Deep Learning Framework for Cyber Threat Situational Awareness Based on Email and URL Data Analysis

R. Vinayakumar, K. P. Soman, Prabaharan Poornachandran, S. Akarsh and Mohamed Elhoseny

Abstract Spamming and Phishing attacks are the most common security challenges we face in today's cyber world. The existing methods for the Spam and Phishing detection are based on blacklisting and heuristics technique. These methods require human intervention to update if any new Spam and Phishing activity occurs. Moreover, these are completely inefficient in detecting new Spam and Phishing activities. These techniques can detect malicious activity only after the attack has occurred. Machine learning has the capability to detect new Spam and Phishing activities. This requires extensive domain knowledge for feature learning and feature representation. Deep learning is a method of machine learning which has the capability to extract optimal feature representation from various samples of benign, Spam and Phishing activities by itself. To leverage, this work uses various deep learning architectures for both Spam and Phishing detection with electronic mail (Email) and uniform resource locator (URL) data sources. Because in recent years both Email and URL resources are the most commonly used by the attackers to spread malware. Various datasets are used for conducting experiments with deep learning architectures. For comparative study, classical machine learning algorithms are used. These datasets are collected using public and private data sources. All experiments are run till 1,000 epochs with varied learning rate 0.01–0.5. For comparative study various classical machine learning classifiers are used with domain level feature extraction. For deep learning architectures and classical machine learning algorithms to convert text data into numeric representation various natural language processing text representation methods are used. As far as anyone is concerned, this is the first attempt, a framework that can examine and connect the occasions of Spam and Phishing activities

R. Vinayakumar (✉) · K. P. Soman · S. Akarsh
Center for Computational Engineering and Networking (CEN),
Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India
e-mail: vinayakumarr77@gmail.com

Prabaharan Poornachandran
Centre for Cyber Security Systems and Networks, Amrita School of Engineering,
Amrita Vishwa Vidyapeetham, Amritapuri, India

M. Elhoseny
Department of Information Systems, Faculty of Computer and Information,
Mansoura University, Mansoura, Egypt

A. E. Hassanien and M. Elhoseny (eds.), *Cybersecurity and Secure Information Systems*, Advanced Sciences and Technologies for Security Applications, https://doi.org/10.1007/978-3-030-16837-7_6

from Email and URL sources at scale to give cyber threat situational awareness. The created framework is exceptionally versatile and fit for distinguishing the malicious activities in close constant. In addition, the framework can be effectively reached out to deal with vast volume of other cyber security events by including extra resources. These qualities have made the proposed framework emerge from some other arrangement of comparative kind.

Keywords Spam detection · Phishing detection · Email · URL · Image spam · Machine learning · Deep learning

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 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 [1].

Spamming and Phishing are one of the major challenges in the cyber security since it targets to steal financial and personal information [1]. Spamming is the use of the electronic messaging system to send unwanted messages. The most popular form of Spam being Email Spam commonly referred to as 'junk mail'. Spamming remains economically viable because advertisers have no operating costs besides managing the mailing list, IP range domain names, servers, and infrastructure. Since the barrier to entry is so low, Spammers are numerous, and the volume of unwanted mail has become very high.[1] Besides the fact that Spams are annoying it tends to be dangerous especially if it's part of a Phishing scam. Spam Emails are sent to users in a huge quantity by the Spammers and the cybercriminals, to achieve one or more of the followings

1. They tend to make money from the small percentage of recipients that actually respond to such Emails.
2. Carry out Phishing scams to obtain passwords, credit card numbers, bank account details and more.
3. Infect the recipient's computer with malicious code.

[1] http://theconversation.com/four-email-problems-that-even-titans-of-tech-havent-resolved-37389.

Phishing is a malicious activity or type of social engineering attack often used to steal user data, including login credentials and credit card numbers.[2] It obtains confidential information through fraudulent Emails that appear to be legitimate by the attacker that masquerades as a trusted entity. When the recipient clicks the malicious link, it leads to installation of malware, blocking the part of ransomware attack or leaking sensitive information. Such attacks can have devastating results. For individuals, Phishing includes unauthorized purchase, stealing of funds and identity theft.

There are various techniques to detect Phishing and Spamming such as blacklisting and heuristics [2, 3]. While solutions such as Email/URL blacklisting have been effective to some degree, their reliance on the exact match with the blacklisted entries makes it easy for attackers to evade. Blacklisting is a technique that comes under the category of a list based filter which contains a list of senders who are blacklisted i.e., there IP address and Email address are blocked. However, the main issue with the blacklisting technique is that when a new Email or URL arrives, these filters check if it already exists in the blacklisted record. If not it fails to classify any new malicious Email or domain as illegitimate. Also, it may take a long time to detect these using heuristic techniques to appear on blacklists.

Therefore, machine learning techniques have been used which provide better results than classical blacklisting and heuristic techniques. Support vector machine (SVM) is the most popular classical machine learning based classifier to detect Spam and Phishing Emails. It builds a feature map based on the predefined transformations and train sets. Other classifiers such as K-nearest neighbour (KNN) also used for Spam and Phishing Email filtering where decisions can be made based on the K-nearest train input, samples are chosen using a predefined similarity function. Also, Navie Bayes classifier used which is a simple probabilistic classifier. Boosting technique can also be incorporated which depends on sequential adjustment during each stage of the classification process. To convert email into email vectors, tf-idf and hand crafted feature engineering is used. However, the major disadvantage with classical machine learning algorithms is that it relies on feature engineering [4, 5]. With the selection of the best feature, the accuracy can be increased. However, to achieve that, domain knowledge is required. If the feature engineering is not done correctly, the predictive power of the algorithm decreases. Also, with the classical machine learning algorithms the models can be predicted. Feature extraction is the most time-consuming part of classical machine learning workflow. In recent days, the application of deep learning architectures are leveraged for various cyber security use cases, detection of malicious domain names [6–9], detection of malicious and phishing URL [9, 10], phishing Email detection [11–18], intrusion detection [19–21], traffic analysis [22–24], malware detection [25, 26]. This has the capability to extract the optimal features by itself without relying on the feature engineering. Moreover, deep learning architectures are more robust in an adversarial environment in comparison to classical machine learning classifiers. Therefore, we propose the

[2]https://digitalguardian.com/blog/2017-data-breach-report-finds-phishing-emailattacks-still-potent.

use of deep learning technique which can elevate these shortcomings since with deep learning features will be automatically created by the neural network when it learns. Deep learning shifts the burden of feature design also to the underlying learning system along with classification learning typical of earlier multiple layer neural network learning. The objective of this work is set as follows

1. The authors propose christened DeepSpamPhishNet (DSPN), scalable framework which has the capability to handle a large volume of Spam and Phishing activities data [6, 7]. To analyse the data, big data technique is used [27].
2. The efficacy of classical machine learning and deep learning architectures are evaluated on various data sources.
3. DSPN leverage deep learning architectures, specifically a hybrid in house model convolutional neural network-long short-term memory to automatically detect Spam and Phishing activities and give an alert to the network admin inside an organization.
4. The data storage capacity of the proposed system can be enhanced by simply adding the resource to the distributed architecture.

The rest of the chapter are organized as follows. Section 2 provides background knowledge about Email and URL. Section 3 discusses the related works for Spam and Phishing detection of Email and URL. Section 4 discusses the mathematical details of deep learning architectures and text representation methods of NLP. Section 5 discusses the description of dataset. Section 6 includes experiments, results and observations for Spam and Phishing detection of Email and URL. Section 7 discusses the proposed architecture, DeepSpamPhishNet (DSPN). Conclusion, future work directions and discussions are placed in Sect. 8.

2 Background Knowledge

2.1 Electronic-Mail (Email)

Electronic mail (Email) remains for electronic mail. It is the message dispersed by electronic means among personal computer (PC) clients in a network. An Email will be sent from one client and can be conveyed to many. Email works across computer networks which today is basically the Internet. Some early Email frameworks required the creator and the beneficiary to both is online in the meantime, in a similar manner as texting. The present Email frameworks depend on a store-and-forward model. Email servers accept, forward, convey, and store messages. Neither the clients nor their PCs are required to be online all the while; they have to associate just quickly, regularly to a mail server or a webmail interface, for whatever length of time that it takes to send or get messages. There is a standard structure for messages. Email substances are fundamentally delegated to the header and the body. The Email header gives us normal insights about the message. The details of the clients of the 'from' and 'to' closes are likewise stored here. The Email header comprises of the accompanying parts.

- Subject
- Sender (From :)
- Date and Time (On)
- Reply-to
- Recipient (To :)
- Recipient Email address
- Attachments.

In the body part the genuine content is stored. This will be in the format of content. This field could likewise incorporate signatures or content produced naturally by the sender's Email framework.

2.2 Uniform Resource Locator (URL)

A uniform resource locator (URL) which is a subnet of Uniform Resource Identifier (URI) can be used to find the location of resources from a computer network. A URL consists of two parts. The first part defines the type of protocol, for example, http, https or others and the second part defines the location of resources through domain name or IP address.

In Fig. 1, the first part https denotes the protocol; "amrita.edu" is a primary domain name, www.amrita.edu denotes hostname, center/computational-engineering-networking defines the path to a particular resource specifically a webpage on the domain name and edu is a top-level domain name. In recent days, URL is most commonly used tool to spread malicious and phishing activities. Most of the time a user by themselves is not known whether the URL belongs to either benign or malicious or phishing. Thus unsuspecting users visits the websites through the URL presented in Email, web search results and others. Once the URL is compromised,

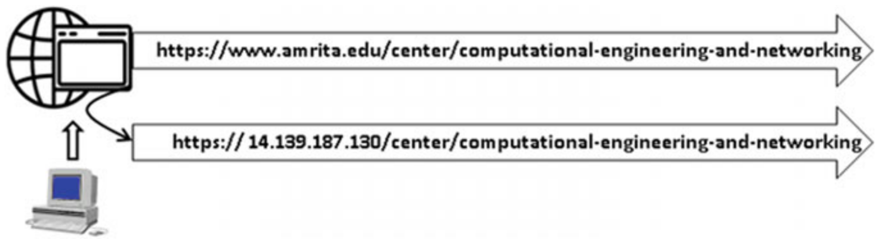

https://www.amrita.edu/center/computational-engineering-and-networking
https:// 14.139.187.130/center/computational-engineering-and-networking

Fig. 1 Example of a uniform resource locator

an attacker imposes an attack. These compromised URLs are typically termed as malicious URLs. As a security mechanism, finding the nature of a particular URL using the necessary mechanism will alleviate the aforementioned discussed attacks.

3 Related Works

This section discusses the related works of Spam and Phishing detection using Email and URL data sources in detail.

