. Entropy attribute, Dictionary matching and n-grams are the different features that are commonly used [52]. Classical machine learning techniques needs labor intensive, time consuming and domain expertise for manual feature engineering and selecting the features for DGA is also a challenging task as DGA are highly dynamic and volatile. To overcome these drawbacks recently researchers started using the deep learning algorithms for DGA detection because of the following advantages automatic feature extraction, availability of large amount of data, ability to generalize well, providing high accuracy. [53] has done a detailed analysis of the performance of LSTM on DGA detection and categorization. For comparative study the HMM and bigram text representation with random forest classifier was used. LSTM performed well in all the experiments in compared to the other approaches. [7] performs both DGA binary class classification and multi class classification using different deep learning architectures LSTM, RNN, IRNN, GRU, CNN, CNN-LSTM and also comparing the performance of the mentioned deep learning architectures using the conventional machine learning bigram-logistic regression (LR) algorithm (N-gram based) which clearly showed that deep learning gives significant performance improvements over the machine

learning techniques. [6] Uses the big data approach to detect the DGA in scale for real time by using the apache spark big data platform. Both the binary class and multi class DGA classification is performed using machine learning algorithms (Random Forest, Decision tree and Naïve Bayes) and deep learning architectures (RNN and LSTM) comparison using one training dataset and two diversified test datasets one collected from the publicly available sources and other collected from an internal network. LSTM has outperformed all the other algorithm's both in the binary class and multi class classification tasks. [8] has developed a scalable cyber-threat situational awareness platform for the DNS data analysis in real time and gives alerts and early warnings for threats in the networks. LSTM, GRU, CNN-LSTM, RNN, I-RNN, CNN, Bigram with logistic regression were used with character level embedding layer. LSTM architecture gave highest accuracy because of the ability to remember sequential information. It performs packet level analysis and also time based analysis by capturing the temporal aspects of the DNS network traffic. [54] uses state the art Convolutional Neural Networks (CNN) architectures Alexnet, VGG 16, VGG 19, Squeeze Net, Inception V4, Residual Net for DGA domain classification achieving a very good accuracy but this method involves huge computing and requires high configuration Graphical Processing Unit (GPU) system. [55] uses dataset collected from large amount of real time network traffic compared to previous works which uses less data or data which is not real time and generated synthetically. CNN, LSTM deep learning architectures are used for binary and multi class DGA classification but the resulted accuracy is comparatively less and underperforming compared to the dictionary based methods and they assumed that same DGA generated domains do not repeat for long time. [56] performs a comparative analysis of 5 various benchmark deep learning architectures using the character level embedding and tested on dataset with 100K domain names but lacks the comparison between conventional machine learning algorithms performance with deep learning architectures. [57] Performs a more detailed comparative analysis using HMM (statistical markov model), C4.5 (greedy top down decision tree based algorithm), Extreme Learning machines (ELM), SVM (uses kernel function for classification in higher dimensions using hyper plane), LSTM, Recurrent SVM, CNN+LSTM, Bi-LSTM machine learning algorithms and deep learning architectures using a real world dataset consisting of 1 benign domain class (collected from alexa) and 37 DGA classes performing experiments on multi class classification but lacks binary classification experiments using the same algorithms. Following in this work we evaluate the SPOOFNet architecture for DGA detection and

categorization on both the public and private data sets. This module makes the following contributions

1. A comprehensive analysis of SPOOFNet architecture and compared with the other deep learning architectures and classical machine learning logistic regression with n-gram text representation.
2. This work shows how the DNS data can be correlated with other cyber events to create situational awareness framework.
3. In experimental analysis, we have used both the public and private data sets.

6.1.2 Domain generation algorithms

Domain generation algorithms (DGAs) are the algorithms used by the bots to periodically generate the domain names using various techniques like random permutations, combinations of characters and using dictionary based characters generation. Few of the well-known and as well as most commonly used DGAs are BankPatch, Conficker, Bonnana, Murofet, and Bobax.

The bot residing in the infected machine has to communicate with the C&C server and instead of using a static hard coded domain name which can be easily blacklisted it uses the DGA generated domain name (which is highly dynamic) and queried to DNS server for domain resolution to C&C server IP address. DGA generates a huge number of domains but out of the only few will lead to successful domain resolutions and remaining will be result in NXDomain response from the DNS server as shown in Figure 2.

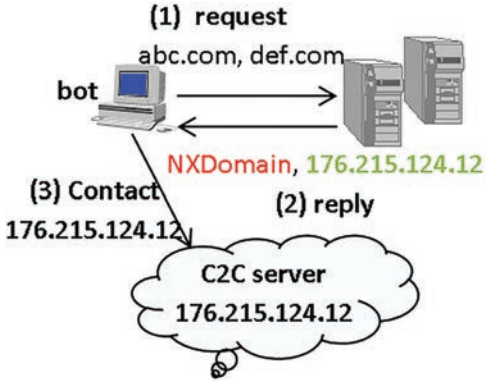

Figure 2 Communication between bot or malware and C2C server.

6.1.3 Description of data set

There are two types are data set are collected. Data set 1 is collected from public sources and Data set 2 is collected from private source [8]. For both the data sets legitimate domain names are collected from Alexa and OpenDNS. Malicious domain names are collected from publically available DGA algorithms, 360-dga, OSINT real time DGA feeds and DGArchieve. Malicious domain names for Data set 2 are collected privately inside CEN Ethernet LAN by following [8]. The detailed statistics of Data set 1 and Data set 2 are reported in Table 1. A sample DNS log is shown in Figure 3.

6.1.4 Proposed architecture

The overview of the proposed architecture is shown in Figure 4. It is typically called as DeepDGANet (DDN). It composed of 3 different sections. They are

1. Embedding
2. Feature extraction
3. Classification

Embedding composed of preprocessing, in preprocessing the characters of domain name are transformed into small letters to avoid over fitting. Otherwise, the architectures might need more computation to learn the significant features. The domain names are transformed into numeric vectors using text representation methods of NLP. These numeric vectors are passed into deep

Table 1 Detailed statistics of data set

Data set	Legitimate	DGA generated	Total
Data set 1 Training	1713121	1010121	2723242
Data set 1 Testing	655683	135056	790739
Data set 2 Testing	7462	112340	119802

Figure 3 DNS log.

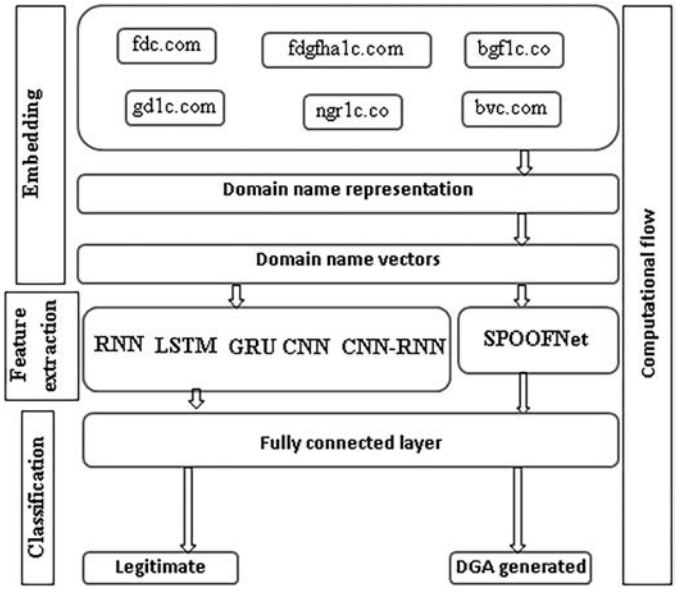

Figure 4 DeepDGANet (DDN).

learning architectures to extract optimal features. Finally these features are passed into fully connected layers for classification. The fully connected layer contains $sigmoid$ nonlinear activation function with binary cross entropy as loss function. $Sigmoid$ activation function outputs 0 or 1, where 0 indicates legitimate and 1 indicates DGA generated. The binary cross entropy is defined mathematically as follows

$$loss(pd, ed) = -\frac{1}{N}\sum_{i=1}^{N}[ed_i \log pd_i + (1 - ed_i)\log(1 - pd_i)] \tag{28}$$

where *pd* is a vector of predicted probability for all samples in testing data set, *ed* is a vector of expected class label, values are either 0 or 1.

