Knowledge and Information Systems (2022) 64:1457–1500
https://doi.org/10.1007/s10115-022-01672-x

SURVEY PAPER

Applications of deep learning for phishing detection: a systematic literature review

Cagatay Catal[1] · **Görkem Giray**[2] · **Bedir Tekinerdogan**[3] · **Sandeep Kumar**[4] · **Suyash Shukla**[4]

Received: 23 June 2021 / Revised: 27 February 2022 / Accepted: 5 March 2022 /
Published online: 23 May 2022

Abstract

Phishing attacks aim to steal confidential information using sophisticated methods, techniques, and tools such as phishing through content injection, social engineering, online social networks, and mobile applications. To avoid and mitigate the risks of these attacks, several phishing detection approaches were developed, among which deep learning algorithms provided promising results. However, the results and the corresponding lessons learned are fragmented over many different studies and there is a lack of a systematic overview of the use of deep learning algorithms in phishing detection. Hence, we performed a systematic literature review (SLR) to identify, assess, and synthesize the results on deep learning approaches for phishing detection as reported by the selected scientific publications. We address nine research questions and provide an overview of how deep learning algorithms have been used for phishing detection from several aspects. In total, 43 journal articles were selected from electronic databases to derive the answers for the defined research questions. Our SLR study shows that except for one study, all the provided models applied supervised deep learning algorithms. The widely used data sources were URL-related data, third party

✉ Cagatay Catal
ccatal@qu.edu.qa

Görkem Giray
gorkemgiray@gmail.com

Bedir Tekinerdogan
bedir.tekinerdogan@wur.nl

Sandeep Kumar
sandeep.garg@cs.iitr.ac.in

Suyash Shukla
suyashshukla2811@gmail.com

1 Department of Computer Science and Engineering, Qatar University, Doha, Qatar

2 Independent Researcher, Izmir, Turkey

3 Information Technology Group, Wageningen University & Research, Wageningen, The Netherlands

4 Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee, India

information on the website, website content-related data, and email. The most used deep learning algorithms were deep neural networks (DNN), convolutional neural networks, and recurrent neural networks/long short-term memory networks. DNN and hybrid deep learning algorithms provided the best performance among other deep learning-based algorithms. 72% of the studies did not apply any feature selection algorithm to build the prediction model. PhishTank was the most used dataset among other datasets. While Keras and Tensorflow were the most preferred deep learning frameworks, 46% of the articles did not mention any framework. This study also highlights several challenges for phishing detection to pave the way for further research.

Keywords Phishing detection · Malicious URL prediction · Deep learning · Machine learning · Systematic literature review (SLR) · Cybersecurity

1 Introduction

Phishing attacks aim to steal confidential information using sophisticated methods, techniques, and tools such as phishing through content injection, social engineering, online social networks, and mobile applications. While there are many different definitions of phishing, the following one is provided by The Anti-Phishing Working Group (APWG) [9]: "*Phishing is a crime employing both social engineering and technical subterfuge to steal consumers' identity data and financial account credentials*". This APWG is an international coalition that consists of 2200 institutions. Each year APWG publishes a phishing activity trend report. According to the report published in 2020, 34% of attacks targeted software-as-a-service (SaaS)/webmail users. In addition, after the COVID-19 pandemic in March 2020, the number of phishing attacks dramatically increased everywhere [9]. It is also interesting to see that SSL (Secure Sockets Layer) is used by 75% of phishing websites, and therefore, it can be inferred that SSL protocol does not always lead us to a legitimate website.

There are many other types of phishing attacks, however, most of the time this process is initiated with an email that scares users to take some immediate actions. In addition to the email communication media, phishing attacks can also target online social networks, blogs, forums, VoIP, mobile apps, and messaging platforms [11]. Recently, phishing scams addressing different systems including blockchain platforms have emerged. Since cryptocurrencies such as Bitcoin and Ethereum hit their highest prices in the market, cybercriminals targeted these digital assets [82]. These attacks may cause not only financial loss but also the loss of Intellectual Property (IP) and valuable confidential user information. It may also weaken the trust [65] and affect national security [55]. As such, phishing detection is more important than ever before.

Browsers can be considered as the first protection layer against phishing attacks. Blacklists provided by denunciation platforms such as PhishTank, SafeBrowsing, SmartScreen are used by the browser protection mechanisms [17]. It is also possible to use specialized security software such as Intrusion Detection System (IDS) or Intrusion Prevention System (IPS) for phishing detection [31]. The problem with the denunciation platforms is that the zero-day phishing attack, which is related to a newly designed phishing site, cannot be identified because it will not be on the blacklist for a while [56, P19]. In addition, the effort required for managing blacklists is too immense because phishing websites have a very short lifetime and new ones are designed quickly [5]. Also, a single character modification in the URL makes the website unknown for blacklists [17]. Since some phishing attacks such as spear phishing address only specific organizations and individuals, these websites might not end up in blacklists [71].