3.1 Related Work on Spam and Phishing Email Detection

The detailed survey on existing solutions for email spam detection is reported in [2]. Various feature engineering methods were followed and various classical machine learning algorithms were used for classification. In [28, 29] reported the neural networks have performed well in comparison to the classical machine learning classifiers. Recently, [30] discussed the importance of deep learning architectures for email spam detection over classical machine learning classifiers. They used convolutional neural network with character and word level Keras embedding as deep learning architecture and Support vector machine with one hot encoding text representation. Significant improvement in accuracy was obtained for Spam classification based on convolutional neural network using word embedding method. Research has shown that long short-term memorys and convolutional neural network performance relatively better [31]. Convolutional neural network seems to be the best working algorithm with F1-score 84.0%. Gated recurrent unit-CRF can be used to encode lines using convolutional neural network to predict a sequence of zone types per line reaching accuracies of 98%. With two strategies connected to the binary text classification issue that is Spam filtering, the Support vector machine turned out to be significantly more successful than the Navie Bayes algorithm in halting unsolicited Emails [32]. A framework to detect Phishing through Emails is discussed in [33] which can protect a user from being exposed to phish. Following, in this work the application of various deep learning architectures are evaluated for email spam detection.

The detailed survey on phishing email detection is done by Almomani et al. [3]. Phishing attacks can be classified as deceptive Phishing and malware based Phishing. There are various tools available commercially that operate on the client side such as SpoofGuard, NetCraft, CallingID, CloudMark, eBay toolbar and internet explorer Phishing filter. These tools also rely upon blacklisting and whitelisting, which is a technique used to prevent Phishing assaults by checking Web addresses embedded in Emails or by checking the site specifically. In the blacklisting process, at a standard interval of time, Phishing websites are detected and are updated to user machine by the search engines or users. However, blacklisting requires time for new Phishing sites to be accounted for and blacklisted. Whitelisting is a gathering of "good" URL contrasted with outside connections in receiving incoming Emails. It appears more promising, however, creating a list of reliable sources is tedious and it is a tremendous task. The two major problems that are encountered in blacklisting technique is that a high number of false positives are produced which allows the phish Email to get through also the ham Emails are getting filtered. Therefore, whitelisting techniques are also not effective enough for detecting Phishing attacks.

Phishing Emails have put an average computer user at risk of personal and financial data loss especially since it have become active than ever before. Hamid et al. [34] proposes an approach for feature selection which is a combination of both content based and behavior based i.e., a Hybrid feature selection approach. Based on Email header, the approach can be used to mine the attacker behavior. An accuracy of 94% is achieved using this approach with the test corpus being publicly available data. The Phishing Email can be detected by observing sender behavior using the behavior based feature. As a disassembly tool, all the features are obtained using Mbox2Xml. In [35] presents the idea of Phishing terms weighting which assesses the weight of Phishing terms in each Email. The pre-processing stage is upgraded by applying content stemming and WordNet ontology to advance the model with word equivalent words. The model connected the knowledge discovery procedures utilizing five well known classification algorithms and accomplished an eminent improvement, 99.1% accuracy was achieved utilizing the Random forest algorithm and 98.4% utilizing J48, which is as far as anyone is concerned the most noteworthy precision rate for an accredited dataset. This paper additionally gives relative report comparable proposed classification strategies. In [36] proposed to identify Phishing Emails through hybrid features. The hybrid features comprise of content based, URL based, and behavior based features. In view of an arrangement of 500 Phishing Emails and 500 authentic Emails, the proposed technique accomplished overall accuracy of 97.25% and error rate of 2.75%. This promising outcome confirms the adequacy of the proposed hybrid features in distinguishing Phishing Email. A study [33] center around recognizing fake Email which is a type of Phishing attacks by proposing a novel structure to precisely distinguish Email Phishing attacks as well as advertisements or pornographic Emails consider as attracting ways to launch Phishing. The approach can identify and alert all sort of tricky messages in order to help clients in decision making. In a study [37], a portion of the early outcomes on the classification of Spam Email utilizing deep learning and machine learning methods. To represent Emails Word2vec is utilized instead of using the popular keyword or other rule based methods. To create a learning model, vector representations are given as an input into neural network. Experiments [38] considers the detection of a phishing Email as a classification problem and this paper describes using classical machine learning algorithms to group Emails as phishing or ham. Maximum accuracy of 99. 87% is achieved in classification of Emails using Support vector machine and Random forest classifier. Smadi et al. [39] put forward a model for identifying Phishing Emails that extract the feature set from the different Email parts in the preprocessing stage and it is classified using J48 classification algorithm. In the experiments, a total of 23 features have been used. For train, test and validation, ten-fold cross-validation is used. The main aim is to improve the overall metrics by concentrating on the preprocessing stage and find out the best algorithm that can be used. The benefits of using preprocessing stage are shown in the results. The highest achieved accuracy is for the Random forest algorithm which is 98.87%. The merits and capabilities of ten different algorithms are shown with help of experimentation. In study [40], for detecting Phishing Email and to calculate its accuracy the multilayer feed forward network is used.

Methods like tf-idf along with SVD and NMF representations followed by machine learning techniques for classifying Emails as either legitimate or Phishing is used in [11]. During training, Decision tree and Random forest showed the highest accuracy. While testing it was seen that these methods performed less where over fitting because the dataset was highly imbalanced. Use of word embedding and Neural Bag-of-grams with deep learning architectures such as convolutional neural network/recurrent neural network/LSTM and classical neural network, multilayer perceptron is described in [12] in which long short-term memory network has achieved better results than others. This paper [15] evaluates the performances of classical machine learning techniques such as Logistic regression, and Support vector machine to classify whether it is Phishing or legitimate. Convolutional neural network/recurrent neural network/multilayer perceptron architecture along with the Word2vec embedding used in this work [16] has outperformed former rule based and classical machine learning based models. In the proposed system, no external data was provided to train the model. Convolutional neural network had a slightly better performance over recurrent neural network model on subtask1 (Email with header information) and recurrent neural network perform well for subtask 2 (Email without header information), on the test data. For subtask 1, the convolutional neural network managed a score of 95.2%, almost comparable to recurrent neural network and for subtask 2; the recurrent neural network managed a score of 93.1%, making the recurrent neural network a better and more versatile overall performer. A model using Keras Word Embedding and convolutional neural network to classify legitimate and Phishing Emails are discussed in the paper [17] combining these two will give a dense vector representation for words which are then used to classify Emails [18]. Following, in this work deep learning architectures are leveraged for phishing email detection.

3.2 Related Works on Malicious and Phishing URL Detection

There has been a sudden change and use of online trades over the earlier decade. With the increase in the sophistication of cybercriminals, Malicious and Phishing attacks have also increased. The constant development of Internet has prompted the fast spread of Phishing, malware and Spamming. Malicious URL tricks the unsuspecting user to become the victims. The most common techniques used to detect Malicious and Phishing is through blacklisting, However it lacks the ability to detect newly generated malicious URL. The approaches to detect malicious URL can be classified into two major categories; (i) Blacklisting or Heuristics, and (ii) Machine learning approaches.

Blacklisting is the classical approach to detect malicious URL by maintaining a list of known blacklisted URLs such that when a new URL is visited, a database lookup is performed. If that URL is found in the list during the lookup operation, it will be declared as malicious by generating a warning message. Otherwise, the URL will be regarded as benign. A huge number of new URLs are being generated using algorithms which can bypass the blacklists. In such cases, blacklisting cannot keep up with the exhaustive list. Therefore, blacklisting method cannot be considered as ideal for the rapidly changing technology even though many existing anti-virus systems use this technology due to its simplicity and efficiency. An extension of blacklisting based methods can be used in which a “blacklist of signatures” is created. This approach is known as heuristic approach, in which, a common attack is mapped to a signature on the basis of its behavior. The web pages will be scanned by the intrusion detection systems to find such signatures and they will set a flag in case of any suspicious findings. Since this method can detect the threats associated with the new URLs also, it offers good generalization capability compared to blacklisting. However, the use of heuristic method is limited to common threats and moreover, it can be bypassed using obfuscation techniques.

Machine learning approaches analyses the information regarding the URL and its corresponding website to extract the relevant features of URL and utilize both malicious and benign URLs to train the classical machine learning based prediction model. The features using which the model is trained can be of two types - static and dynamic. In the static approach, the analysis of a web page can be performed on the basis of the available information without going for the URL execution. During this analysis, the lexical attributes of the URL string, its host information, HTML and JavaScript contents are filtered out. This method is much safer and secure than the dynamic approaches as no execution is required. The distribution of these features are different for malicious and genuine URLs and hence a classical machine learning based prediction model which can make trustworthy predictions about the unseen or new URLs can be built based on the extracted features. The presence of a more guarded environment for collection of relevant information and the capability to generalize all kind of threats have made this technique suitable for user exploration. In dynamic analysis, techniques like monitoring the abnormal behavior of the system is performed for checking any anomaly. Dynamic methods suffer from inborn risks, and poses difficulties in implementation and generalization. To classify URL as Phishing or non Phishing category, a study [41] proposed feature based approaches. The variety of features for URL is extracted by studying the structure of URL. Two different algorithms are being used for the classification of URL. To build an efficient classifier, Random forest is used. Also, a novel approach for detecting Phishing URL by mining public dataset is introduced. The advantages, drawbacks and limitation in research in the field of Phishing detection is discussed in paper [42]. The identification of best anti-phishing techniques will help the industries and academia. Another study [43] focuses on detecting and predicting whether a URL is good or bad using simple algorithms. Also comparison with two other algorithms namely Support vector machine and Logistic regression is shown. A study [44] expects to give a complete study and an auxiliary comprehension of Malicious URL detection techniques utilizing classical machine learning. Moreover, this paper discusses the open research challenges and helps the classical machine learning researchers, professionals and practitioners in cyber security industry and also the engineering in academia to understand the state of art to facilitate research and practical application. Based on URLs lexical and host based features a study [45] classifies URL automatically. For each URL, cluster labels are derived by clustering the entire dataset. Scalable machine learning problem is addressed and batch learning is preferred over online learning. Examination of raw data is carried out along with the assessment of accuracy of the various feature subsets. The significance of bigrams is surveyed and reinforced by utilizing the chi-squared and data pick up trait assessment techniques. Online URL reputation services are utilized as a part of request to arrange URLs, and the class is utilized as a supplemental source of data that would empower the framework to rank URLs. The classifier accomplishes 93–98% precision by recognizing a substantial number of Phishing while keeping up a low false positive rate. The URL characterization and the URL classification systems work in conjunction to give URLs a rank. In a study [46], the utilization of URLs is investigated as contribution for classical machine learning models connected for Phishing site prediction. Along these lines, a correlation between a feature-engineering approach took after by a Random forest classifier against a novel technique in light of recurrent neural network. It is resolved that the recurrent neural network approach gives an accuracy rate of 98.7% even without the need of manual features, beating by 5% the Random forest method. This implies it is a versatile and quick acting proactive detection framework that does not require full substance examination. In [47] propose URLNet, an end-to-end deep learning framework to learn a non-linear URL embedding for malicious URL detection directly from the URL. They also propose advanced word-embeddings to solve the problem of too many rare words observed in this task. An extensive experiment on a large scale dataset was conducted and shows a significant performance gain over existing methods. Also ablation studies were conducted to evaluate the performance of various components of URLNet.