6.1.5 Results and observations

All Deep learning architectures and classical machine learning algorithms are trained using the training data set of data set 1. The training data set is randomly divided into 70% and for train and 30% for validation. Validation data helped to monitor the train accuracy over epochs. The performance of the trained models is evaluated on the test data set of Data set 1. In order to know the how trained model is able to generalize well on the unseen samples,

the trained models is evaluated on the Data set 2 testing. In all experiments with both testing data sets, the deep learning architectures performed well in comparison to the classical machine learning algorithms. Moreover, the performances obtained by various deep learning architectures are almost closer in both the testing data sets. Thus voting methodology can be employed to enhance the DGA detection rate. This is remained as one of the significant direction towards future work. The detailed results for test data of Data set 1 and Data set 2 are reported in Tables 2 and 3 respectively. The ROC curve for test data of Data set 1 and Data set 2 is shown in Figures 5 and 6 respectively.

6.1.6 Conclusion, future work and limitations

In this sub module, the SPOOFNet architecture is evaluated for DGA detection. Additionally, the performance of other deep learning architectures, benchmark models and classical machine learning with bigram text representation method are evaluated. The performances obtained by deep learning architectures are closer. Thus voting methodology can be applied to enhance the performance in detecting the DGA generated domain names. This is remained as one of the significant direction towards future work. Deep learning architectures performed well in comparison to the classical machine learning algorithms.

Table 2 Detailed test results on Data set 2

Models	Accuracy	Precision	Recall	F1-score
RNN	0.718	0.999	0.700	0.823
LSTM	0.754	1.000	0.738	0.849
GRU	0.770	0.999	0.755	0.860
CNN	0.721	0.999	0.703	0.825
CNN-RNN	0.789	0.998	0.776	0.873
SPOOFNet	0.807	0.999	0.795	0.885
Logistic regression bigram	0.700	0.999	0.681	0.810

Table 3 Detailed test results on Data set 1

Models	Accuracy	Precision	Recall	F1-score
RNN	0.982	0.921	0.980	0.949
LSTM	0.983	0.932	0.971	0.951
GRU	0.983	0.926	0.976	0.950
CNN	0.982	0.922	0.978	0.949
CNN-RNN	0.984	0.947	0.958	0.953
SPOOFNet	0.984	0.948	0.960	0.954
Logistic regression bigram	0.974	0.948	0.899	0.923

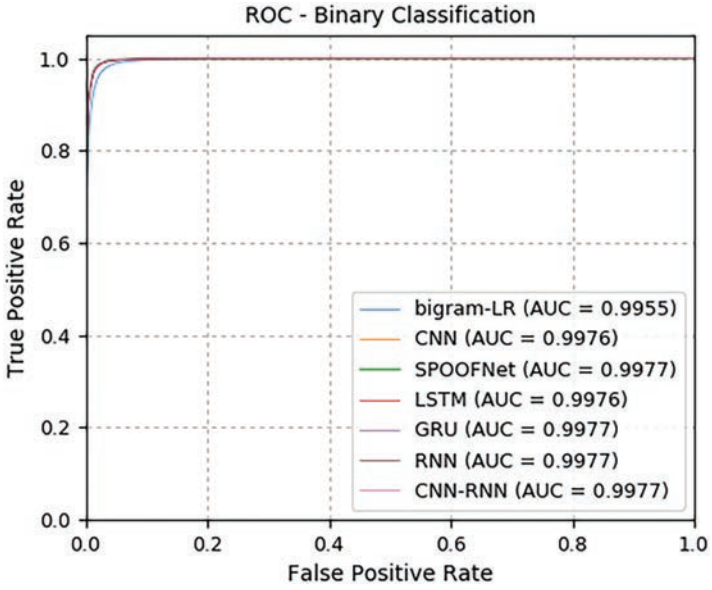

Figure 5 DeepDGANet (DDN).

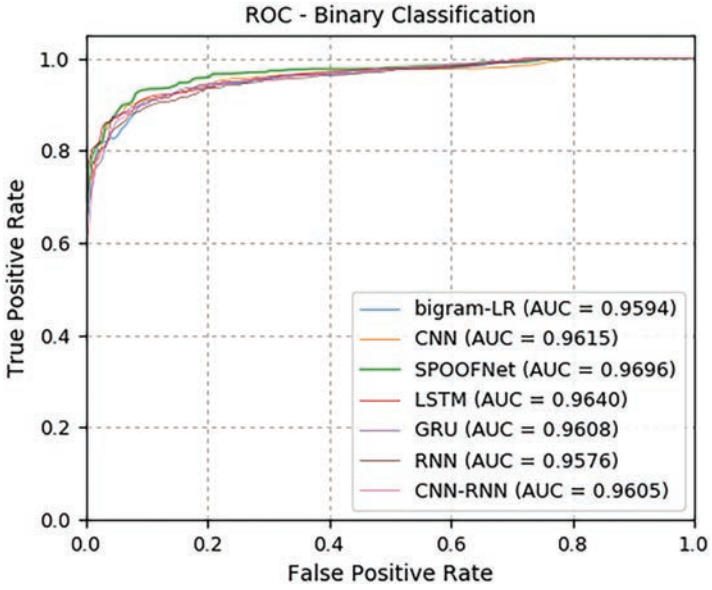

Figure 6 DeepDGANet (DDN).

This is primarily due to the reason that, most of the deep learning architecture has used Keras embedding as DGA representation method. Keras embedding has the facility to capture the sequence information among the characters in the domain name. In recent days, the numbers of DGA malwares are rapidly growing. Most of these malwares are self-similar in nature. Thus DGA categorization has been considered as significant task. This helps to get the detailed statistics of DGA malware family. This is remained as one of the other significant direction towards future work. Additionally, the performance of the proposed deep learning architecture is not evaluated in an adversarial environment. This type of study helps to identify the robustness of the architecture in real time. This can be one of the other future directions towards future work.

6.2 DeepURLNet (DUN): Detecting Malicious URLs Using SPOOFNet

6.2.1 Introduction

The Internet is a global computer network which has enabled people to easily communicate and share information. There is a massive amount of information available on the Internet for just about every field. The application of Internet ranges from personal communication, business transaction, entertainment purpose like web surfing, promotional campaigns, financial transaction, online shopping, and reservations and so on. With the plenty of positive aspects that Internet has to offer, it is also accountable for the security and privacy concerns. The Internet is the source of all the information that is freely available, is being misused such as visiting the unknown sites, Internet theft and unknowingly provides information to the third party. There is a great deal of anonymity to the authenticity of the source through which the information's are exchanged. Recent days, uniform resource locator (URL) is used most commonly to spread malware to end user hosts. Because of the ease of attack and the amount of confidential information that gets available directly from the user made attacker's interested in URL based attacks and many diversified techniques like Phishing (redirecting user to malicious URL which pretends to be original), Drive by download (user un intentional and unaware download of malware just by visiting the malicious URL), spam attacks by exploiting the plugins and browsers vulnerabilities. Blacklisting was the most commonly used method to detect malicious URL, blacklisting completely fails to detect new URLs.