Due to these obstacles of the blacklists-based detection approach, heuristics-based approaches were developed [17]. Several features from the URL and the page content are used to build a prediction model for categorizing websites into malicious and legitimate classes. Different machine learning algorithms were commonly applied by many researchers for phishing detection so far [26, 85].

Several researchers presented different categorization approaches for phishing detection techniques. Basit et al. [11] categorized counter measurements into the following four categories: Machine Learning (ML), Deep Learning (DL), Scenario-based Techniques (ST), and Hybrid Techniques (HT). Deep learning is a sub-branch of Machine Learning that automatically discovers features and builds end-to-end prediction systems. For ST, different scenarios are taken into account and attacks are detected with the help of these scenarios. HT techniques are combinations of different approaches to achieve better results in accuracy and precision/recall evaluation metrics.

Recently, deep learning algorithms provided state-of-the-art results in different research problems such as face recognition and image classification. They were also successfully applied for several cybersecurity problems, namely malware detection, phishing detection, intrusion detection, spam email detection, and website defacement detection [43]. Although deep learning algorithms have been widely applied recently for phishing detection, there is a lack of a systematic overview of the use of these algorithms in phishing detection. Therefore, this research aims to present an overview of where and how DL algorithms have been used. To this end, we performed a Systematic Literature Review (SLR) to address nine research questions (RQs) defined in this research. We selected 43 journal articles from electronic databases to respond to these research questions. All the RQs are presented in Sect. 3. In Sect. 2.2., we also compare our research with the existing review/survey studies on phishing detection. To the best of our knowledge, this is the most comprehensive, up-to-date, and in-depth SLR on the use of DL for phishing detection.

The following sections are organized as follows: Sect. 2 discusses the background and related work. Section 3 describes the research methodology. Sections 4 and 5 present the results of the SLR and the discussion, respectively. Finally, Sect. 6 provides conclusions and future work.

2 Background and related work

In the following subsection, the background on deep learning is provided in Sect. 2.1. Subsequently, Sect. 2.2 presents the related review studies and compares these with our study.

2.1 Deep learning (DL) and DL algorithms

2.1.1 Deep neural network (DNN)

The DNN model is among the principal created Artificial Neural Networks. It varies from the shallow NNs in terms of the number of hidden layers, which are more in DNN. Even though specific model designs may have varieties based on various problem prerequisites, the DNN model mainly comprises three different types of layers (input layer, an output layer, and hidden layer). The DNN model can be efficiently used for classification and regression problems [27].

However, the parameter's size increases with the increase in the number of input features, which affects the computational performance. The backpropagation learning algorithm can be used to learn the DNN model. The weight adjustment can be made by back-propagating the error in the output layer neurons to the preceding layers. The hyperparameters of the model and their tuning is a significant issue for the DNN model. The appropriate selection of these hyperparameters affects the performance of the model. Hence, finding the best model parameters is a major issue [28].

2.1.2 Convolutional neural network (CNN)

The CNN model contains convolution layers that are dependent upon the convolution operation. The convolution process has been first acquainted [25] with identifying hidden patterns from the image sequentially, i.e., from identifying low-level features to high-level features. The lower layers in the network are particularly responsible for identifying basic features; subsequent layers are responsible for identifying complex features. The CNN model is most frequently used for the classification problem in image processing.

The CNN architecture comprises four different layers: convolution layer, max-pooling layer, dropout layer, and the Multilayer Perceptron (MLP) layer. The MLP layer is fully connected in CNN architecture. Equation 1 shows a basic convolution operation, where f represents a feature map, t represents time, k represents kernel, v represents a variable, and p represents the input.

$$f(t) = f(p * k)(t) = \sum_{v=-\infty}^{\infty} p(v) * k * (t - v) \tag{1}$$

Equation 2 portrays a neural network design in which W indicates weight, I indicates input, b indicates bias, and y indicates the neuron's output. After obtaining the neuron's output, the softmax function is used to get the final output. The softmax function and the final output (denoted by O) are displayed in Eqs. 3 and 4 [30].

$$y_i = \sum_j W_{ij} * I_j + b_i \tag{2}$$

$$O = \text{softmax}(y) \tag{3}$$

$$\text{softmax}(y_i) = \sum\nolimits_j \frac{\exp(y_i)}{y_j} \tag{4}$$

2.1.3 Recurrent neural network (RNN)

RNN belongs to the Neural Network (NN) category, in which the previous stage's output is used as input in the current stage [19]. The inputs and outputs are free of one another in conventional neural networks. However, in examples like predicting the upcoming word in the sentence, past words are needed, and subsequently, it is required to remember past words. The RNN illuminated this problem with the assistance of a Hidden Layer. The RNN model is different from the DNN model in terms of processing inputs; the RNN model processes input recurrently using internal memory, as shown in Fig. 1. The principle and most significant component of RNN is the Hidden state, which recollects some sequence data.