3.3 Related Works on Image Spam Detection

Initially, optical character recognition (OCR) techniques are followed to convert spam images into texts and these texts are passed as input to text based spam filtering texts [48]. The performance highly relies on the OCR techniques. This method doesn't work in an adversarial environment. This is due to spammer perform various adversarial manipulation of image contents. Following, in recent days researcher develops solution which directly classify Email attachment as spam or non-spam. There are many approaches proposed based on machine learning. All these approaches rely on feature engineering. These methods can be grouped into different categories based on low level image features, based on high level image features, combination of low level and high level features, based on textual features. The detailed studies of these techniques are discussed in detail [48]. The performance of these methods relies on feature engineering. In recent days, the application of convolution neural network surpassed the human level performance in many of the computer vision tasks. To transfer these performances towards image spam detection, this work applies convolutional neural network and combination of convolutional neural network and recurrent structures with the publically available dataset.

3.4 Related Works on Email Categorization

Email categorization has been remained as a significant research domain in recent days due to the occurrence of a large number of legitimate email traffic. In [49] discussed the major challenges involved in email categorization in comparison to the classical document classification. They have used 2 different publically available large dataset. To convert the email into email vectors bag-of-words is used. The bag-of-words representation is passed into various classical machine learning classifiers such as Maximum entropy, Navie Bayes, Support vector machine, Wide-margin window. They also discussed the importance of time based split in dividing the data into train and test dataset. Yang and Park [50] discussed the importance of header and metadata information towards email categorization. They used tf-idf text representation with Navie Bayes classifier. This classifier are implemented using rainbow package. Mock [51] implemented an add-on using cosine coefficient with nearest neighbour classifier. Islam and Zhou [52] proposed multi stage classification approach for email categorization which substantially reduced the false positive rate. Eugene and Caswell [31] evaluated the performance of deep learning architectures for email prioritization. Following, in this work the application of deep learning architectures are leveraged for email categorization.

4 Text Representation and Deep Learning Architectures

4.1 Text Representation

This section discusses various text representations which can be used to convert text into numerical vector.

4.1.1 Non-sequential Text Representation—Bag-of-words (BoW)

Bag-of-word is a method for feature extraction in text data, used as representation in natural language processing. In this model, basically, text is represented as a bag-of collection of its words which doesn't keep information related to specific order or structure and grammar. The two things involved in the model are vocabulary of the known words and the frequency of its occurrence. To estimate the frequency of occurrence, term document matrix (tdm) and term frequency-inverse document frequency (tf-idf) is used. Term document matrix is a way of representing the text in which a matrix is constructed based on frequency of occurrence in the document. The horizontal rows in the matrix represent the documents and the vertical columns represent the terms that occur in the corpus. Term frequency-inverse document matrix is a numerical statistic to determine how relevant a term is in a given document. It is the product of term frequency-inverse document frequency. The term frequency increases with the increases proportionally with the number of times a word appears in a corpus. It can be represented as $tf(t, d)$ where t represents term and d represents the document. Since, term frequency is the raw count of number of times the term appears in the document, the simplest term frequency scheme can be given as

$$tf(t, d) = f_{t,d} \tag{1}$$

where $f_{t,d}$ denotes the row count.

The inverse document frequency shows how much information does a particular word provides may it be a common or a rare word. It is the logarithmically scaled inverse of the number of documents in the corpus to the number of documents that contain a particular term. It can be represented as

$$idf(t, D) = \log \frac{N}{d \in D : t \in d} \tag{2}$$

where N is the total number of documents in the corpus and $d \in D : t \in d$ denotes the documents in which the term t appears. Since, tf-idf is the product of the both term frequency and the inverse document frequency and it can be represented as

$$tf - idf(t, d, D) = tf(t, D).idf(t, D) \tag{3}$$

The term frequency-inverse document frequency will assign weights such that the common terms are filtered out.

The matrices of tdm and tf-idf methods may not be in square form $N \times N$ which is a rare coincidence, however more like of $M \times N$ term document matrix. Term document matrix is very unlikely to be symmetric. Therefore, we introduce singular value decomposition which is an extension of the symmetric diagonal decomposition (SVD). Using SVD, we can find solution to the matrix approximation problem also known as low-rank matrix approximation problem, and then develop its application by approximating term document matrices. Non-negative network factorization is most imperative method which decays the term document matrix into document topic matrix and a topic-term matrix. In this procedure, a document term matrix is built with the weights of different terms (commonly weighted word frequency information) from a set of documents. This matrix is factored into a term feature and a feature document matrix.

4.1.2 Sequential Text Representation—Keras Word Embedding

A word embedding is an approach in which the documents and words are represented using a dense vector representation. It is an improvement over the classical bag-of-word model encoding method in which each word are represented using large sparse vectors. In an embedding, words are spoken to by dense vectors where a vector speaks to the projection of the word into a continuous vector space. The position of a word inside the vector space is found out from content and it is referred as embedding.

Keras offers an Embedding layer that can be utilized for neural network on text information. The information must be integer encoded. Each word will be represented by remarkable number esteem. Keras Tokenizer API is used for tokenization during data preparation stage. The Embedding layer is initialized with random weights and will learn embedding for all of the words in the train dataset. The Embedding layer is defined as the first hidden layer of a network. It must specify 3 arguments:

1. input_dim: vocabulary size of the text data.
2. output_dim: vector space size in which words will be embedded.
3. input_length: length of input sequences.

4.2 Methods

4.2.1 Logistic Regression

This is one of the most commonly used classical machine learning algorithm for both classification and prediction. Generally, it is a statistical algorithm which analyze when there are one or more independent variables determining the output. It is a special type of linear regression, where in, the Logistic regression predicts the probability of outcome by fitting the data to a logistic function given as

$$\sigma(z) = \frac{1}{1+e^{-z}}. \tag{4}$$

4.2.2 Deep Learning Architectures

Deep learning is the subfield of machine learning that exploits multilayer artificial neural networks (ANNs) to enhance performance by achieving state-of-the-art accuracy in complex tasks including computer vision, speech synthesis and recognition, language translation and many others [53]. Deep learning can be differentiated from classical machine learning approaches by its remarkable ability to learn representations automatically by itself from various forms of data such as audio, video, text, or images without the need of getting introduced to hand-written rules or domain expert knowledge. The flexible architecture enables them to learn directly from the raw data and also to enhance better prediction accuracy when more data is provided. In order to enhance the performance and to achieve low latency inference for the deep neural network (DNN) which is computationally intensive, the GPU accelerated inference platforms required.

Convolutional neural network and recurrent neural network are most commonly used deep learning architectures. Convolutional neural network have been widely employed in the image processing domain to extract the complex features through layer by layer by applying the filters on rectangular area. The complex features represent the hierarchical feature representations in which the features at the higher level are formed by the integration of several lower level features. The hierarchical representation of features in convolutional neural network enables them to handle data provided in different abstraction levels effectively. Set of convolution and pooling operations along with a non-linear activation function forms the basic convolutional neural network constituents. In recent days the advantage of using the ReLU as non-linear activation function in deep architectures is widely discussed due to ReLU as non-linear activation function is easy to train in comparison to logistic *sigmoid* or tanh non-linear activation function. Recurrent neural network is mainly used for sequential data modeling in which the hidden sequential relationships in variable length input sequences are learnt by them. Recurrent neural network approaches have the credit of many successful accomplishments in the area of natural language processing and speech synthesis and recognition [53]. During initial period, the applicability of ReLU non-linear activation function in recurrent neural network was not successful due to the fact that recurrent neural network results in large outputs. As the research evolved, authors showed recurrent neural network raised vanishing and exploding gradient problem in learning long-range temporal dependencies of large scale sequence data modeling. To overcome this issue, research on recurrent neural network progressed on the 3 significant categories. One is towards on improving optimization methods in algorithms; Hessian-free optimization methods belong to this category [53]. The second one is towards introducing complex components in a recurrent hidden layer of network structure; long short-term memory proposed in [53], a variant of long short-term memory network reduced parameters set; gated recurrent unit [53], and clock-work recurrent neural network [53]. Third one is towards the appropriate weight initializations; Recently, [53] authors have showed recurrent neural network with ReLU involving an appropriate initialization of identity matrix to the corresponding recurrent weight matrix can perform better compared to long short-term memory. They named the newly formed architecture of recurrent neural network as identity-recurrent neural network. The basic idea behind Identity recurrent neural network is that, while in the case of deficiency in inputs, the recurrent neural network stays in same state indefinitely in which the recurrent neural network composed of ReLU and initialized with identity matrix.

Recurrent structures: Recurrent neural network is an add-on to the classical feed forward network which is commonly used in sequence data modeling. The past state information of recurrent neural network is stored using the cyclic connection and can also help in finding present state. Recurrent neural network has performed well in numerous field of artificial intelligence such as computer vision, natural language processing, and speech processing and so on. The hidden state vector is recurrently updated using transition function in concord with the current input vector and the previously hidden state. This type of transition function is trained using the backpropagation through time (BPTT). While in the process of backpropagating error across many time-steps, the weight matrix has to be multiplied with the gradient signal. This causes the vanishing issue when a gradient becomes too small and exploding gradient issue when a gradient becomes too large [53]. To alleviate, research on recurrent neural network were focused on 3 significant directions. The first one was focused on improving the optimization algorithms such as Hessian-free optimization methods [53]. The contributions in the second direction includes the addition of complex components in recurrent hidden layer of network structure to introduce models such as long short-term memory [53], gated recurrent unit [53] which is a more compact version of long short-term memory with reduced set of model parameters, and the works in the third direction are concerned about appropriate weight initializations with an identity matrix typically which is known by the name identity-recurrent neural network [53].

Long short-term memory was introduced to tackle the vanishing and exploding gradient problem by ensuring the constant error flow. Unlike simple recurrent neural network units, long short-term memory adopted memory blocks. A memory block can be considered as a complex processing unit which comprises of one or more than one memory cell and a set of multiplicative gating units; namely the input and output gate. Memory block which acts as the primary unit can house the information across various time steps. It holds a built in self-connection called constant error carousel (CEC) with value 1 which will be triggered when no value is received from the external signal. The adaptive multiplicative gating units are responsible for controlling the states of a memory block over different time-steps. The entrance and denial for the input flow of cell activation to a memory cell is controlled by an input gate. The output states from a memory cell to other nodes are controlled by corresponding output gate. An extra component called forget gate [53] is attached to a memory block instead of CEC since the internal values of a memory cell can increase without any constraints. The forget gate in long short-term memory aids the network to forget and remember its former state values. Hence, it is being employed as the standard component in current long short-term memory architectures. And also, additional peephole connection is made from the internal states to all the gates for learning the precise timing of the outputs [53].