To detect new URLs, machine learning based solutions are used. There are various machine learning based solutions exist and most of them are based on various feature engineering approaches. In [58] discussed various features with classical machine learning classifiers to detect malicious URLs. [59] uses machine learning classifiers for the long and short malicious URL classification. Using features as domain age, link and domain creation difference for long URL's and link creation click lag, referring domains type features for the short URL's. They have used Naive Bayes, Decision tree, Random forest classifiers for only binary class classification of the benign and malicious domains along with providing feature ranks for choosing the optimal features and inferred that referring domains type feature carries more information for classification. [60] Perform a detailed survey regarding the machine learning techniques for malicious URL detection and also proposes a general processing framework using machine learning along with practical challenges and open problems. Feature representation consists of Feature Collection, Feature preprocessing, Blacklisting features, lexical features (both conventional and advanced), Host based features, content based features. [61] Performs a survey on various phishing attacks used in Email filtering. Methods for measuring the effectiveness and the ranking for the features are proposed. Both the supervised learning and unsupervised learning algorithms are applied for the phishing detection. Types of phishing attacks and protecting approaches are discussed along with various machine learning algorithms SVM, K nearest neighbors, Naive Bayes, Boosting with Bag of words and Term frequency inverse document frequency (tf-idf) text representation based techniques are discussed. Architecture for a robust classifier model is proposed. [62] Discusses history and evolution of phishing, total attack incidents reported, Statistics of phishing websites based on domain type and country, phishing life cycle. Also discusses different types of performance evaluation metrics, features of URL, tools and data sets available, Taxonomy of Phishing defense methods, open issues and challenges. These approaches rely on feature engineering. This requires extensive domain knowledge and considered as one of the significant task. In recent days the application of deep learning algorithms are leveraged towards malicious URL detection. These methods performed well in comparison to the classical machine learning methods. This completely avoids feature engineering method where it can obtain optimal features itself.

In [63] proposed a malicious URL detection system using CNN and compares the result with other two algorithms SVM and Logistic Regression (LR). Using a dataset of 75,643 malicious URLs and 3,44,821 benign URLs

accuracy rate of more than 96% is achieved in detecting malicious URLs. In [64] evaluated performance of various different deep learning algorithms and also hybrid network to detect malicious URL. Deep learning algorithms perform better than hand crafted feature mechanism. They achieved a highest accuracy of 0.9996 for LSTM and 0.9995 for hybrid network of CNN and LSTM. In [62] evaluated the performance for classification of both the malicious URL and DNS using the character level embedding with CNN architecture. It proposes a online threat intelligence system which identifies and blacklisting both the malicious URL and DNS on the dataset. It is collected from the public available sources. But lacks the comparison with other machine learning algorithms and deep learning architectures. In [65] proposed a deep learning model for detecting malicious URL from the URL directly which also helps in resolving the problem of detecting the rare words found in the URL. For learning the embedding's of URL it applies CNN to the URL at both words and characters level. Proposed URLNet has around similar accuracy for Word level and Character level URL whereas URLNet has much better performance as compared to other methods. This framework helps URLNet to learn the embedding's for new words found at the testing time. In [66] designed a character level deep neural networks for URL and DNS detection system which automatically extracts the malicious feature by using some Natural Language Processing (NLP) methods which maps the strings of DNS and URL into vector form. Character level deep learning model outperforms the other baseline approaches like Features based and Word level CNN. [67] proposed a method for detecting and categorizing malicious URL. For binary classification it uses deep neural network and feature selection using stacked restricted Boltzmann machine. [68] compares the traditional machine learning algorithm random forest tree with the LSTM deep learning architecture with 3 fold cross validation. 14 features for statistical and lexical analysis of URL's. Clearly RNN getting 5% better performance than the Random forest tree classifier algorithm without requiring labor intensive and time consuming manual feature extraction. In [69] uses gated recurrent neural networks (GRU) for classifying the malicious URL. Performs multi class classification (using 6 classes legitimate, XSS injection, SQL injection, sensitive file attack, directory transversal, other attacks) and compared with machine learning method random forest algorithm by using 21 different features and GRU outperformed the random forest with 4 randomly chosen testing sets. In [70] discuss about how malicious actor may bypass the AI malicious threat detection system by using some AI algorithm. Created an algorithm called Deepphish which learns to create better phishing attacks

using LSTM. Two malicious actors were able to increase their efficiency from 0.69% to 20.9%, and from 4.91% to 36.28%, respectively by using the algorithm on their data by the measurement of percentage of attacks unable to detect by a proactive detection system. Following in this work we evaluate the SPOOFNet architecture for malicious URL detection on both the public and private data sets. This module makes the following contributions.

1. A comprehensive analysis of SPOOFNet architecture and compared with the other deep learning architectures and classical machine learning logistic regression with n-gram text representation.
2. This work shows how the URL data can be correlated with other cyber events to create situational awareness framework.
3. In experimental analysis, we have used both the public and private data sets.

6.2.2 Uniform resource locator (URL)

Uniform Resource Locator (URL) is a subset of Uniform Resource Identifier (URI) which is unique string character identifier used to locate web resources unambiguously across the web, shown in Figure 7. Commonly URL's are web addresses used to locate the web servers for different applications web pages (http), database access, file transfer (FTP) and Email. A typical URL structure looks like http://www.xxxxx.com:xxxx/. It starts with the protocol identifier followed by the domain name along with domain extensions like followed by an optional port number.

URL's can be classified into Static URL and Dynamic URL based on the temporal behavior of URL. Static URL will not change over time and it remains constant without getting updated over time. A static URL looks like http://www.xxx.com/archive/january.htm. Updating can be time consuming because of the more URL's. Dynamic URL are URL's change over time and can be updated from the database, can be personalized based

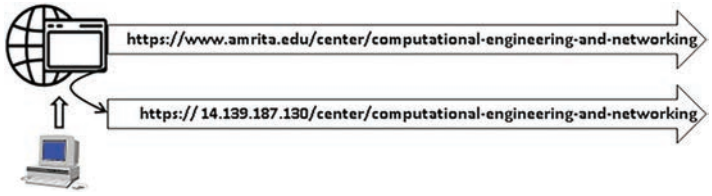

Figure 7 Uniform resource locator (URL).

on the user, no need of updating search indexing, can be changed easily. Widely used ecommerce, content management systems, blogs and forums where content is dynamic and updates frequently. A dynamic URL look like http://xxxx.com/p/xxxxx/issues/detail?id=31. URL's consists of different lengths based on browsers and generally it will be around 2,000 characters. Because of the large size of the URL it becomes difficult for remembering, sharing and entering them so many URL shortening services started to appear which will not only shortens the URL but also obfuscate them. This led attacks redirecting the users to identical malicious domains created by them and steal confidential information like personal details, credentials and bank details.

6.2.3 Description of data set

There are two types data set collected from public sources. Data set 1 is for malicious URL detection and Data set 2 is for phishing URL detection. For both the data sets legitimate URLs are collected from Alexa and DMOZ, malicious URLs for Data set 1 is collected from MalwareDomains and MalwareDomainList and phishing URLs for Data set 2 is collected from Phishtank and OpenPhish. The detailed statistics of Data set 1 and Data set 2 are reported in Table 4.

6.2.4 Proposed architecture

The overview of the proposed architecture is shown in Figure 8. It is typically called as DeepURLNet (DUN). It composed of 3 different sections. They are

1. Embedding
2. Feature extraction
3. Classification

Embedding composed of preprocessing, in preprocessing; the characters of URL are transformed into small letters to avoid over fitting. Otherwise, the architectures might need more computation to learn the significant features. The URLs are transformed into numeric vectors using text representation methods of NLP. These numeric vectors are passed into deep learning architectures to extract optimal features. Finally these features are passed into

Table 4 Detailed statistics of data set

Data set	Legitimate	Malicious URL	Phishing URL	Total
Data set 1 Training	452315	195632	185621	833568
Data set 1 Testing	4000	10000	-	14000
Data set 2 Testing	20000	-	152016	172016

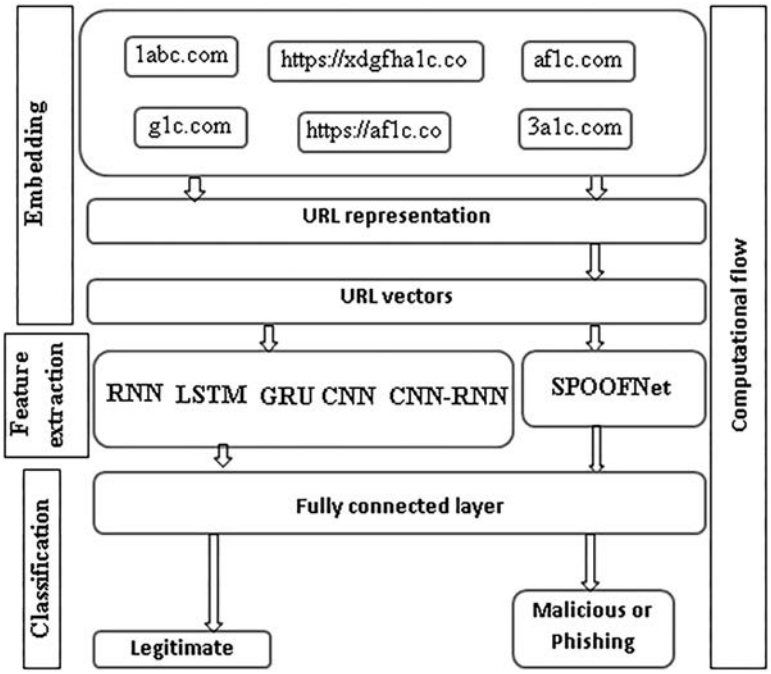

Figure 8 DeepURLNet (DUN).