The design of the RNN model comprises various layers and units in each layer. The hidden layer unit contains data about input history in the "state vector." The Back Propagation

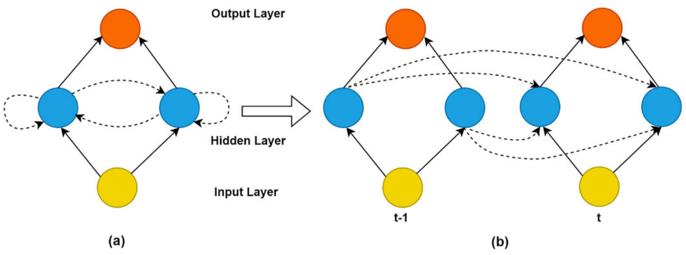

Fig. 1 The architecture of **a** RNN and **b** RNN over a time step [1]

Through Time (BPTT) algorithm can be used to learn the RNN model, whereas the most popular SGD and RMSProp algorithms can be used to optimize RNN parameters. The training of the RNN model is difficult than the other models due to its reliance over time. Hence, the RNN model's complexity increases with the increase in the learning period. However, the principle point of utilizing an RNN model is to learn long-term dependencies. It has been found in the literature that the learning of RNN is difficult for long-term dependencies [50].

2.1.4 Long short-term memory network (LSTM)

The LSTM network is an extension of an RNN model, which is acquainted with handling circumstances where RNNs fizzle [35]. In the RNN model, the previous stage's output is used as input in the current stage. The most famous applications of RNN are in the area of speech recognition [81]. However, they are not capable of storing past information for a long overrun. Thus, the LSTM network appeared, which is an advancement of the RNN model and is designed to remember long-term dependencies compared to the traditional RNNs.

The LSTM network comprises of LSTM layer, which is formed by merging LSTM units. An LSTM unit is made up of cells; each cell comprises various gates (input, output, and forget gate) that are responsible for regulating the information flow in the network. These features help each cell to remember the necessary information for a longer period [35]. The forward pass of an LSTM unit is displayed through Eqs. (4)–(8).

$$A_f = S_f(W_{f} * L_i + U_f * L_{o-1} + b_f) \tag{5}$$

$$A_i = S_f\left(W_{i} * L_i + U_i * L_{o-1} + b_i\right) \tag{6}$$

$$A_o = S_f(W_{o} * L_i + U_0 * L_{o-1} + b_o) \tag{7}$$

$$V_c = A_f * V_{c-1} + A_i * C_f(W_{c} * L_i + U_c * L_{o-1} + b_c) \tag{8}$$

$$L_o = A_o * H_f(V_c) \tag{9}$$

Here, L_i and L_o represent the input and output to the LSTM unit, respectively. A_f, A_i, and A_{o} are the activation vectors for the forget gate, input gate, output gate, respectively. Similarly, V_c represents a cell state vector, S_f represents a sigmoid function, and H_f and C_f are hyperbolic tangent function. W and U are the weight matrices, and b is the bias vector [32].

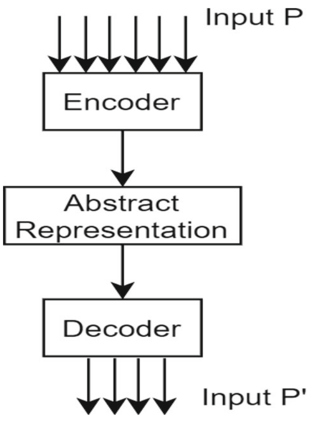

Fig. 2 The architecture of an autoencoder

2.1.5 Autoencoders

The autoencoder is a variant of the artificial neural network model that lies in unsupervised learning algorithms. The main purpose of an autoencoder is to reduce the dimensionality of the input data. The autoencoders can also be used for learning the generative model of data [18]. They convert the input data into some abstract representation and then from that representation to its original form utilizing the encoder function. The autoencoder attempts to approximate the identity function during this cycle. The main advantage of this model is to extract useful information and filter useless information from the data.

The autoencoder model is a feed-forward neural network model with a single hidden layer similar to a Multilayer Perceptron (MLP) model [13]. The target of the autoencoder model is to reconstruct the input. In contrast, the MLP model's target is to predict the output based on the input data. The input and output layer nodes are the same for both models. As Fig. 2 displays, the autoencoder model converts the input data P into some abstract representation Q utilizing weight matrix W. Then, Q will be converted into its original form utilizing the encoder function to get new input data P'.