Long short-term memory is generally considered as a mapping between input sequence and its corresponding output sequence based on the values of three multiplicative units namely input, output and forget gate which are updated iteratively on a memory cell in the recurrent hidden layer of long short-term memory network. LeCun et al. [53] proposed a new recurrent neural network, named as an identity-recurrent neural network with minor changes to recurrent neural network that has significantly performed well in capturing temporal dependencies with long range. The minor changes are related to initialization tricks such as to initialize the appropriate recurrent neural network weight matrix using an identity matrix or its scaled version and use a non-linear activation function. Moreover, the performance of identity recurrent neural network is closer to long short-term memory in 4 important tasks; two toy problems, language modeling and speech recognition. In one of the toy problem, identity recurrent neural network outperformed long short-term memory networks. LeCun et al. [53] introduced a variant of long short-term memory network i.e. gated recurrent unit. It make use of a more compact set of parameters in which input gate and forget gate are combined together to form new gating units called update gate whose primary focus is to focus balance the state between the previous activation and the candidate activation without peephole connections and output activations. Architecture of unit in recurrent neural network, long short-term memory is shown in Fig. 2 and gated recurrent unit is shown in Fig. 3. In this work, there are three types of recurrent structures are used. In general recurrent structures accept $x = (x_1, x_{2,}, \ldots, x_T)$ (where $x_t \in R^d$) as input and maps to hidden input sequence $h = (h_1, h_2, \ldots, h_T)$ and output sequences $o = (o_1, o_2, \ldots, o_T)$ from $t = 1$ to T by iterating the following equations.

Recurrent neural network:

$$h_t = \sigma(w_{xh}x_t + w_{hh}h_{t-1} + b_h) \tag{5}$$

$$o_t = sf(w_{ho}h_t + b_o) \tag{6}$$

Long short-term memory:

$$i_t = \sigma(w_{xi}x_t + w_{hi}h_{t-1} + w_{ci}c_{t-1} + b_i) \tag{7}$$

$$f_t = \sigma(w_{xf}x_t + w_{hf}h_{t-1} + w_{cf}c_{t-1} + b_f) \tag{8}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(w_{xc}x_t + w_{hc}h_{t-1} + b_c) \tag{9}$$

$$o_t = \sigma(w_{xo}x_t + w_{ho}h_{t-1} + w_{co}c_t + b_o) \tag{10}$$

$$h_t = o_t \odot \tanh(c_t) \tag{11}$$

Gated recurrent unit:

$$u_t = \sigma(w_{xu}x_t + w_{hu}h_{t-1} + b_u) \tag{12}$$

$$f_t = \sigma(w_{xf}x_t + w_{hf}h_{t-1} + b_f) \tag{13}$$

$$c_t = \tanh(w_{xc}x_t + w_{hc}(f \odot h_{t-1}) + b_c) \tag{14}$$

$$h_t = f \odot h_{t-1} + (1 - f) \odot c \tag{15}$$

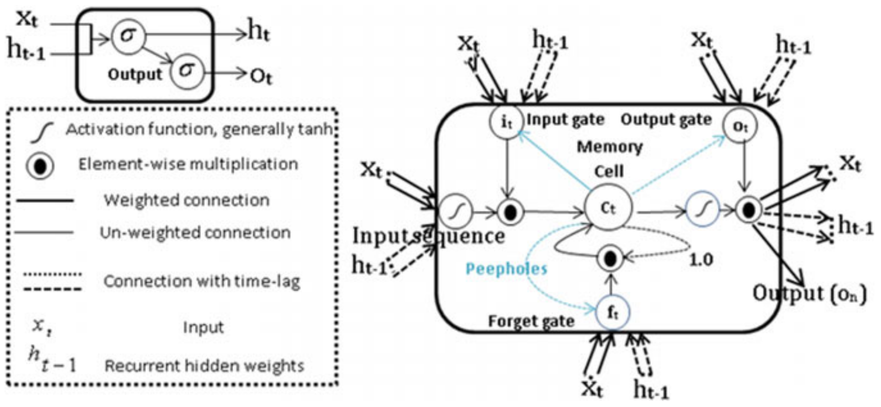
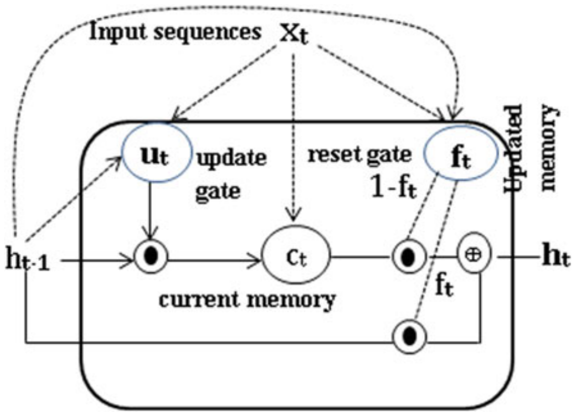


Fig. 2 Architecture of unit in recurrent neural network and long short-term memory

Fig. 3 Architecture of unit in gated recurrent unit



where w term denotes weight matrices, b term denotes bias, σ denotes *sigmoid* activation function, sf at output layer denotes non-linear activation function; in this work *sigmoid* is used, *tanh* denotes tanh non-linear activation function, i, h, f, o, c denotes input, hidden, forget, output and cell activation vectors, in gated recurrent unit input gate and forget gate are combined and named as update gate u.

Convolutional neural network: A convolutional neural network belongs to the class of deep feed forward ANNs. In order to minimize preprocessing, a variation of multilayer perceptron design is used in convolutional neural network. They are also called shift invariant or space invariant ANN (SIANN) considering their shared weight architecture and translation invariance characteristics. Convolutional neural network are almost same to classical neural networks. They can be seen as fabricated neurons with weight and bias values assigned to them. Their functioning involves receiving inputs, performing dot product of the inputs and applying non-linear mapping. The entire network behaves as one single differentiable score function. A convolutional neural network is basically a sequence of different layers. The three types of layers in convolutional neural network are convolutional, pooling, and fully connected layer. A convolutional layer composed of convolutional operation that depends on the dimension of the data. In this work convolutional 1D or temporal convolution is used. A common practice is to add a layer known as pooling layer between convolution layers in convolutional neural network. Such layers will reduce the representation's spatial size which in turn reduces the number of parameters and computations required in the network, and also help in controlling over fitting. Convolutional neural network includes pooling layers that may be local or global. Similar to the regular neural networks, neurons in a fully connected layer have complete connections to all activations in the past layer. Hence their activations can be calculated using matrix multiplication followed by a bias offset. Fully connected layers build connections from every neuron in a layer to the neurons in another layer. So it can be said that these networks holds the principle of classical multilayer perceptron also. The features learnt by convolutional layer are typically called as feature maps. These feature maps can be passed into recurrent structures such as recurrent neural network, long short-term memory, and gated recurrent unit to capture the sequence information in the feature maps. Architecture of convolutional neural network and convolutional neural network with recurrent structures is shown in Figs. 4 and 5 respectively.

In this work, a 1D data $x = (x_1, x_{2,}, \ldots, x_{n-1}, x_n, cl)$ passed as input (where $x_n \in R^d$ denotes features and $cl \in R$ denotes a class label). Convolution 1D operation generates a new feature map fm by using convolution with a filter $w \in R^{fd}$ where f denotes the features which results in a new set of features. A new feature map fm from a set of features $x_{i:i+f-1}$ is obtained as

$$h_i^{fm} = \tanh(w^{fm} x_{i:i+f-1} + b) \tag{16}$$

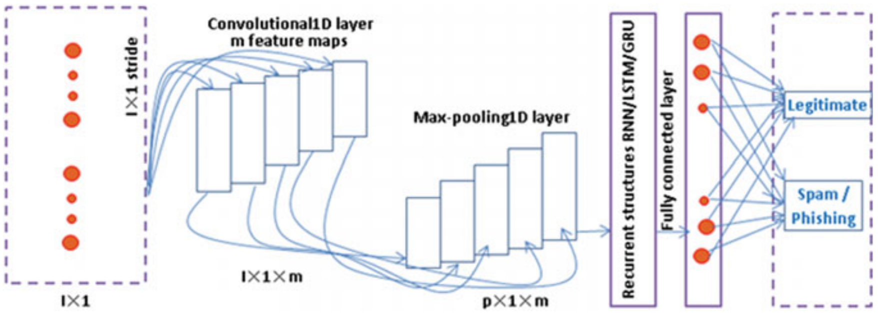


Fig. 4 Architecture of Convolutional neural network. All connections and hidden layers and its units are not shown. m denotes number of filters, ln denotes number of input features and p denotes reduced feature dimension, it depends on pooling length

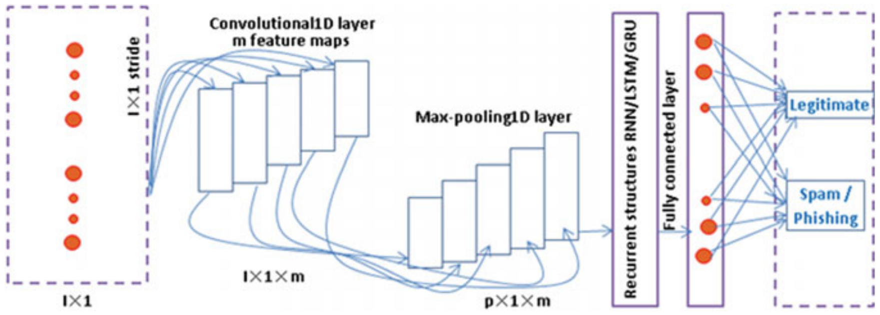


Fig. 5 Architecture of hybrid of Convolutional neural network and recurrent structures. All connections and hidden layers and its units are not shown. *m* denotes number of filters, *ln* denotes number of input features and *p* denotes reduced feature dimension, it depends on pooling length

where $b \in R$ denotes a bias term. The filter h is employed to each set of features f, $\{x_{1:f}, x_{2:f+1}, \ldots, x_{n-f+1}\}$ as to generate a feature map as

$$h = [h_1, h_2, \ldots, h_{n-f+1}] \tag{17}$$

where $h \in R^{n-f+1}$ and next we apply the max-pooling operation on each feature map as $\overrightarrow{h} = \max\{h\}$. This obtains the most significant features in which a feature with highest value is selected. However, multiple features obtain more than one features and those new features are fed to fully connected layer for classification. Otherwise, new feature map can also be passed into recurrent structure to capture the sequential information. A fully connected layer contains the *sigmoid* non-linear activation function that gives the values '0' or '1'. A fully connected layer is defined mathematically as

$$o_t = soft\max(w_{ho}h + b_o) \tag{18}$$

5 Description of Dataset

In this work, there are two types of datasets are used for Email and URL. One is publically available data and second one is privately collected samples. For private datasets, we have collected the Email and URL samples and manually assigned a label. Email samples which are collected from publically available sources are typically called as Spamdataset1. The detailed statistics is reported in Table 1. Email samples which are collected from private sources are typically called as Spamdataset2. The detailed statistics of Spamdataset2 is reported in Table 2. Email samples which are collected from public and private sources are typically called as SpamPhishingdataset1. The detailed statistics of SpamPhishingdataset1 is reported in Table 3.