fully connected layers for classification. The fully connected layer contains $sigmoid$ nonlinear activation function with binary cross entropy as loss function. $Sigmoid$ activation function outputs 0 or 1, where 0 indicates legitimate and 1 indicates malicious or phishing generated. The $sigmoid$ and binary cross entropy is defined mathematically as follows

$$loss(pd, ed) = -\frac{1}{N}\sum_{i=1}^{N}[ed_i \log pd_i + (1 - ed_i)\log(1 - pd_i)] \qquad (29)$$

where *pd* is a vector of predicted probability for all samples in testing data set, *ed* is a vector of expected class label, values are either 0 or 1.

6.2.5 Results and observations

Initially, all deep learning architectures and classical machine learning algorithms are trained using the train data set of Data set 1. Data set 1 is a combination of legitimate, malicious and phishing URLs. The train data of Data set 1 is randomly divided into 70% for train and 30% for valid. Validation

data helped to monitor the train accuracy over epochs. The performance of the trained models is evaluated on the test data set of Data set 1. Data set 1 is followed random split. It has not given any importance to time information. Because, Time split of data facilitates to meet the zero day malware detection. Testing of Data set 2 is done by giving importance to time information. All the samples of testing of Data set 2 are unseen in training data. The detailed results for testing data set of Data set 1 and Data set 2 is reported in Tables 5 and 6 respectively. The ROC curve for the both the data set is shown in Figures 9 and 10. The performance obtained by all deep learning architectures and classical machine learning classifiers on Data set 1 is good in compared to Data set 2. This is due to the Data set 2 is completely unseen during testing. Moreover, the performance obtained by deep learning architecture is good in comparison to the classical machine learning algorithms.

6.2.6 Conclusion, future work and limitations

In this sub module, the SPOOFNet architecture is evaluated for malicious and phishing URL detection. Additionally, the performance of other deep learning architectures and classical machine learning with bigram text representation method are evaluated. The performances obtained by deep learning architectures are closer. Thus voting methodology can be applied to enhance the performance in detecting the malicious and phishing URLs.

Table 5 Detailed test results on Data set 1

Models	Accuracy	Precision	Recall	F1-score
RNN	0.976	0.968	0.999	0.983
LSTM	0.986	0.980	1.000	0.990
GRU	0.984	0.978	1.000	0.989
CNN	0.975	0.966	1.000	0.983
CNN-RNN	0.978	0.970	1.000	0.985
SPOOFNet	0.988	0.984	1.000	0.992
Logistic regression bigram	0.923	0.904	0.999	0.949

Table 6 Detailed test results on Data set 2

Models	Accuracy	Precision	Recall	F1-score
RNN	0.893	0.956	0.921	0.938
LSTM	0.897	0.954	0.928	0.941
GRU	0.894	0.953	0.926	0.939
CNN	0.893	0.954	0.923	0.938
CNN-RNN	0.895	0.952	0.928	0.940
SPOOFNet	0.902	0.950	0.939	0.944
Logistic regression bigram	0.891	0.955	0.920	0.937

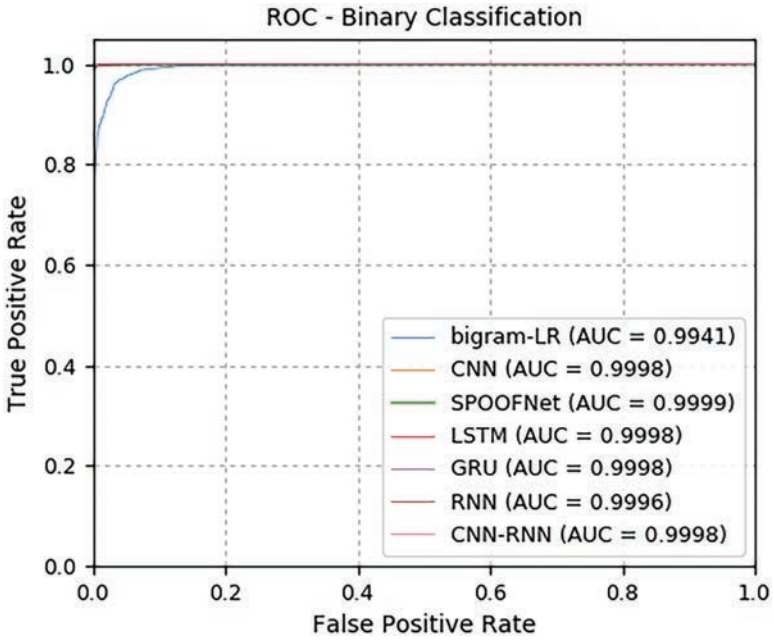

Figure 9 ROC curve for Data set 1.

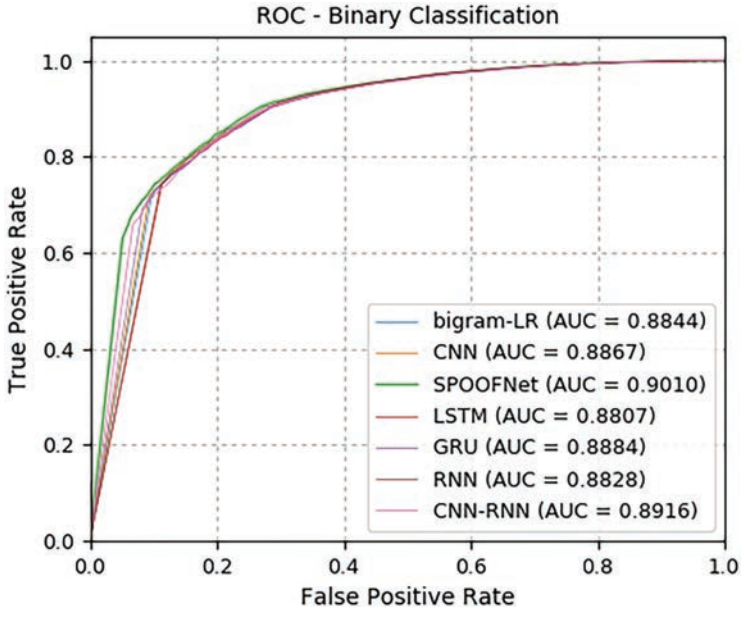

Figure 10 ROC curve for Data set 2.

This is remained as one of the significant direction towards future work. Deep learning architectures performed well in comparison to the classical machine learning algorithms. This is primarily due to the reason that, most of the deep learning architecture has used Keras embedding as DGA representation method. Keras embedding has the facility to capture the sequence information among the characters in the domain name. The performance of the proposed deep learning architecture is not evaluated in an adversarial environment. This type of study helps to identify the robustness of the architecture in real time. This can be one of the other future directions towards future work.