2.1.6 Restricted Boltzmann machines (RBMs)

RBMs are frequently utilized as an unsupervised learning algorithm that learns the probability distribution over the input data [54]. RBMs are very much capable of detecting hidden patterns in an unsupervised fashion. However, its training process is difficult and can be considered as a disadvantage. The RBM is a bipartite and undirected graph comprised of visible and hidden layers, in which there is no association among the layers. Every cell in the network is responsible for processing the input data and deciding whether the nerve node will transmit it or not. Inputs after multiplication with weights and addition with bias will be provided to the activation function. The generated output will re-enter as input to the network in the reconstruction phase.

As shown in Fig. 3, neurons in both the layers (visible and hidden) form a bipartite graph. Each neuron in the visible layer is fully connected with the neurons in the hidden layer, whereas the same layer neurons are not connected among themselves [84]. The Gibbs sampler strategy will train an RBM model to minimize the log-likelihood of the data and model.

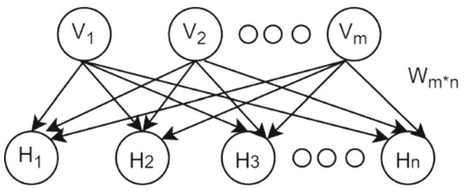

Fig. 3 The architecture of an RBM network

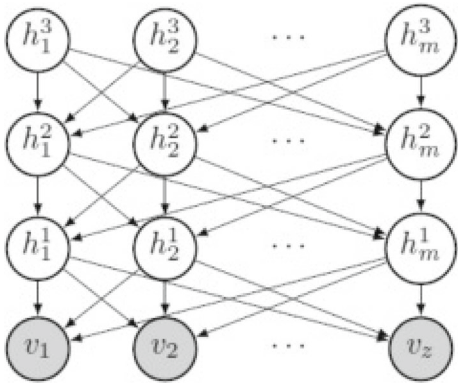

Fig. 4 The architecture of a DBN network [54]

2.1.7 Deep belief network (DBN)

The DBN model belongs to the category of an artificial neural network model. The DBN model comprises various RBM models in which the dependencies of hidden and visible layers are explored. It is a probabilistic generative model that comprises latent variables. The DBN model aims to identify different independent features from the input data utilizing unsupervised learning [46]. The disadvantage of the DBN model is similar to the RBM model. The DBN model first learns the input set in a probabilistic fashion, and then different independent features will be detected in the input. After the unsupervised learning of input, the classification will be done with the help of supervised learning.

As Fig. 4 shows, the DBN model training involves two steps; the first step involves stacked RBM learning and then backpropagation learning. The stacked RBM learning is performed utilizing the iterative Contrastive Divergence (CD) algorithm, whereas the backpropagation learning will be conducted using different optimization algorithms [37]. The hyperparameters are the same in both (RBM and DBN). The hyperparameters can also be optimized using the CD algorithm.

2.2 Related work

There are various studies on phishing detection and malicious URL detection utilizing machine learning (ML) and deep learning (DL) techniques. This motivated researchers to provide a summary of work done on this topic. So, in this sub-section, we discuss the existing reviews related to phishing detection.

In 2013, Khonji et al. [38] presented a survey of various phishing mitigation techniques. They have discussed phishing detection by blacklists, heuristics, visual similarity, and data mining solutions and found that the solutions based on ML techniques are most promising.

They found that detection at hour zero and low false-positive rate are critical measures for phishing solutions. Varshney et al. [70] have discussed the advantages and disadvantages of various web phishing detection techniques. They have also discussed the research gaps in web phishing detection that can be explored in the future. Dou et al. [20] have presented a systematic literature review on software-based phishing detection techniques. They have discussed phishing detection taxonomy, detection features, datasets, detection techniques, and performance evaluation measures. Goel and Jain [29] have also provided a taxonomy of mobile phishing attacks and various solutions to detect mobile phishing attacks. They have also discussed various challenges in mobile phishing detection solutions.

The applications of DL techniques for cyber-security problems (intrusion detection, android malware detection, malware classification, phishing detection, and spam detection) have been discussed by Mohammed Harun Babu et al. [48] and Berman et al. [14]. They found that the DL techniques are prominent in providing solutions for these problems. Similarly, Sahoo et al. [58] and Ferreira [24] have discussed ML techniques to detect malicious URLs. They found that the ML online algorithms are gaining attention for malicious URL detection due to training data size. They also suggested the use of appropriate feature extraction techniques for the success of the ML model. Wong [80] discusses different techniques for malicious web content detection and found that DL techniques with feature extraction provide an effective solution.