URL samples which are collected from public sources are typically called as URL-dataset1. The detailed statistics of URLdataset1 is reported in Table 4. URL samples which are collected from private sources are typically called as URLdataset2. The detailed statistics of URLdataset2 is reported in Table 5. Spam and Phishing URL samples which are collected from both the public and private sources are typically called as SpamPhishURLdataset1. The detailed statistics of SpamPhishURLdataset1 is reported in Table 6. For image spam classification, in this work publically available benchmark dataset is used. The detailed statistics of spam and non-spam images are reported in Table 7. For Email categorization, this work used the privately collected data. The detail of Email categorization dataset is reported in Table 8.

Table 1 Detailed statistics of dataset collected from public source for spam email detection

Category	Legitimate	Spam
Train	102,768	98,915
Test	28,607	39,183

Table 2 Detailed statistics of dataset collected from private source for spam email detection

Category	Legitimate	Spam
Test	14,289	19,606

Table 3 Detailed statistics of dataset collected from public and private source for phishing email detection

Category	Legitimate	Spam and Phishing
Test	17,625	24,744

Table 4 Detailed statistics of Dataset collected from public source for spam URL detection

Category	Legitimate	Spam
Train	253,854	210,235
Test	40,000	100,000

Table 5 Detailed statistics of dataset collected from private source for spam URL detection

Category	Legitimate	Spam
Test	10,000	68,008

Table 6 Detailed statistics of dataset collected from private source for phishing URL detection

Category	Legitimate	Spam and Phishing
Test	10,000	18,008

Table 7 Detailed statistics of image spam detection dataset

Category	Non-spam	Spam
Train	725	839
Test	85	89

Table 8 Detailed statistics of email categorization dataset

Class	Number of samples
Academic	800
Personal	700
Trash	1,400

For Email spam detection, Email samples are collected from Lingspam,[3] PU,[4] CSDMC2010,[5] TREC[6] Spam Assian and Enron.[7] The malicious URL samples are collected from MalwareDomains,[8] MalwareDomainList,[9] JWSPAMSPY[10] and MalwareURL.[11] The phishing URL samples are collected from Phishtank[12] and OpenPhish.[13] The legitimate URL samples are collected from Alexa,[14] DMOZ[15] and Majestic.[16] The private datasets for Email and URL are collected from inside an Ethernet LAN.

6 Experiments on Spam and Phishing Detection

To handle large amount of data, Apache Spark[17] cluster computing platform is used. The framework uses distributed algorithms and all experiments related to deep learning architectures are run on GPU enabled machines. All classical machine learning algorithms are implemented using scikit-learn[18] and deep learning architectures are implemented using TensorFlow[19] with Keras.[20]

[3]http://www.aueb.gr/users/ion/data/lingspam_public.tar.gz.
[4]http://www.aueb.gr/users/ion/data/PU123ACorpora.tar.gz.
[5]www.csmining.org/.
[6]https://plg.uwaterloo.ca/~gvcormac/spam/.
[7]http://www.cs.bgu.ac.il/~elhadad/nlp16.html.
[8]http://www.malwaredomains.com/.
[9]https://www.malwaredomainlist.com/.
[10]http://www.joewein.de/sw/blacklist.htm.
[11]https://www.malwareurl.com/.
[12]https://www.phishtank.com/.
[13]https://openphish.com/.
[14]https://www.alexa.com/siteinfo.
[15]http://www.dmoz.org/.
[16]https://github.com/rlilojr/Detecting-Malicious-URL-Machine-Learning.
[17]https://spark.apache.org/.
[18]https://scikit-learn.org/.
[19]https://www.tensorflow.org/.
[20]https://keras.io/.

6.1 Experiments on Spam and Phishing Email Detection

6.1.1 Proposed Architecture—DeepSpamPhishEmailNet (DSPEN)

The proposed architecture for spam and phishing Email detection is shown in Fig. 6, named as DeepSpamPhishEmailNet (DSPEN).

- **Embedding**: In embedding, preprocessing is done on the dataset. Initially all characters are converted into small letters. Thus this can avoid regularization issue [8]. Otherwise, the deep learning architectures might need extra parameters to learn the significant characteristics which differentiate between the small and capital letters. All special characters are removed. We have taken only the words which are occurred frequently. In this way, the top 50,000 words are considered and formed Email vectors. All these operations are done using Keras tokenizer. To convert all Email vectors of same length, this work sets maximum length to 5,000. Email vectors which have maximum length than the 5,000 are discarded and less than the 5,000 are filled with zeros. These Email vectors are passed into embedding. In this work Keras embedding is used. It takes 3 parameters. These 3 parameters

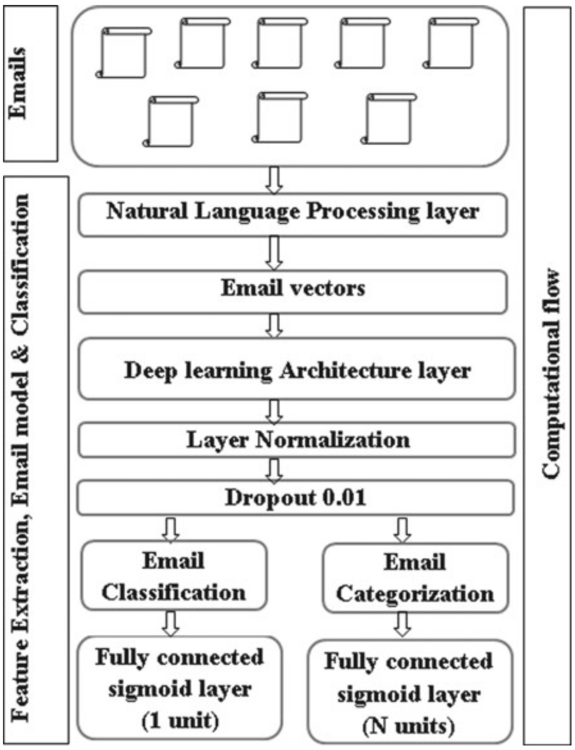

Emails
Feature Extraction, Email model & Classification
Natural Language Processing layer
Email vectors
Deep learning Architecture layer
Layer Normalization
Dropout 0.01
Email Classification
Email Categorization
Fully connected sigmoid layer (1 unit)
Fully connected sigmoid layer (N units)
Computational flow

Fig. 6 DeepSpamPhishEmailNet (DSPEN)

are hyper parameter which means these parameters can have direct impact on the performance. To choose these 3 parameter we run three trials of experiments on different values. One is maximum length of the vector which is set to 5,000, second one is embedding size which is set to 128 and maximum features which is the dictionary length, 50,000. This facilitates to convert the continuous URL vectors into dense vector representation by preserving the sequential information. This coordinatively works with the deep learning architectures to optimize the weights. These weights are initialized randomly at beginning and adjusted during backpropogation.

- **Optimal feature extraction**: It receives the Email representation from the embedding layer and extracts optimal features which can be used to distinguish between malicious or phishing or legitimate Email. There are different deep learning layers are used. These can be grouped into two categories. These are recurrent structures and convolutional neural network. Recurrent structures help to learn the sequential information whereas convolutional neural network helps to learn the spatial information. In order to learn both the spatial and sequential information hybrid of convolutional neural network and recurrent structures are used. In this work hybrid architecture performed well in compared to the other deep learning layers. Recurrent structures such as recurrent neural network, long short-term memory and gated recurrent unit have used 128 as the hidden unit's size. The hidden unit size is fixed based on several trials of experiments on different hidden units size. In convolutional neural network, filter size is set to 128 with filter length 3 and non-linear activation function *ReLU*. Convolutional neural network follows pooling, here maxpooling is used with pooling length 4. In convolutional neural network, we run three trials of experiments to set the values for filters and the filter length. To speed up the training and to avoid overfitting, batch normalization [53] and dropout is used in between the deep learning layers and classification. Additionally recurrent structures are used before the classification in order to learn the sequential information. In this case, the units' size is set into 70. Hidden unit sizes for recurrent structures are chosen by running three trials of experiments. The detailed configuration details of best performed architecture i.e. convolutional neural network-long short-term memory is shown in Table 9.
- **Classification**: This section composed of fully connected layer. In fully connected layer each neuron has connection to every neuron in the previous layer. In the case of convolutional neural network and convolutional neural network-recurrent neural network, convolutional neural network-long short-term memory, convolutional neural network-gated recurrent unit, the classification section composed of two fully connected layers. The first fully connected layer composed of 128 units which uses *ReLU* non-linear activation function and second fully connected layer contains 1 neuron with *sigmoid* non-linear activation function. To estimate the loss, binary cross entropy is used. Mathematically *sigmoid* and binary cross entropy is defined as follows

Table 9 Detailed configuration details of convolutional neural network-long short-term memory architecture

Layer (type)	Output shape	Param #
embedding_3 (Embedding)	(None, 50,000, 128)	6,400,000
conv1d_3 (Conv1D)	(None, 49,998, 128)	49,280
max_pooling1d_3 (MaxPooling1)	(None, 12,499, 128)	0
lstm_3 (LSTM)	(None, 70)	55,720
batch_normalization_2 (Batch)	(None, 70)	280
dropout_3 (Dropout)	(None, 70)	0
dense_3 (Dense)	(None, 256)	18,176
activation_3 (Activation)	(None, 256)	0
dropout_4 (Dropout)	(None, 256)	0
dense_4 (Dense)	(None, 1)	257
activation_4 (Activation)	(None, 1)	0
Total params: 6,523,713		
Trainable params: 6,523,573		
Non-trainable params: 140		

$$sigmoid = \sigma(z) = \frac{1}{1+e^{-z}} \tag{19}$$

$$loss(pr, ep) = -\frac{1}{N}\sum_{j=1}^{N}[ep_j \log pr_j + (1-ep_i)\log(1-pr_j)] \tag{20}$$

Here *ep* is a vector of expected class label, *pr* is a vector of predicted probability. To minimize the loss we used *adam* optimizer via backpropogation.

6.1.2 Results and Observations

To evaluate the deep learning architectures we have used different datasets. Deep learning architectures have used Keras embedding as email representation method. For comparative study tf-idf as email representation method with Logistic regression is used for classification. The train data, Spamdataset1 was used to train all deep learning architectures and classical machine learning algorithms. During training to monitor the train accuracy the train dataset is randomly divided into 70% train and 30% validation datasets. Finally the trained models are evaluated on the test data, Spamdataset1. With the aim to evaluate how well the models are able generalize on entirely new test samples. Spamdataset2 is used. In this case, only convolutional neural network-long short-term memory is tested. This is due to convolutional neural network-long short-term memory performed well in compared to other deep learn-

Table 10 Detailed test results of public data for spam email

Method	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
Public dataset								
CNN	0.956	0.935	0.992	0.963	25,916	2691	322	38,861
RNN	0.956	0.933	0.995	0.963	25,808	2799	188	38,995
LSTM	0.957	0.934	0.996	0.964	25,862	2745	173	39,010
GRU	0.956	0.933	0.995	0.963	25,796	2811	184	38,999
CNN-RNN	0.958	0.938	0.994	0.965	26,014	2593	229	38,954
CNN-LSTM	0.959	0.938	0.995	0.965	26,016	2591	204	38,979
CNN-GRU	0.959	0.938	0.995	0.965	26,019	2588	202	38,981
tf-idf LR	0.833	0.842	0.875	0.858	22,178	6429	4911	34,272
Private dataset								
CNN-LSTM	0.752	0.970	0.590	0.734	13,937	352	8045	11,561

Table 11 Detailed test results of public and private data for phishing email

Method	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
CNN-LSTM	0.653	0.629	0.990	0.769	3172	14,453	255	24,489

ing architectures. The performance is less on Spamdataset2 in compared to Spamdataset1. This is primarily due to the reason that the test Email samples of Spamdataset2 is entirely unseen samples and collected entirely in a different environment. Moreover, the trained model of convolutional neural network-long short-term memory on Spamdataset1 is evaluated on SpamPhishdataset1. The performance obtained by convolutional neural network-long short-term memory model is very less in compared to previous test datasets results. The detailed results are reported in Table 10 for Spamdataset1 and Spamdataset2 and Table 11 for SpamPhishdataset1.