6.3 DeepEmailNet (DEN): Detecting Spam Email Using SPOOFNet

6.3.1 Introduction

Email refers to electronic mail which is one of the effective medium for sharing information. It helps in various sectors such as business, private, personal and government areas to share information without any expense or cost but with high efficiency through Internet. Currently spam Emails end up with big problems over Internet. The hassle of spam Email is increasing every year in a large rate. The recent research clearly shows that spam Emails are sent to Email accounts with massive unwanted data. With spam Email, malwares are passed easily to individual and organizations which leads to criminal activities like stealing information, money etc. Spam Emails troubles various issues such as occupying massive area in inbox, spreading viruses, causing loss of treasured information and many others. Computerized Email filtering is a useful approach to counter the junk mail at the instant. There are various methods such as rule based filter, blacklist, whitelist, checksum database for filtering spam Emails. These methods aren't effective because it passes false positives spam Emails. There are no techniques which give 100% results but when compared to other techniques self-learning system is a better option. Self-learning system makes use of machine learning algorithms which extracts information from large dataset and model the data according to the problem. The machine learning techniques mainly focus on Email spam detection in different ways such as characteristic selection, feature extraction and construction of a classifier. Feature selection is a way of figuring out relevant features from a large dataset. The main aim of this is to take away noise and making the classifier performance more by reducing the computational complexity. For feature extraction many text mining techniques are used such as term frequency–inverse document frequency (tf-idf), bag of words, word

embeeding etc. These methods are largley used in text classification. After feature extraction, the model is been implemented using various machine learning algorithms. Most commonly used conventional machine learning algorithms are Naive Bayes (NB), Decision tree (DT), Random forest (RF), Adaptive Boosting (AB) and Support vector machine (SVM) [71–86].

Various feature selection methods are implemented to detect spam mails. Initially, conventional features such as dictionary based, HTML features, text features and readability features used [87]. In this, text and readability features were obtained using Bag off word (BoW) and tf-idf text representation with multilayer perceptron (MLP) for classification. Navies bayes classifier was a probabilistic classifier which targets the classes and find the instance of model by applying bayes theorem [72]. In [71] discussed how decision tree helped in getting higher accuracy and navie bayes used to lesser the number of features and size. Artificial neural network (ANN) claimed better result when compared to other conventional machine learning classifiers by using back-propagation technique [83]. Meanwhile, deep learning algorithm comes as a breakthrough in identifying spam Emails [84]. This method learns the optimal features from the raw data implicitly. Convolutional neural network (CNN) was used for Email spam detection by extracting the features using word embedding technique in both character and word level [85]. CNN showed good performance in comparison to SVM with feature engineering approach. Comparing conventional machine learning classifiers and deep learning algorithms the performance and accuracy were obtained high when deep learning algorithms, CNN and long short term memory (LSTM) are applied [86].

In this sub module the application of deep learning architectures are applied for Email spam detection. Sequential Email representation i.e. Keras embedding is used to transform Email into numeric vectors. For comparative study non-sequential representation tf-idf with SVM is used. The performance of keras embedding is evaluated with over other advanced text representation word embedding, neural bag of words and FastText. Additionally, this paper presents a new in house model christened DeepEmailNet which uses SPOOFNet architecture which can be used to detect spam emails in a daily email flow.

6.3.2 Description of data set

There are two types of data set is used. The Data set 1 is collected privately and Data set 2 is collected from public source [88, 89]. The Data set 1 contains 35,704 Email samples for training and 13,206 samples for testing. Training

data of Data set 1 composed of 21,442 ham Email samples and 14,262 spam Email samples. Testing data of Data set 1 composed of 7,877 ham Email samples and 5,329 spam Email samples. The Data set 2 contains 48,758 Email samples for training and 20,897 samples for testing. It is shown in Figure 11. Training data of Data set 2 composed of 21,247 ham Email samples and 27,511 spam Email samples. Testing data of Data set 2 composed of 8,993 ham Email samples and 11,904 spam Email samples. Token length versus frequency of Email tokens for Data set 2 is shown in Figure 12.

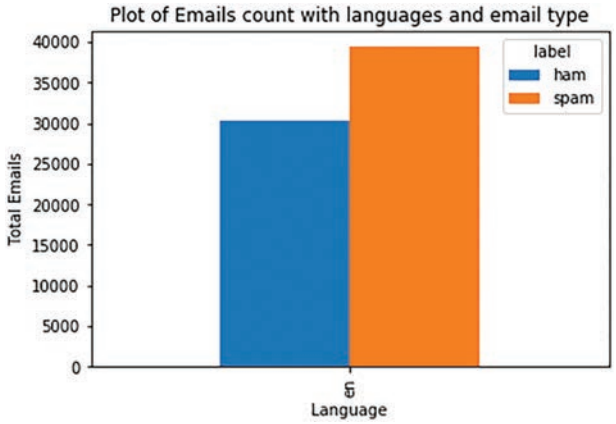

Figure 11 Email samples.

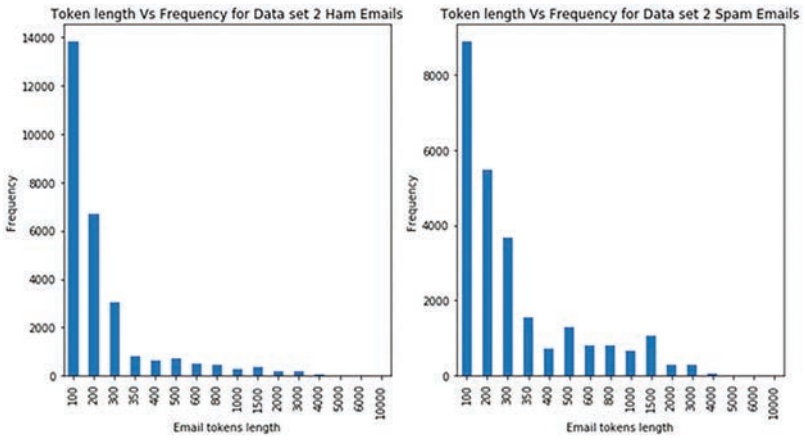

Figure 12 Token length Vs Frequency for Email samples of Data set 2.

6.3.3 Proposed architecture

An overview of proposed deep learning architecture is shown in Figure 13. It is typically called as DeepEmailNet (DEN). It is composed of 3 different sections. The preprocessing and conversion from Email to numeric vectors is done in an input layer. Preprocessing includes converting all capital characters into lower case letters. An Email is transformed into numeric vectors using Keras embedding, word embedding, neural bag of words, FastText and bag-of-words with tf-idf. The optimal features are extracted from the Email vectors and modeled using various deep learning architectures and SVM. All deep learning architectures used fully connected layer as their last layer or output layer and used $sigmoid$ as an activation function with binary cross entropy as loss function. The binary cross entropy loss function is defined mathematically as follows

$$loss(p, e) = -\frac{1}{N}\sum_{i=1}^{N}[e_i \log p_i + (1 - e_i) \log (1 - p_i)] \tag{30}$$

where *p* is a vector of predicted vector for testing data set, *e* is a vector of expected class label, values are either 0 or 1.

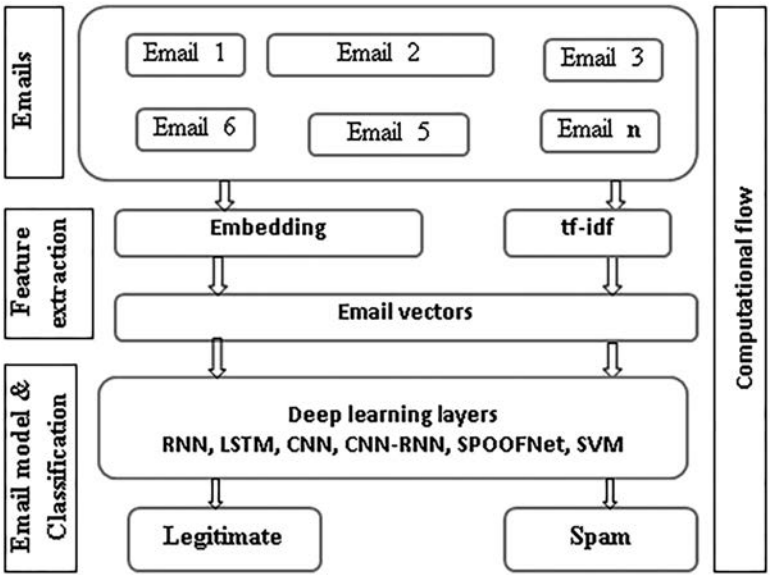

Figure 13 DeepEmailNet (DEN).