Zuraiq and Alkasassbeh [87] discussed various Content-Based, Heuristic-Based, and Fuzzy Rule-Based phishing detection approaches, along with their advantages and disadvantages. Kiruthiga and Akila [39] have discussed the applications of machine learning techniques for phishing website detection. They discussed some conventional ML techniques and newly developed systems like PhishScore and PhishChecker for phishing website detection. Singh and Meenu [62] have also reviewed ML techniques for phishing website detection. They have also discussed various protection approaches from these attacks. Benavides et al. [12] have presented a systematic literature review on DL-based phishing detection techniques. They have characterized, classified, and analyzed various DL based solutions for phishing attacks. They found that DL based algorithms have not been explored enough for the detection of cyber threats.

3 Research objectives and method

This section describes the research objectives and the method used in this study. An SLR approach was adopted to synthesize the knowledge on phishing detection via deep learning algorithms. The research method was based on well-known review protocol and guidelines [40] and our experience in SMS and SLR studies [15, 16, 67, 69].

3.1 Goal and research questions

The scope and goal of this study were formulated using the Goal-Question-Metric approach [10] as follows.

Analyze the state-of-the-art in phishing detection.

for the purpose of exploration and analysis.

with respect to the approaches, data sources, datasets, feature selection techniques, DL algorithms, evaluation parameters and validation approaches, and implementation platforms used in machine learning model life cycle; and reported challenges and proposed solutions.

from the point of view of machine learning researchers.
in the context of deep learning.

As Kitchenham et al. [41] pointed out, research questions (RQs) must embody secondary studies' goals. Accordingly, the purpose of this study can be broken down into the following nine RQs.

RQ1. Which machine learning categories (i.e., supervised/unsupervised/semi-supervised/reinforcement learning) have been applied in deep learning-based phishing detection studies?

RQ2. What data sources/features (e.g., URL, email text, etc.) have been used for the development of the phishing detection models?

RQ3. Which public datasets have been evaluated during the development of the models?

RQ4. What kind of feature selection techniques have been used?

RQ5. Which deep learning algorithms (e.g., CNN, LSTM) have been applied?

RQ6. What kind of evaluation metrics and evaluation approaches have been used?

RQ7. Which deep learning algorithm worked best for phishing detection studies?

RQ8. Which deep learning implementation platforms have been preferred for the implementation of the models?

RQ9. What are the challenges and research gaps in phishing detection?

Figure 5 shows how our RQs are mapped to the machine learning model life cycle proposed by Amershi et al. [7]. We omitted the feedback loops and the iterations throughout the life cycle for the sake of simplicity. In the model requirements stage, designers decide what type of models are most appropriate for the phishing detection problem. During data engineering stages, teams look for available data sources and datasets, clean data if required, and prepare labeled datasets for supervised learning if labels are not already present. Feature engineering refers to the activities for extracting and selecting informative features for machine learning models [7]. For some approaches using DL algorithms, feature engineering stage is less explicit and combined with model training stage [7]. In model evaluation, teams evaluate output models using evaluation metrics and approaches, and chose the best performing model. Teams may prefer to use an implementation platform to develop models. During this model life cycle, teams may face with some challenges. After model development, the chosen model deployed and monitored on a production environment. We excluded these two stages since our primary studies did not include any information on these stages.

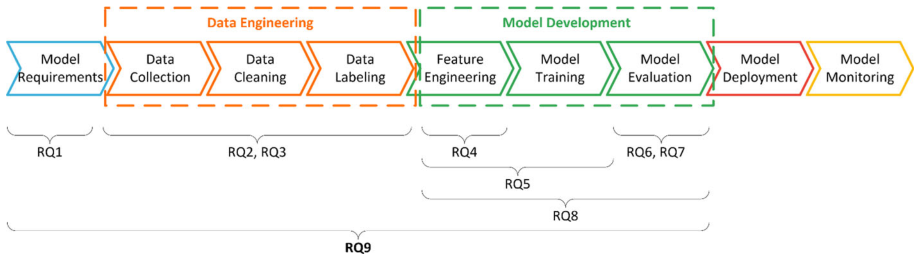

Fig. 5 Research questions mapped to the machine learning model life cycle adapted from Amershi et al. [7]