6.1.3 Conclusion

In this sub module, the efficacy of classical machine learning with tf-idf text representation and deep learning architectures with embedding is used for spam and phishing detection of Email. During test experiments, the trained model is evaluated on entirely unseen samples of test Email. This helps to know how well the methods are generalizable on the entirely new test samples. In all the experiments, deep learning architectures with Keras embedding performed well in comparison to the tf-idf with classical machine learning algorithm. Keras embedding with hybrid convolutional neural network and long short-term memory performed well in comparison to the other deep learning architectures. This is due to Keras embedding has the capability to learn sequential information in the data. The performance of the reported results can be further enhanced by following hyper parameter selection approach and moreover other text representations such as word embedding models, skip-gram and continuous bag-of-words (CBOW), glove and neural bag-of-words can be used. Moreover, the robustness of the proposed method is not discussed in an adversarial environment. These works remained as significant directions towards future works.

6.2 Experiments on Spam and Phishing URL Detection

6.2.1 Proposed Architecture—DeepSpamPhishURLNet (DSPURLN)

The proposed architecture for spam and phishing URL detection is shown in Fig. 7, named as DeepSpamPhishURLNet (DSPURLN). It composed 3 important sections, they are

- **Embedding**: In embedding, initially all characters are converted into small letters. Thus this can avoid regularization issue [8]. Otherwise, the deep learning architectures might need extra parameters to learn the significant characteristics which

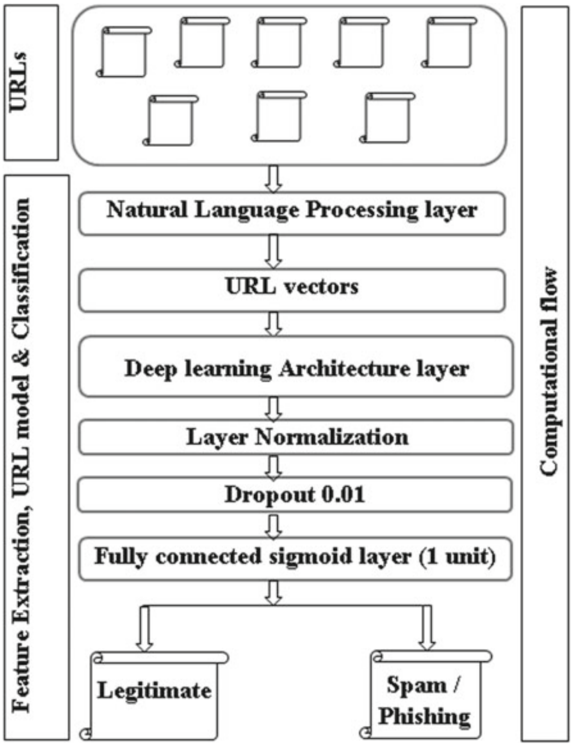

URLs
Feature Extraction, URL model & Classification
Natural Language Processing layer
URL vectors
Deep learning Architecture layer
Layer Normalization
Dropout 0.01
Fully connected sigmoid layer (1 unit)
Legitimate
Spam / Phishing
Computational flow

Fig. 7 DeepSpamPhishURLNet (DSPURLN)

differentiate between the small and capital letters. During training, a dictionary is created which contains a unique key for every character. Here, the dictionary length is 123 which mean the data contains 123 unique characters. Each character is mapped to an index of that character in a dictionary. The URL vectors are converted into same length by choosing the maximum length. Here maximum length remained as one of the hyper parameter. In this work, maximum length is fixed into 1,135. URL vectors less than the maximum length is filled with 0's and URL vectors greater than the maximum length are discarded. These URL vectors are passed into embedding. In this work Keras embedding is used. It takes 3 parameters. One is maximum length of the vector which is set to 1,135, second one is embedding size which is set to 128 and maximum features which is the dictionary length. We run experiments with embedding size 32, 64, 128 and 256. Experiments with 128 performed well, so the embedding size is set into 128. This facilitates to convert the continuous URL vectors into dense vector representation by preserving the sequential information. This coordinatively works with the deep learning architectures to optimize the weights. These weights are initialized randomly at beginning and adjusted during backpropogation.

- **Optimal feature extraction**: It receives the URL representation from the embedding layer and extracts optimal features which can be used to distinguish between malicious or phishing or legitimate URL. There are different deep learning layers are used. These can be grouped into two categories. These are recurrent structures and convolutional neural network. Recurrent structures help to learn the sequential information whereas convolutional neural network helps to learn the spatial information. In order to learn both the spatial and sequential information hybrid of convolutional neural network and recurrent structures are used. In this work hybrid architecture performed well in compared to the other deep learning layers. Recurrent structures such as recurrent neural network, long short-term memory and gated recurrent unit have used 128 as the hidden unit's size. The hidden unit size is identified based on running three trials of experiments on the various hidden unit sizes such as 32, 64, 128, 256. Experiments with 128 performed well in comparison to the 256. In convolutional neural network, filter size set to 128 with filter length 3 and non-linear activation function *ReLU*. To set the filter size and filter length, three trials of experiments are run on various filter size and filter length. Convolutional neural network follows pooling, here maxpooling is used with pooling length 4. To speed up the training and as well as to avoid overfitting, batch normalization [53] and dropout is used in between the deep learning layers and classification. Additionally recurrent structures are used before the classification in order to learn the sequential information. In this case, the units' size is set into 70. The detailed configuration details of best performed architecture i.e. convolutional neural network-long short-term memory is shown in Table 12.
- **Classification**: This section composed of fully connected layer. In fully connected layer each neuron has connection to every neuron in the previous layer. In the case of convolutional neural network and convolutional neural network-recurrent neural network, convolutional neural network-long short-term memory, convolutional neural network-gated recurrent unit, the classification section composed of two

Table 12 Detailed configuration details of convolutional neural network-long short-term memory architecture

Layer (type)	Output shape	Param #
embedding_1 (Embedding)	(None, 1,135, 128)	15,872
conv1d_1 (Conv1D)	(None, 1133, 128)	49,280
max_pooling1d_1 (MaxPooling1)	(None, 283, 128)	0
lstm_1 (LSTM)	(None, 70)	55,720
batch_normalization_1 (Batch)	(None, 70)	280
dropout_1 (Dropout)	(None, 70)	0
dense_1 (Dense)	(None, 256)	18,176
activation_1 (Activation)	(None, 256)	0
dropout_2 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 1)	257
activation_2 (Activation)	(None, 1)	0
Total params: 139,585		
Trainable params: 139,445		
Non-trainable params: 140		

fully connected layer. The first fully connected layer composed of 128 units which uses *ReLU* non-linear activation function and second fully connected layer contains 1 neuron with *sigmoid* non-linear activation function. To estimate the loss, binary cross entropy is used. Mathematically *sigmoid* and binary cross entropy is defined as follows

$$sigmoid = \sigma(x) = \frac{1}{1 + e^{-z}} \tag{21}$$

$$loss(pr, ep) = -\frac{1}{N}\sum_{j=1}^{N}[ep_j \log pr_j + (1 - ep_i)\log(1 - pr_j)] \tag{22}$$

Here *ep* is a vector of expected class label, *pr* is a vector of predicted probability for all testing samples. To minimize the loss we used *adam* optimizer via backpropogation.

6.2.2 Results

In order to identify the optimal deep learning architecture for URL analysis, various deep learning architectures are used and evaluated on different datasets. Deep learning architectures have used Keras embedding as URL representation method. For comparative study tri-gram as URL representation method with Logistic regression is used for classification. The size of tri-gram representation is very large.

To minimize feature hashing is used. This facilitates to reduce the computational time and also to achieve better performance. The train data, URLdataset1 was used to train all deep learning architectures and classical machine learning algorithms. During training to monitor the train accuracy the train dataset is randomly divided into 70% train and 30% validation datasets. Finally the trained models are evaluated on the test data, URLdataset1. With the aim to evaluate how well the models are able generalize on entirely new test samples URLdataset2 is used. In this case, only convolutional neural network-long short-term memory is tested. This is due to convolutional neural network-long short-term memory performed well in compared to other deep learning architectures. The performance is less on Spamdataset2 in compared to Spamdataset1. This is primarily due to the reason that the test URL samples of URLdataset2 is entirely unseen samples and are collected entirely in a different environment. Moreover, the trained model of convolutional neural network-long short-term memory on URLdataset1 is evaluated on URLPhishdataset1. The performance obtained by convolutional neural network-long short-term memory model is very less in compared to previous test datasets results. More importantly, the performances obtained by convolutional neural network-long short-term memory models on all three different test datasets are closer. This indicates that the convolutional neural network-long short-term memory has learned the complete URL representation. This can be deployed in real time to detect the malicious activities in near real time. In the case of Email Analysis, the performances obtained by convolutional neural network-long short-term memory model have large difference on three different test datasets. The detailed results are reported in Table 13 for URLdataset1 and URLdataset2 and Table 14 for SpamPhishURLdataset1.

Table 13 Detailed test results for spam URL

Method	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
Public dataset								
CNN	0.954	0.941	0.999	0.969	33,700	6300	100	99,900
RNN	0.962	0.950	0.999	0.974	34,760	5240	60	99,940
LSTM	0.985	0.979	1.00	0.989	37,840	2160	0	100,000
GRU	0.982	0.976	1.000	0.988	37,560	2440	20	99,980
CNN-RNN	0.991	0.988	0.999	0.993	38,740	1260	60	99,940
CNN-LSTM	0.995	0.994	1.000	0.997	39,360	640	0	100,000
CNN-GRU	0.992	0.989	1.000	0.994	38,840	1160	20	99,980
tri-gram LR	0.895	0.872	0.999	0.931	25,320	14,680	60	99,940
Private dataset								
CNN-LSTM	0.993	0.994	0.998	0.996	9618	382	142	67,866

Table 14 Detailed test results of public and private data for phishing URL

Method	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
CNN-LSTM	0.984	0.979	0.996	0.987	9618	382	78	17,930

6.2.3 Conclusion

In this sub module, the efficacy of classical machine learning with tri-gram text representation and deep learning architectures with Keras embedding is used for spam and phishing detection of URL. During test experiments, the trained model is evaluated on entirely unseen samples of test URL. This helps to know how well the methods are generalizable on the entirely new test samples. In all the experiments, deep learning architectures with Keras embedding performed well in comparison to the tri-gram with classical machine learning algorithm. Keras embedding with hybrid convolutional neural network and long short-term memory performed well in comparison to the other deep learning architectures. This is due to Keras embedding has the capability to learn sequential information in the data. The performance of the reported results can be further enhanced by following hyper parameter selection approach and moreover the advantageous of time split in dividing the data into train and test datasets is not discussed. Time split helps to meet zero day malware detection. Moreover, the robustness of the proposed method is not discussed in an adversarial environment. These works remained as significant directions towards future works.