6.3.4 Results and observations

To evaluate the efficacy of various deep learning architectures and SVM, various experiments were run for Data set 2. All experiments related to deep learning architectures were run on GPU enabled computer. Hyper parameters for deep learning and SVM models were selected for training set based on running various trials of experiments. RNN and LSTM model composed of a single RNN and LSTM layer with 128 units and memory blocks. Both models followed dropout of 0.4 to reduce the overfitting. Dropout randomly removes the number of neurons along with the connections. Dropout followed fully connected layer for classification. CNN model contains 64 filters with filters length 3. This follows max-pooling with pooling length of 3. The pooling features were passed into 2 fully connected layer for classification. First fully connected layer contains 128 units and followed dropout and again fully connected layer with 1 neuron to classify the Email as spam or ham. Instead of passing the pooling output to fully connected layer, we passed into the RNN and LSTM layer in second set of experiments. These layers collectively work with the CNN and captured the sequential information. These RNN and LSTM layer followed fully connected layer for classification. The detailed statistical measures of all models were reported in Tables 7 and 8. SPOOFNet architecture performed well in comparison to all the other deep learning architectures. Same experiments were done for Data set 2. Various advanced text representation were used to obtain numerical features and followed SPOOFNet architecture for classification. The performances obtained by advanced word embedding models were closer and these models performed well in comparison to the classical, non-sequential text representation methods. The detailed statistical measures were reported in Table 7 for Data set 1 and Table 8 for Data set 2. In Table 8, TP denotes true positive, TN denotes true negative, FP denotes false positive and FN denotes false negative.

Table 7 Detailed test results for Data set 1

Model	Accuracy	Precision	Recall	F1-score
RNN	0.939	0.917	0.987	0.951
LSTM	0.951	0.929	0.994	0.960
CNN	0.948	0.925	0.994	0.958
CNN-RNN	0.949	0.923	0.998	0.959
SPOOFNet	0.953	0.929	0.997	0.962
SVM+tf-idf	0.928	0.901	0.988	0.942

Table 8 Detailed test results for Data set 2

Model	Accuracy	Precision	Recall	F1-score	TP	TN	FP	FN
SPOOFNet	0.960	0.939	0.995	0.966	8278	11790	764	65
Word embedding + SPOOFNet	0.961	0.939	0.996	0.966	8269	11807	773	48
NBOW+ SPOOFNet	0.945	0.932	0.973	0.952	8202	11540	840	315
FastText	0.960	0.938	0.997	0.966	8256	11814	786	41
tf-idf + SPOOFNet	0.729	0.973	0.536	0.691	8867	6357	175	5498
tdm + SPOOFNet	0.704	0.988	0.483	0.649	8974	5728	68	6127
tdm with NMF + SPOOFNet	0.730	0.973	0.540	0.694	8866	6399	176	5456
tf-idf with SVD + SPOOFNet	0.732	0.973	0.542	0.696	8866	6421	176	5434

6.3.5 Conclusion, future work and limitations

This sub module presents DeepEmailNet which uses SPOOFNet to detect spam within the daily Email flow. Various deep learning architectures are evaluated for Email spam detection with the publically available benchmark corpus. For comparative study, classical machine learning technique, SVM with tf-idf as text representation method is used. To preserve the sequential information in an Email, Keras embedding is used with deep learning architectures. Keras embedding facilitates to learn the semantic and contextual similarity of words in an Email. Deep learning architectures with Keras embedding as Email representation method are performed well in comparison to the SVM with tf-idf. The SPOOFNet architecture performed well in comparison to the others. Moreover, the performances of various deep learning architectures are closer.

7 Proposed Architecture: ScaleNet

Instead of relying on one layer security, it is better to have different layers of security stack for an organizational security system. To achieve this, the proposed system ScaleNet, collects data from myriad of sources in a distributed manner. Data gathering from different sources is an important task to secure systems from malicious attack and to provide deep visibility into the

organization threat landscape. In real time, the collected data is parsed and aggregated and then sent to real time and non-real time analysis engine that runs highly scalable distributed deep learning architectures. A deep learning module is an essential part of a cognitive security system so that there is no need to spend expensive feature engineering to extract optimal features. What makes the system cognitive is the ability to perceive the data from sensors and respond accordingly. The detailed proposed architecture is shown in Figure 14. The word embedding and deep learning architectures have the ability to provide context for the data which are collected from internal and external sources.

The best performed model, a combination of convolutional neural network (CNN) and long short-term memory (LSTM) is employed in ScaleNet. The deep learning model is lightweight, real time cyber intelligence and distributed across the organizations infrastructure. Like a brain learns to recognize an object, ScaleNet learns to detect any type of malicious activity. This entirely follows a new method to cyber security that is proactive and predictive.

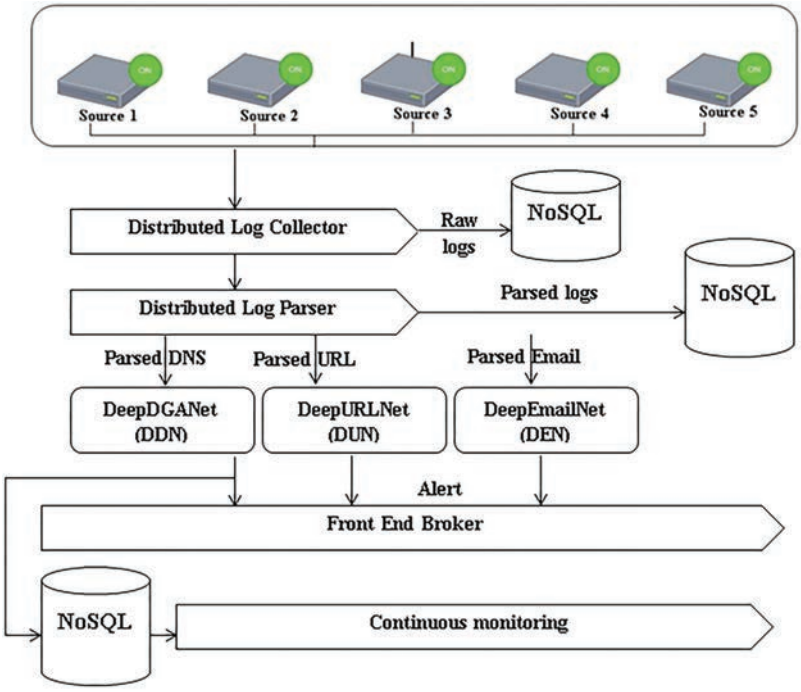

Figure 14 ScaleNet.

This can detect the most evasive unknown malware in real-time across an organization's endpoints, servers and mobile devices. To cope up with entirely new kinds of malicious activity, ScaleNet is continuously trained. Training is done on hundreds of millions of malicious and benign activities. The platform provides protection against cyber threats by learning on its own to distinguish between malicious and legitimate malware. This enables the immediate prediction of malware that has never been seen before. Deep learning follows data agnostic learning process that immediately detects and prevents zero day and advanced persistent threats (APT) attacks across the enterprise's endpoints, servers and mobile devices. The ScaleNet is an autonomous analysis entity that can detect even the tiny mutations and evasion methods in real-time at a pre-execution level. This completely avoids supplementary analysis in a remote server or sandboxing appliance.

8 Conclusion and Future Work

This work proposes ScaleNet, a distributed, highly scalable and unified architecture for an organizational security. It discusses about the data-driven analysis for cyber security use cases and design of the framework. ScaleNet give organizations the situational awareness and decision capabilities required to detect and take proactive security measure from advance threats. ScaleNet integrates all sub modules of cyber security use cases into a unified security solution that makes the platform more robust. The framework provides a scalable design and acts as a distributed monitoring and reporting system. In this work, authors used deep learning approaches to analyze pertinent signals in large amounts of situational awareness data which is beneficial for cyber security. Additionally, the conventional machine learning algorithms are used for comparative study. In most of the cases, deep learning techniques outperformed the conventional machine learning algorithms. This is due to the deep learning techniques have the capability to learn the abstract level feature representation by passing the raw input data to many hidden layers. The nonlinearity in hidden layer facilitates to extract optimal feature representation with respect to the task. Moreover, deep learning architectures have remained as a black box. Thus an adversary may not be able to reverse engineer them easily. In order to defeat the deep learning based detector, an adversary may require the same set of training samples.