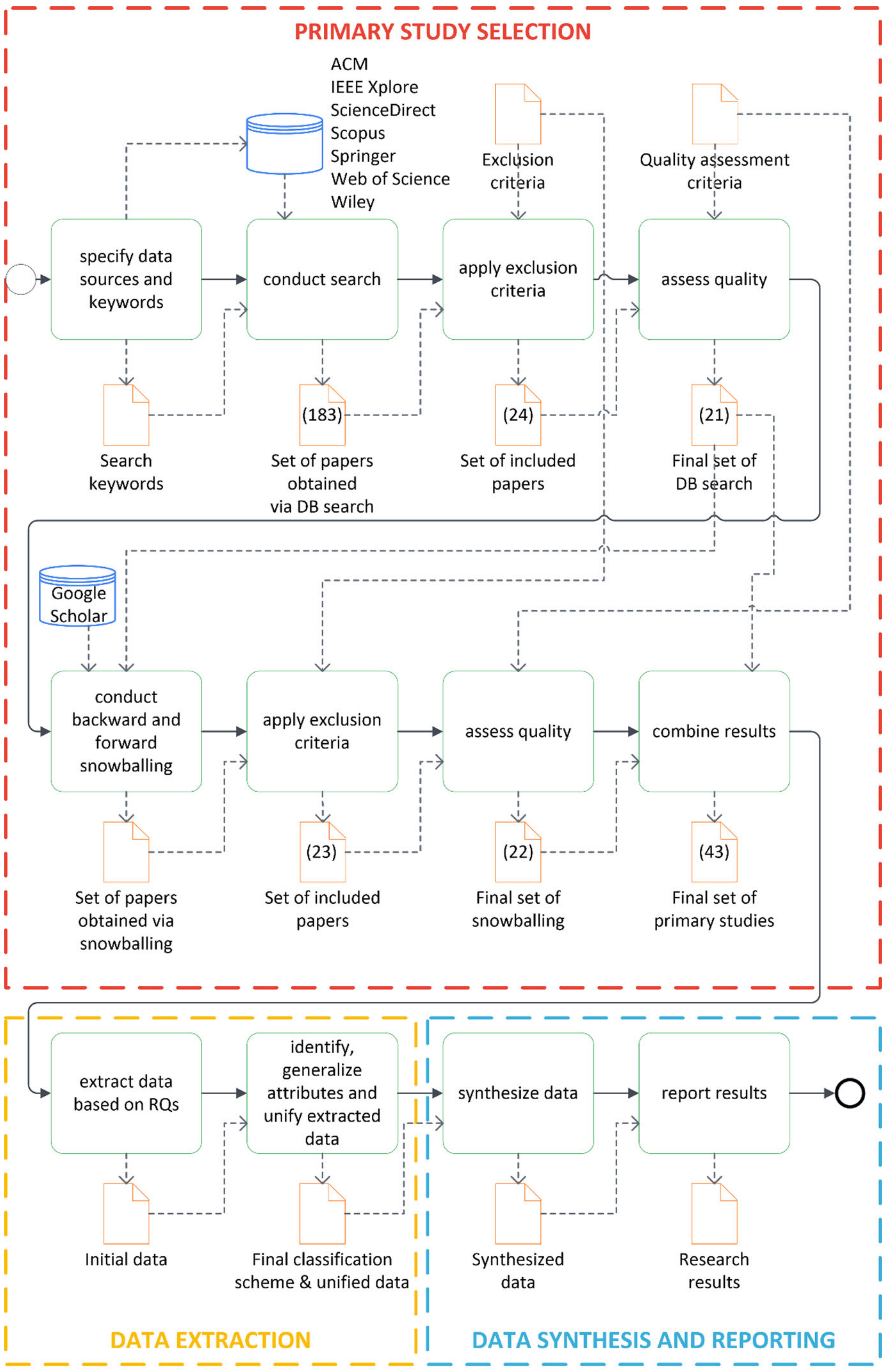

Figure 6. SLR process used in this study

Fig. 6 SLR process used in this study

As per our SLR process, three main phases, i.e., primary study selection, data extraction, and data synthesis and reporting, followed the definition of the goal and RQs. Figure 6 shows our SLR process and the Sects 3.2–3.4 present each of these phases in detail.

3.2 Primary study selection

The database search strategy was applied to identify the first set of relevant primary studies. Seven widely used online databases, i.e., ACM, IEEE Xplore, ScienceDirect, Scopus, Springer, Web of Science, and Wiley, were searched by using two key concepts: *"deep learning" AND "phishing"*. The keywords were kept general to have a high recall and relatively lower precision in the database search. Although this required more effort from the authors, obtaining a broader initial set of papers substantially decreased the possibility of missing some relevant studies.

The database search was conducted in August 2020. No criteria were set for the publication date. The database search yielded 183 papers. All the results were combined in a Google sheet. This sheet included some useful information about the papers, i.e., title, abstract, keywords, publication date, to be used in further steps.