6.3 Experiments on Email Categorization

6.3.1 Proposed Architecture, Results and Observations

The proposed architecture for Email categorization is similar to Fig. 6, named as DeepEmailCat (DEC). In deep learning layers, we have used only the convolutional neural network-long short-term memory. In this work we have changed only the parameter values in compared to the architecture reported in Table 12. The detailed configuration details of best performed architecture i.e. convolutional neural network-long short-term memory is shown in Table 15.

The datasets for email categorization is considerably less, reported in Table 8. Thus, only cross validation is done. Fivefold cross validation is applied for convolutional neural network-long short-term memory with Keras embedding and Logistic regression with tf-idf representation. The fivefold cross validation accuracy for email categorization is reported in Table 16. Deep learning architecture, convolutional neural network-long short-term memory performed well in comparison to the classical machine learning with tf-idf representation. This is due to the fact that tf-idf representation completely discards the sequence information.

Table 15 Detailed configuration details of convolutional neural network-long short-term memory architecture

Layer (type)	Output shape	Param #
embedding_4 (Embedding)	(None, 5,000, 128)	64,000
conv1d_4 (Conv1D)	(None, 4998, 64)	24,640
max_pooling1d_4 (MaxPooling1)	(None, 1249, 64)	0
lstm_4 (LSTM)	(None, 70)	37,800
batch_normalization_3 (Batch)	(None, 70)	280
dropout_5 (Dropout)	(None, 70)	0
dense_5 (Dense)	(None, 128)	9088
activation_5 (Activation)	(None, 128)	0
dropout_6 (Dropout)	(None, 128)	0
dense_6 (Dense)	(None, 1)	129
activation_6 (Activation)	(None, 1)	0
Total params: 135,937		
Trainable params: 135,797		
Non-trainable params: 140		

Table 16 Detailed results of fivefold cross validation for Email categorization

Method	Accuracy
CNN-LSTM	0.971
tf-idf LR	0.922

6.3.2 Conclusion

This sub module has discussed the efficacy of hybrid of convolutional neural network and long short-term memory with Keras embedding over classical machine learning algorithms with tri-gram text representation for email categorization. Convolutional neural network-long short-term memory performed well in comparison to the classical machine learning algorithm. This is primarily due to the fact that convolutional neural network-long short-term memory uses Keras embedding which helps to learn the sequential information. Due to the fewer amounts of data, only cross validation is done. The performance of the proposed approach has to be evaluated on entirely new data samples of Email in order to know the generalization capabilities of both classical machine learning and convolutional neural network-long short-term memory. This is remained as one of the significant direction towards future work.

6.4 Experiments on Image Spam Detection

6.4.1 Proposed Architecture, Results and Observations

The proposed architecture for image spam detection is shown in Fig. 8, named as DeepSpamImageNet (DSPIN). It composed of 3 different sections. In preprocessing, the given image is resized. In this work, the images are resized into 64 * 64. These are passed into convolutional neural network to extract optimal features. Finally these features are passed into classification. This composed of fully connected layer with non-linear activation function, *sigmoid* to classify the image into spam or non-spam. The detailed configuration details of best performed architecture is shown in Table 17.

- convolutional neural network with 1 CNN layer
- convolutional neural network with 2 CNN layers
- convolutional neural network with 3 CNN layers

To identify the optimal parameters for filter size, filter length, pooling and pooling length two trials of experiments are run till 200 epochs. Based on the results, the filter size is set into 32 * 32 with filter length 3 * 3, pooling to maxpooling, pooling length into 2 * 2. To avoid over fitting the maxpooling follows dropout 0.2. To identify the convolutional neural network structure, the above mentioned different convolutional neural network architectures are used. To identify the optimal parameters such as filter size, filter length, pooling and pooling length two trials of experiments are run till 200 epochs. Based on the results, the filter size is set into 64 * 64 with filter length 2 * 2, pooling to maxpooling, pooling length into 2 * 2. To avoid over fitting the maxpooling follows dropout 0.2. The performance obtained by convolutional neural network 3 layers is less in comparison to the convolutional neural network 2 layers and convolutional neural network 2 layers performed well in comparison to the convolutional neural network 1 layer. The train data which is reported Table 7 is randomly divided into 90% train and 10% validation dataset. Validation dataset facilitates to monitor the train accuracy. Both convolutional neural network 1 layer and convolutional neural network 2 layers architectures are run till 1,000 epochs

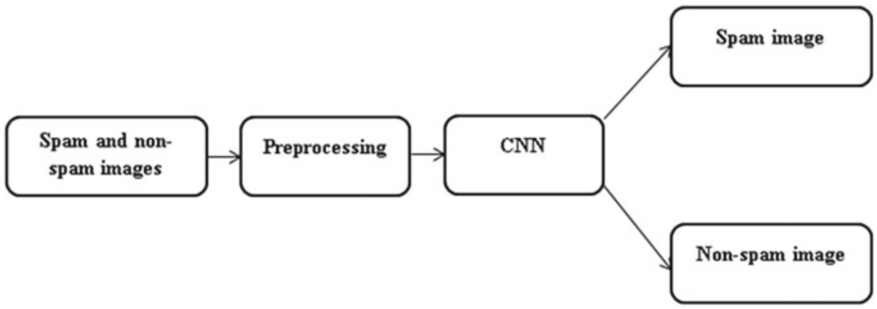


Fig. 8 Architecture of DeepSpamImageNet (DSPIN)

Table 17 Detailed configuration details of proposed architecture

Layer (type)	Output shape	Param #
conv2d_1 (Conv2D)	(None, 32, 64, 64)	896
activation_1 (Activation)	(None, 32, 64, 64)	0
max_pooling2d_1 (MaxPooling2)	(None, 32, 32, 32)	0
dropout_1 (Dropout)	(None, 32, 32, 32)	0
conv2d_2 (Conv2D)	(None, 64, 32, 32)	8256
activation_2 (Activation)	(None, 64, 32, 32)	0
max_pooling2d_2 (MaxPooling2)	(None, 64, 16, 16)	0
dropout_2 (Dropout)	(None, 64, 16, 16)	0
flatten_1 (Flatten)	(None, 16,384)	0
dense_1 (Dense)	(None, 256)	4,194,560
activation_3 (Activation)	(None, 256)	0
dropout_3 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 1)	257
activation_4 (Activation)	(None, 1)	0
Total params: 4,203,969		
Trainable params: 4,203,969		
Non-trainable params: 0		

Table 18 Detailed test results of image spam detection

Method	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
CNN 1 layer	0.966	0.966	0.966	0.966	82	3	3	86
CNN 2 layer	0.989	1.000	0.978	0.989	85	0	2	87

and each epoch checkpoints are saved based on the validation accuracy. Finally, the checkpoints are loaded and tested on the test dataset. The detailed test results of convolutional neural network 1 layer and convolutional neural network 2 layers are reported in Table 18. The performance obtained by convolutional neural network 2 layers is considerably good in comparison to the convolutional neural network 1 layer.

6.4.2 Conclusion

In this sub module, the application of convolutional neural network is used for image spam detection. The proposed method doesn't rely on any feature engineering. Feature engineering is considered as one of the daunting task and moreover these are vulnerable in an adversarial environment. There are two different convolutional neural network architectures are used. The performance of convolutional neural network 2 layers is good compared to convolutional neural network 1 layer. Thus the reported results can be further enhanced by following hyper parameter selection approach. Moreover, the performance of the proposed method has to be evaluated on unseen test samples to identify the generalizable capability of convolutional neural network architecture. Moreover, the significance of generative adversarial networks (GANs) can be used in order to make the robust convolutional neural network architecture. These are remained as significant directions towards future work.

7 DeepSpamPhishNet (DSPN)

The proposed architecture for Spam and Phishing activity detection is shown in Fig. 9, named as DeepSpamPhishNet (DSPN). It composed of 3 sections. They are (1) Data collection and preprocessing (2) Deep learning architectures (3) Classification.

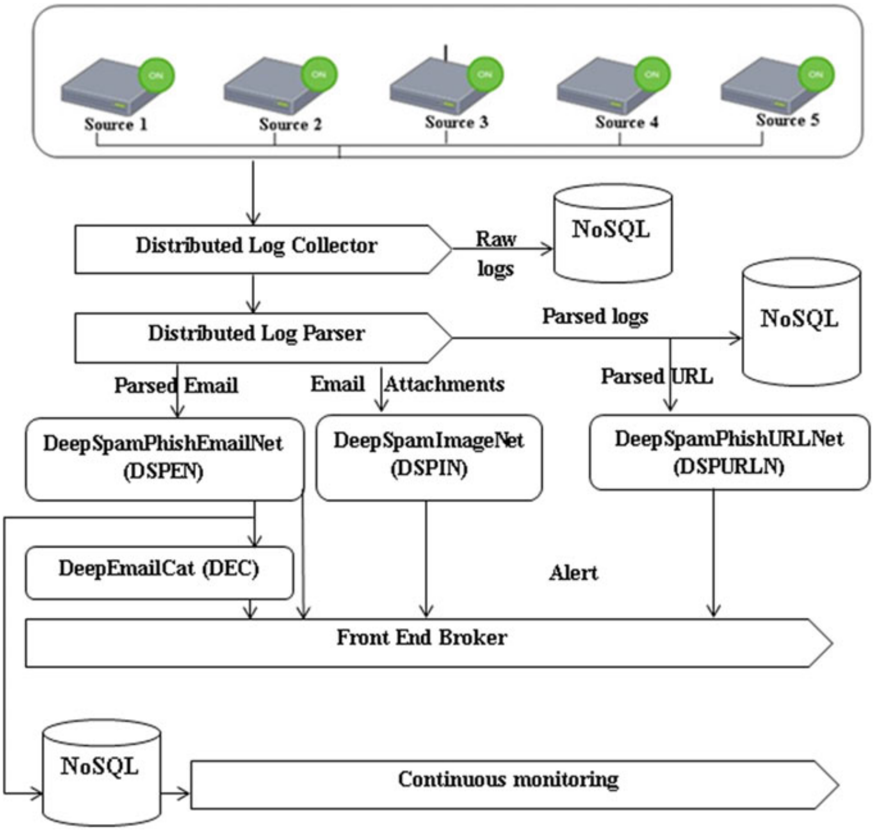


Fig. 9 DeepSpamPhishNet (DSPN)

In data collection, from various sources Email and URL data is collected in a distributed manner and stored in NoSQL data base. These are in turn passed into distributed data parser which facilitates to extract important information from the raw data. Finally, the preprocessed data is passed into NoSQL data bases for future use and passed into deep learning architectures. Deep learning architectures help to classify the samples into legitimate and spam/phishing activities. These detailed results are further passed into Front End Broker. Moreover, the domain name is continuously monitored in order to avoid malicious activities. The proposed architecture has the capability to collect data from various sources, preprocess it and detect the malicious activities in nature. This can be deployed in an organization level to monitor and to detect the malicious activities in a timely manner.