The proposed system does not contain a sub module for malware binary analysis. This can provide detailed information of the malware. Thus the proposed framework can be enhanced by adding malware binary analysis.

Most of the organizational security system which are available in online are one-step behind. These systems detect the attacks that had happened already, reconfigure the protection mechanism against the known attacks. The predictive analytics in cyber security facilitates cyber defenders to know the chances of occurrence of cyber attacks from historic and current data. Thus linear and non-linear methods can be employed to forecast events. This is remained as one of significant future direction towards research.

The aim of the work is not only to automate a system which can distinguish the legitimate and benign activities but also scale up and predicting future cyber threats as remained as one of the complex analytics to security challenges. It provides alert and warning capabilities that empower network admin to continuously monitor for over the horizon threats from outside a client's network and control, such as the presence of botnets and other changes in network. The framework shows how different aspects of cyber security logs can be collected, correlated various aspects together and employed machine learning algorithms to identify the patterns from benign and normal activity.

Acknowledgments

This research was supported in part by Paramount Computer Systems and Lakhshya Cyber Security Labs. We are grateful to NVIDIA India, for the GPU hardware support to research grant. We are also grateful to Computational Engineering and Networking (CEN) department for encouraging the research.

References

[1] Smith, S. (2015). Cybercrime will Cost Businesses over $2 Trillion by 2019.-juniper research, 2015. Available at https://www.juniperresearch.com.

[2] Buczak, A. L., and Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2), 1153–1176.

[3] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436.

[4] Cognitivesecurity IBM white paper. Available at https://cognitivesecuritywhitepaper.mybluemix.net/.

[5] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2018). Evaluating deep learning approaches to characterize and classify malicious URL's. *Journal of Intelligent & Fuzzy Systems*, 34(3), 1333–1343.

[6] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2018). Detecting malicious domain names using deep learning approaches at scale. *Journal of Intelligent & Fuzzy Systems*, 34(3), 1355–1367.

[7] Vinayakumar, R., Soman, K. P., Poornachandran, P., and Sachin Kumar, S. (2018). Evaluating deep learning approaches to characterize and classify the DGAs at scale. *Journal of Intelligent & Fuzzy Systems*, 34(3), 1265–1276.

[8] Vinayakumar, R., Poornachandran, P., and Soman, K. P. (2018). Scalable Framework for Cyber Threat Situational Awareness Based on Domain Name Systems Data Analysis. In *Big Data in Engineering Applications* (pp. 113–142). Springer, Singapore.

[9] Vinayakumar, R., Soman, K. P., Poornachandran, P., and Sachin Kumar, S. (2018). Detecting Android malware using long short-term memory (LSTM). *Journal of Intelligent & Fuzzy Systems*, 34(3), 1277–1288.

[10] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Deep encrypted text categorization. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (pp. 364–370). IEEE.

[11] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Secure shell (ssh) traffic analysis with flow based features using shallow and deep networks. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (pp. 2026–2032). IEEE.

[12] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Evaluating shallow and deep networks for secure shell (ssh) traffic analysis. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (pp. 266–274). IEEE.

[13] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Evaluating shallow and deep networks for secure shell (ssh) traffic analysis. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (pp. 266–274). IEEE.

[14] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Applying deep learning approaches for network traffic prediction. In *2017 International Conference on, Advances in Computing, Communications and Informatics (ICACCI)*, (pp. 2353–2358). IEEE.

[15] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Long short-term memory based operation log anomaly detection. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 236–242. IEEE.

[16] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Deep android malware detection and classification. In *2017 International Conference on, Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1677–1683. IEEE.

[17] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Applying convolutional neural network for network intrusion detection. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1222–1228. IEEE.

[18] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Evaluating effectiveness of shallow and deep networks to intrusion detection system. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1282–1289. IEEE.

[19] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). Evaluation of recurrent neural network and its variants for intrusion detection system (ids). *International Journal of Information System Modeling and Design (IJISMD)*, 8(3):43–63.

[20] Mohan, V. S., Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2018). Spoof net: Syntactic patterns for identification of ominous online factors. In *2018 IEEE Security and Privacy Workshops (SPW)*, (pp. 258–263). IEEE.

[21] Labrinidis, A., and Jagadish, H. V. (2012). Challenges and opportunities with big data. *Proceedings of the VLDB Endowment*, 5(12), 2032–2033.

[22] Big data analytics in cyber defense. Ponemon Institute Research Report.

[23] Vinayakumar, R., Soman, K. P., and Poornachandran, P. A deep-dive on machine learning for cybersecurity use cases. In *Machine Learning for Computer and Cyber Security: Principle, Algorithms, and Practices*. CRC press, USA.

[24] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12, 2825–2830.

[25] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436.

[26] Manning, C. D. (2008). Prabhakar Raghavan, and Hinrich Schutze. *Introduction to information retrieval*.

[27] Turney, P. D., and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37, 141–188.

[28] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

[29] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

[30] Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., and Mikolov, T. (2016). FastText. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

[31] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[32] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

[33] Rong, X. (2014). Word2vec parameter learning explained. arXiv preprint *arXiv:1411.2738*.

[34] Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

[35] Iyyer, M., Manjunatha, V., Boyd-Graber, J., and DauméIII, H. (2015). Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Vol. 1, pp. 1681–1691).

[36] Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2), 179–211.

[37] Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.

[38] Gers, F. A., Schmidhuber, J., and Cummins, F. (1999). Learning to forget: Continual prediction with LSTM.

[39] Gers, F. A., Schraudolph, N. N., and Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. *Journal of machine learning research*, 3, 115–143.

[40] Le, Q. V., Jaitly, N., and Hinton, G. E. (2015). A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.
[41] Talathi, S. S., and Vartak, A. (2015). Improving performance of recurrent neural network with relu nonlinearity. *arXiv preprint arXiv:1511.03771*.
[42] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
[43] Koutnik, J., Greff, K., Gomez, F., and Schmidhuber, J. (2014). A clockwork rnn. *arXiv preprint arXiv:1402.3511*.
[44] Schuster, M., and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11), 2673–2681.
[45] Graves, A., and Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5–6), 602–610.
[46] Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.
[47] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems* (pp. 649–657).
[48] Lai, S., Xu, L., Liu, K., and Zhao, J. (2015). Recurrent Convolutional Neural Networks for Text Classification. In *AAAI* (Vol. 333, pp. 2267–2273).
[49] Sommer, R., and Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy (SP)*, (pp. 305–316). IEEE.
[50] Verma, R. (2018). Security Analytics: Adapting Data Science for Security Challenges. In *Proceedings of the Fourth ACM International Workshop on Security and Privacy Analytics* (pp. 40–41). ACM.
[51] Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W., and Dagon, D. (2012). From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. In *USENIX security symposium* (Vol. 12).
[52] Zhauniarovich, Y., Khalil, I., Yu, T., and Dacier, M. (2018). A Survey on Malicious Domains Detection through DNS Data Analysis. arXiv preprint *arXiv:1805.08426*.

[53] Woodbridge, J., Anderson, H. S., Ahuja, A., and Grant, D. (2016). Predicting domain generation algorithms with long short-term memory networks. *arXiv preprint arXiv:1611.00791*.

[54] Feng, Z., Shuo, C., and Xiaochuan, W. (2017). Classification for DGA-Based Malicious Domain Names with Deep Learning Architectures. In *2017 Second International Conference on Applied Mathematics and information technology* (p. 5).

[55] Yu, B., Gray, D. L., Pan, J., De Cock, M., and Nascimento, A. C. (2017). Inline dga detection with deep networks. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, (pp. 683–692). IEEE.

[56] Yu, B., Pan, J., Hu, J., Nascimento, A., and De Cock, M. (2018). Character Level Based Detection of DGA Domain Names.

[57] Mac, H., Tran, D., Tong, V., Nguyen, L. G., and Tran, H. A. (2017). DGA Botnet Detection Using Supervised Learning Methods. In *Proceedings of the Eighth International Symposium on Information and Communication Technology*, (pp. 211–218). ACM.

[58] Ma, J., Saul, L. K., Savage, S., and Voelker, G. M. (2011). Learning to detect malicious urls. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3), 30.