Table 3 shows the exclusion criteria to filter out irrelevant papers and identify the relevant papers to answer our research questions. First, duplicate papers were removed from the pool (EC1). The papers whose full text was not available (EC2) and not written in English (EC3) were excluded. Only journal papers were included in the final pool to ensure a high-quality final set of papers (EC4). The papers published as conference proceedings, short papers, editorials, and issue introductions were excluded (EC5). The secondary studies were also excluded since the research objectives targeted primary studies (EC6). The studies that used at least one DL algorithm for phishing detection were included (EC7). The papers that address more general problems, such as spam email detection, authorship identification, domain name classification, bot detection in social networks, malware detection, were excluded unless they explicitly included an application of a DL algorithm to phishing detection. While the papers that used both DL and traditional ML algorithms were included, the papers that only used traditional ML algorithms were excluded (EC8). The papers without empirical results were eliminated (EC9) since these studies would not include the required information to answer the RQs.

The authors conducted a pilot selection on a randomly selected subset of papers. Each author applied the exclusion criteria on ten papers. Afterward, the authors held a meeting and explained how they applied the exclusion criteria. This think-aloud application of selection criteria [3] helped to clarify ambiguities and unintended interpretations. Applying the exclusion criteria resulted in a set of papers consisting of 24 papers.

A quality assessment was conducted before data were extracted from the primary studies, as proposed in the literature [34]. Table 4 lists the criteria we used for quality assessment. These criteria were derived from [40] and used in earlier SLRs, such as [69]. For each criterion, the papers were scored using a 3-point Likert scale (yes = 1, somewhat = 0.5, no = 0. For instance, Q1 was scored as 1 if the aim of a study was stated clearly in the introduction (expected place,as 0.5 if the aim was vaguely stated, or not at the expected place, and as 0 if the aim was not stated in the paper.

To maintain a high-quality input of primary studies for this SLR, the papers with a score lower than four points out of eight were excluded. Three studies [57, 66, 83] with a score under our threshold were excluded. The database search yielded a total number of 21 papers.

Table 1 Summary of deep learning algorithms

Deep learning model	Description	Characteristics
DNN	DNN model varies from the shallow NNs in terms of the number of hidden layers, which are more in DNN	*Pros:* It can be efficiently used for classification and regression problems It can solve more complex problems compared to the shallow network *Cons:* The parameter's size increases with the increase in the number of input features More hidden layers increase model complexity
CNN	Suitable for 2-dimensional input data The convergence speed increases by using ReLU	*Pros:* Neurons need not be fully connected like a conventional artificial neural network It can solve more complex problems compared to the shallow network *Cons:* A large labeled dataset may be needed A large number of layers may be required to form the network
RNN	The previous stage's output is used as input in the current stage The RNN model processes input recurrently using internal memory Well suited for speech recognition applications	*Pros:* Useful for modeling sequence data It can process inputs of varying length *Cons:* The training of the RNN model is difficult than the other models due to its reliance on time RNNs are not capable of storing past information for a long overrun

Table 1 (continued)

Deep learning model	Description	Characteristics
LSTM	Extension of an RNN model that works well in situations where RNN fails Each cell in the LSTM unit comprises input, output, and forget gate Works well for large range inputs	*Pros:* LSTM networks can store past information for a long overrun Capable of solving vanishing gradient problem *Cons:* Performance can be affected by different random weight initialization Vulnerable to Overfitting
Autoencoder	Frequently used for feature extraction or dimensionality reduction The target of the autoencoder model is to reconstruct the input using unsupervised learning	*Pros:* No need for labeled data More robust *Cons:* A pre-training stage is needed
RBM	It is a generative model that learns the probability distribution over the input data Detect hidden patterns in an unsupervised fashion Neurons in different layers form a bipartite graph	*Pros:* Missing values can be obtained using Gibbs sampling Can solve the problem of noisy data easily during reconstruction *Cons:* The training process is difficult
DBN	Comprises of the stack of RBM models Allows both (supervised and unsupervised) training of the network	*Pros:* Follows a sequential learning strategy for network initialization The inferences maximize the likelihood *Cons:* The training process may be expensive, just like RBM

Table 2 Summary of related work

Study	Review type	Focus	Description
Khonji et al. [38]	Non-systematic	Phishing detection techniques	Discusses the high-level overview of detection approaches, offensive defense approaches, correction approaches, and prevention approaches Detection at hour zero and low false-positive rate are critical measures Phishing solutions based on ML techniques are the most promising
Varshney et al. [70]	Non-systematic	Web phishing detection	Discusses various web phishing detection techniques and research gaps in web phishing detection They found that the search engine-based phishing detection techniques are the most suitable solutions
Dou et al. [20]	Systematic	Software-based web phishing detection	They have analyzed software-based phishing detection techniques They found that the true-positive rate (TPR), false-positive rate (FPR), and measures related to TPR and FPR are providing valuable performance PhishTank dataset is the most frequently used dataset, and phishing detection by blacklists toolbars are the most commonly used techniques
Goel and Jain [29]	Non-systematic	Mobile phishing attacks	They have discussed mobile phishing attacks and their solutions They have also provided a taxonomy of phishing defense mechanisms that will help users to understand the topic
Mohammed Harun Babu et al. [48]	Non-systematic	Deep Learning for cyber-security	Deep Learning for cyber-security