8 Conclusion, Future Work and Discussions

This work proposes Deep-Spam-Phish-Net (DSPN), a highly scalable deep learning based framework for Spam and Phishing detection. The framework contains two sub modules. A first sub module detects Spam and Phishing Email and second sub module detects Spam and Phishing URLs. The framework has the capability to collect myriad of security logs from various data sources, correlates and use classical machine learning algorithms and deep learning architectures to extract optimal features which can distinguish between benign and malicious activities. In each module the performances of deep learning architectures and classical machine learning algorithms are evaluated. Most of the cases, the deep learning architectures performed well in comparison to the classical machine learning algorithms.

The performance of the proposed framework to detect malicious activities can be enhanced by adding a sub module for DNS log analysis and malware analysis. This is due to, in recent day adversary uses domain generation algorithms (DGAs) to contact command and control (C2C) server. Moreover, in order to identify the detailed characteristics of malware, malware binary analysis is required. By adding these two sub modules to the existing framework, the performance in detecting malicious activities can be enhanced. This is remained as one of the significant direction towards future work.

Acknowledgements 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. Cormack GV (2008) Email spam filtering: a systematic review. Found Trends Inf Retr 1(4):335–455
2. Bhowmick A, Hazarika SM (2016) Machine learning for E-mail spam filtering: review, techniques and trends. arXiv preprint arXiv:1606.01042

3. Almomani A, Gupta BB, Atawneh S, Meulenberg A, Almomani E (2013) A survey of phishing email filtering techniques. IEEE Commun Surv & Tutor 15(4):2070–2090
4. Rao H, Shi X, Rodrigue AK, Feng J, Xia Y, Elhoseny M, Gu L (2019) Feature selection based on artificial bee colony and gradient boosting decision tree. Appl Soft Comput 74:634–642
5. Abdelaziz A, Elhoseny M, Salama AS, Riad AM (2018) A machine learning model for improving healthcare services on cloud computing environment. Measurement 119:117–128
6. Vinayakumar R, Poornachandran P, Soman KP (2018) Scalable framework for cyber threat situational awareness based on domain name systems data analysis. In: Big data in engineering applications. Springer, Singapore, pp 113–142
7. Mohan VS, Vinayakumar R, Soman KP, Poornachandran P (2018) Spoof net: syntactic patterns for identification of ominous online factors. In: 2018 IEEE security and privacy workshops (SPW). IEEE, New York, pp 258–263
8. Vinayakumar R, Soman KP, Poornachandran P (2018) Detecting malicious domain names using deep learning approaches at scale. J Intell & Fuzzy Syst 34(3):1355–1367
9. Vinayakumar R, Soman KP, Poornachandran P, Mohan VS, Kumar AD (2019) ScaleNet: scalable and hybrid framework for cyber threat situational awareness based on DNS, URL, and Email data analysis. J Cyber Secur Mobil 8(2):189–240
10. Vinayakumar R, Soman KP, Poornachandran P (2018) Evaluating deep learning approaches to characterize and classify malicious URLs. J Intell & Fuzzy Syst 34(3):1333–1343
11. Harikrishnan NB, Vinayakumar R, Soman KP, A machine learning approach towards phishing Email detection. In: CEN-Security@IWSPA 2018, pp 22–29. http://ceur-ws.org/Vol-2124/paper7
12. Vinayakumar R, Barathi Ganesh HB, Anand Kumar M, Soman KP, DeepAnti-PhishNet: applying deep neural networks for phishing email detection. In: CEN-AISecurity@IWSPA-2018, pp 40–50. http://ceur-ws.org/Vol-2124/paper9
13. Barathi Ganesh HB, Vinayakumar R, Soman KP, Anand Kumar M, Distributed representation using target classes: bag of tricks for security and privacy analytics. In: Amrita-NLP@IWSPA 2018, pp 11–16. http://ceur-ws.org/Vol-2124/paper10
14. Vazhayil A, Harikrishnan NB, Vinayakumar R, Soman KP, PED-ML: Phishing email detection using classical machine learning techniques. In: CENSec@Amrita, pp 70–77. http://ceur-ws.org/Vol-2124/paper11
15. Unnithan NA, Harikrishnan NB, Akarsh S, Vinayakumar R, Soman KP, Machine learning based phishing e-mail detection. In: Security-CEN@Amrita, pp 65–69. http://ceur-ws.org/Vol-2124/paper12
16. Moha VS, Naveen JR, Vinayakumar R, Soman KP, A.R.E.S : Automatic rogue email spotter crypt coyotes, pp 58–64. http://ceur-ws.org/Vol-2124/paper13
17. Hiransha M, Unnithan NA, Vinayakumar R, Soman KP, Deep learning based phishing E-mail detection CEN-Deepspam, pp 17–21. http://ceur-ws.org/Vol-2124/paper16
18. Unnithan NA, Harikrishnan NB, Vinayakumar R, Soman KP, Detecting phishing E-mail using machine learning techniques. In: CEN-SecureNLP, pp 51–57. http://ceur-ws.org/Vol-2124/paper17
19. Vinayakumar R, Soman KP, Poornachandran P (2017) Applying convolutional neural network for network intrusion detection. In: 2017 international conference on advances in computing, communications and informatics (ICACCI). IEEE, New York, pp 1222–1228
20. Vinayakumar R, Soman KP, 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). IEEE, New York, pp 1282–1289
21. Vinayakumar R, Soman KP, Poornachandran P (2017) Evaluation of recurrent neural network and its variants for intrusion detection system (IDS). Int J Inf Syst Model Des (IJISMD) 8(3):43–63
22. Vinayakumar R, Soman KP, Poornachandran P (2017) Applying deep learning approaches for network traffic prediction. In: 2017 international conference on advances in computing, communications and informatics (ICACCI). IEEE, New York, pp 2353–2358

23. Vinayakumar R, Soman KP, 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). IEEE, New York, pp 2026–2032
24. Vinayakumar R, Soman KP, 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). IEEE, New York, pp 266–274
25. Vinayakumar R, Soman KP (2018) DeepMalNet: evaluating shallow and deep networks for static PE malware detection. ICT Express
26. Vinayakumar R, Soman KP, Poornachandran P (2017) Deep android malware detection and classification. In: 2017 international conference on advances in computing, communications and informatics (ICACCI). IEEE, New York, pp 1677–1683
27. Elhoseny H, Elhoseny M, Riad AM, Hassanien AE (2018) A framework for big data analysis in smart cities. In: International conference on advanced machine learning technologies and applications. Springer, Cham, pp 405–414
28. Clark J, Koprinska I, Poon J (2003). A neural network based approach to automated e-mail classification. In: IEEE/WIC international conference on web intelligence, 2003. WI 2003. Proceedings. IEEE, New York, pp 702–705
29. Ruan G, Tan Y (2010) A three-layer back-propagation neural network for spam detection using artificial immune concentration. Soft Comput 14(2):139–150
30. Lennan C, Naber B, Reher J, Weber L, End-to-end spam classification with neural networks
31. Eugene L, Caswell I, Making a manageable email experience with deep learning
32. Bluszcz J, Fitisova D, Hamann A, Trifonov A (2016) Application of support vector machine algorithm in e-mail spam filtering (Patrick J'ahnichen, Preprint submitted to Patrick J'anichen, Advisor)
33. Mbah KF, Lashkari AH, Ghorbani AA (2017) A phishing email detection approach using machine learning techniques. World Acad Sci Eng Technol Int J Comput Inf Eng 4(1)
34. Hamid IRA, Abawajy J, Kim TH (2013) Using feature selection and classification scheme for automating phishing email detection. Stud Inform Control 22(1):61–70
35. Yasin A, Abuhasan A (2016) An intelligent classification model for phishing email detection. arXiv preprint arXiv:1608.02196
36. Rashwan MA, Al Sallab AA (2012) E-mail classification using deep networks. J Theor Appl Inf 37(2):241–251
37. Hassanpour R, Dogdu E, Choupani R, Goker O, Nazli N (2018) Phishing E-mail detection by using deep learning algorithms. In: Proceedings of the ACMSE 2018 Conference. ACM, New York, p 45
38. Rawal S, Rawal B, Shaheen A, Malik S, Phishing detection in E-mails using machine learning
39. Smadi S, Aslam N, Zhang L, Alasem R, Hossain MA (2015) Detection of phishing emails using data mining algorithms. In: 2015 9th international conference on software, knowledge, information management and applications (SKIMA). IEEE, New York, pp 1–8
40. Zhang N, Yuan Y (2012) Phishing detection using neural network. CS229 lecture notes
41. Sananse BE, Sarode TK (2015) Phishing URL detection: a machine learning and web mining-based approach. Int J Comput Appl 123(13)
42. Varshney G, Misra M, Atrey PK (2016) A survey and classification of web phishing detection schemes. Secur Commun Netw 9(18):6266–6284
43. Abdi FD, Wenjuan L Malicious URL detection using convolutional neural network
44. Sahoo D, Liu C, Hoi SC (2017) Malicious URL detection using machine learning: a survey. arXiv preprint arXiv:1701.07179
45. Feroz MN (2015) Examination of data, and detection of phishing URLs using URL ranking (Doctoral dissertation)
46. Bahnsen AC, Bohorquez EC, Villegas S, Vargas J, Gonzlez FA (2017) Classifying phishing URLs using recurrent neural networks. In: 2017 APWG symposium on electronic crime research (eCrime). IEEE, New York, pp 1–8

47. Le H, Pham Q, Sahoo D, Hoi SC (2018) URLNet: learning a URL representation with deep learning for malicious URL detection. arXiv preprint arXiv:1802.03162
48. Ketari LM, Chandra M, Khanum MA (2012) A study of image spam filtering techniques. In: 2012 fourth international conference on computational intelligence and communication networks (CICN). IEEE, New York, pp 245–250
49. Bekkerman R (2004) Automatic categorization of email into folders: benchmark experiments on Enron and SRI corpora
50. Yang J, Park SY (2002) Email categorization using fast machine learning algorithms. In: International conference on discovery science. Springer, Berlin, Heidelberg, pp 316–323
51. Mock K (2001) An experimental framework for email categorization and management. In: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, New York, pp 392–393
52. Islam MR, Zhou W (2007) Email categorization using multi-stage classification technique. In: Eighth international conference on parallel and distributed computing, applications and technologies, 2007. PDCAT'07. IEEE, New York, pp 51–58
53. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436