[59] Gupta, N., Aggarwal, A., and Kumaraguru, P. (2014). bit. ly/malicious: Deep dive into short url based e-crime detection. In *2014 APWG Symposium on Electronic Crime Research (eCrime)*, (pp. 14–24). IEEE.

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

[61] Almomani, A., Gupta, B. B., Atawneh, S., Meulenberg, A., and Almomani, E. (2013). A survey of phishing Email filtering techniques. *IEEE communications surveys & tutorials*, 15(4), 2070–2090.

[62] Gupta, B. B., Tewari, A., Jain, A. K., and Agrawal, D. P. (2017). Fighting against phishing attacks: state of the art and future challenges. *Neural Computing and Applications*, 28(12), 3629–3654.

[63] Abdi, F. D., and Wenjuan, L. Malicious Url Detection Using Convolutional Neural Network.

[64] Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2018). Evaluating deep learning approaches to characterize and classify malicious URL's. *Journal of Intelligent & Fuzzy Systems*, 34(3), 1333–1343.

[65] Le, H., Pham, Q., Sahoo, D., and Hoi, S. C. (2018). URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection. *arXiv preprint arXiv:1802.03162*.

[66] Jiang, J., Chen, J., Choo, K. K. R., Liu, C., Liu, K., Yu, M., and Wang, Y. (2017). A Deep Learning Based Online Malicious URL and DNS Detection Scheme. In *International Conference on Security and Privacy in Communication Systems* (pp. 438–448). Springer, Cham.

[67] Selvaganapathy, S., Nivaashini, M., and Natarajan, H. (2018). Deep belief network based detection and categorization of malicious URLs. *Information Security Journal: A Global Perspective*, 27(3), 145–161.

[68] Bahnsen, A. C., Bohorquez, E. C., Villegas, S., Vargas, J., and González, F. A. (2017). Classifying phishing URLs using recurrent neural networks. In *2017 APWG Symposium on Electronic Crime Research (eCrime)*, (pp. 1–8). IEEE.

[69] Zhao, J., Wang, N., Ma, Q., and Cheng, Z. (2018). Classifying Malicious URLs Using Gated Recurrent Neural Networks. In *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing* (pp. 385–394). Springer, Cham.

[70] Bahnsen, A. C., Torroledo, I., Camacho, D., and Villegas, S. (2018). DeepPhish: Simulating Malicious AI. In *2018 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE.

[71] Radhakrishnan, A., and Vaidhehi, V. Email Classification Using Machine Learning Algorithms.

[72] Renuka, D. K., Hamsapriya, T., Chakkaravarthi, M. R., and Surya, P. L. (2011). Spam classification based on supervised learning using machine learning techniques. In *2011 International Conference on Process Automation, Control and Computing (PACC)*, (pp. 1–7). IEEE.

[73] Fdez-Riverola, F., Iglesias, E. L., Díaz, F., Méndez, J. R., and Corchado, J. M. (2007). Applying lazy learning algorithms to tackle concept drift in spam filtering. *Expert Systems with Applications*, 33(1), 36–48.

[74] Almeida, T. A., and Yamakami, A. (2010). . Content-based spam filtering. In *2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. IEEE.

[75] Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, (Vol. 62, pp. 98–105).

[76] Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C. D., and Stamatopoulos, P. (2000). Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. *arXiv preprint cs/0009009*.

[77] Woitaszek, M., Shaaban, M., and Czernikowski, R. (2003). Identifying junk electronic mail in Microsoft outlook with a support vector machine. In *Proceedings 2003 Symposium on Applications and the Internet*, (pp. 166–169). IEEE.

[78] Amayri, O., and Bouguila, N. (2010). A study of spam filtering using support vector machines. *Artificial Intelligence Review*, 34(1), 73–108.

[79] Yeh, C. Y., Wu, C. H., and Doong, S. H. (2005). Effective spam classification based on meta-heuristics. In *2005 IEEE International Conference on Systems, Man and Cybernetics* (Vol. 4, pp. 3872–3877). IEEE.

[80] Toolan, F., and Carthy, J. (2010). Feature selection for spam and phishing detection. In *eCrime Researchers Summit (eCrime)*, (pp. 1–12). IEEE.

[81] Wu, C. H. (2009). Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks. *Expert Systems with Applications*, 36(3), 4321–4330.

[82] Tzortzis, G., and Likas, A. (2007). Deep belief networks for spam filtering. In *19th IEEE International Conference on Tools with Artificial Intelligence*, *ICTAI*, 2, 306–309. IEEE.

[83] Guzella, T. S., and Caminhas, W. M. (2009). A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, 36(7), 10206–10222.

[84] Clark, J., Koprinska, I., and Poon, J. (2003). A neural network based approach to automated e-mail classification. In *International Conference on Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC* (pp. 702–705). IEEE.

[85] Christopher Lennan, Bastian Naber, Jan Reher, and Leon Weber. End-to-end spam classification with neural networks.

[86] Campbell, P. D. (2015). Circuit Riders (Doctoral dissertation, Gordon Conwell Theological Seminary).

[87] Almomani, A., Gupta, B. B., Atawneh, S., Meulenberg, A., and Almomani, E. (2013). A survey of phishing Email filtering techniques. *IEEE communications surveys & tutorials*, 15(4), 2070–2090.

[88] Klimt, B., and Yang, Y. (2004). Introducing the Enron Corpus. In *CEAS*.

[89] Spam assasin, Avilable at https://spamassassin.apache.org.

[90] Yu, B., Pan, J., Hu, J., Nascimento, A., and De Cock, M. (2018). Character Level Based Detection of DGA Domain Names.

[91] Jiang, J., Chen, J., Choo, K. K. R., Liu, C., Liu, K., Yu, M., and Wang, Y. (2017). A Deep Learning Based Online Malicious URL and DNS Detection Scheme. In *International Conference on Security and Privacy in Communication Systems* (pp. 438–448). Springer, Cham.

Biographies

R. Vinayakumar is a Ph.D. student at the Amrita Vishwa Vidyapeetham at Coimbatore since July 2015. He has received his BCA from JSS college of Arts, Commerce and Science, Ooty road, Mysore and MCA degree from Amrita Vishwa Vidyapeetham, Mysore. He has several papers in Machine Learning Applied to cyber security. R. Vinayakumar is currently completing a doctorate in Computer Science at the Amrita Vishwa Vidyapeetham at Coimbatore. His Ph.D. work centers on Application of Machine learning and Deep learning for cyber security and discusses the importance of natural language processing, image processing and big data analytics for cyber security. More details available at https://vinayakumarr.github.io/

K. P. Soman has 25 years of research and teaching experience at Amrita School of Engineering, Coimbatore. He has around 150 publications in national and international journals and conference proceedings. He has organized a series of workshops and summer schools in Advanced signal processing using wavelets, Kernel Methods for pattern classification, Deep learning, and Big-data Analytics for industry and academia. He authored books on “Insight into Wavelets”, “Insight into Data mining”, “Support Vector Machines and Other Kernel Methods” and “Signal and Image processing-the sparse way,” published by Prentice Hall, New Delhi, and Elsevier. More details available at https://nlp.amrita.edu/somankp/

Prabaharan Poornachandran is a professor at Amrita Vishwa Vidyapeetham. He has more than two decades of experience in Computer Science and Security areas. His areas of interests are Malware, Critical Infrastructure security, Complex Binary analysis, AI and Machine Learning.

Vysakh S. Mohan is an MTech student at the Amrita Vishwa Vidyapeetham at Coimbatore since July 2016. His MTech work centers on object detection using deep learning. He is an AI enthusiast and developer at Accubits Technologies Inc, who is actively involved in creating artificial intelligence solutions and has several noted research papers in domains like deep learning, computer vision, cyber security and natural language processing. More details available at https://vysakhsmohan.wixsite.com/vysakhsmohan

Amara Dinesh Kumar is an MTech student at the Amrita Vishwa Vidyapeetham at Coimbatore since July 2017. He is an AI researcher and Cyber Security Enthusiast. More details available at https://sites.google.com/view/dineshkumaramara/home