Table 2 (continued)

Study	Review type	Focus	Description
Sahoo et al. [58]	Non-systematic	Malicious URL detection using machine learning	Discusses the applications of ML techniques for malicious URL detection along with their requirements and challenges to develop the ML-based solutions They found that online ML algorithms are finding attention due to the huge size of training data Feature selection is also important for the performance of ML techniques
Wong [80]	Non-systematic	Malicious web content detection using Deep Learning	They reviewed various heuristics-based, ML-based, and DL-based methods for malicious web content detection They found that the DL techniques along with feature extraction provide an effective solution
Ferreira [24]	Non-systematic	Malicious URL detection using machine learning	Discusses the applications of ML techniques for the detection and prevention of malicious URLs They discussed lexical and content-based features for ML-based malicious URL detection They also discussed Naïve Bayes, Support Vector Machine, and Online algorithms for malicious URL detection

Table 2 (continued)

Study	Review type	Focus	Description
Berman et al. [14]	Non-systematic	Deep Learning for cyber-security	They discussed various DL techniques and reviewed studies that are using DL techniques to provide solutions for cyber-security attacks They found that the performance of different approaches varies based on the problem domain The TPR of the best performing DL technique for identifying malicious domain names is 96.01%-99.86%, whereas the TPR for network intrusion detection is 92.33%-100% They have also provided a taxonomy of phishing defense mechanisms that will help users to understand the topic
Zuraiq and Alkasassbeh [87]	Non-systematic	Phishing detection techniques	Reviewed various Content-Based, Heuristic-Based, and Fuzzy Rule-Based phishing detection approaches based on various datasets They found that no approach is better in every situation
Kiruthiga and Akila [39]	Non-systematic	Phishing website detection using machine learning	Discusses the applications of ML techniques for the phishing website detection Naive Bayes, Decision Tree, Support Vector Machine, and Random Forest are the most frequently used algorithms PhishScore and PhishChecker systems have also been proposed for the detection

Table 2 (continued)

Study	Review type	Focus	Description
Singh and Meenu [62]	Non-systematic	Phishing website detection using machine learning	Discusses the applications of ML techniques for phishing website detection along with various approaches protection approaches They found that the ML algorithms have achieved an approximate 99% accuracy for phishing website detection by combining 30 features
Benavides et al. [12]	Systematic	Phishing attack solutions using Deep Learning	Discusses the applications of DL techniques for phishing website detection Classified and analyzed various DL-based solutions DL algorithms have not been explored enough for phishing website detection Still, there is a need to identify one algorithm that can be useful in phishing website detection

Table 3 Exclusion criteria

#	Criterion
EC1	Duplicate papers from multiple sources
EC2	Papers without full text available
EC3	Papers not written in English
EC4	Papers not published in a journal
EC5	Short papers, editorials, issue introductions
EC6	Secondary studies, such as literature review, SMS, SLR
EC7	Papers which do not use deep learning for phishing detection
EC8	Papers which only use traditional ML algorithms
EC9	Papers which do not include empirical results

Table 4 Quality assessment criteria

#	Question
Q1	Are the aims of the study clearly stated?
Q2	Are the scope and context and experimental design of the study clearly defined?
Q3	Are the variables in the study likely to be valid and reliable?
Q4	Is the research process documented adequately?
Q5	Are all the study questions answered?
Q6	Are the negative findings presented?
Q7	Are the main findings stated clearly (regarding creditability, validity, and reliability)?
Q8	Do the conclusions relate to the aim of the purpose of the study, and are they reliable?

Forward and backward snowballing followed the database search to identify the primary studies that might have been overlooked [79]. The papers that cited the 21 primary studies were obtained via Google Scholar and the exclusion criteria were applied to these papers within the scope of the forward snowballing. In addition, the papers that were cited by the 21 primary studies were evaluated against our exclusion criteria within the scope of backward snowballing. Snowballing yielded an additional set of 23 primary studies.

The quality of these 23 papers was assessed using the criteria listed in Table 4. One paper [44] obtained via snowballing was disqualified after quality assessment. Figure 7 shows the quality scores of the selected primary studies via database search and snowballing. Finally, the metadata of the 43 primary studies (listed in Sect. 7.2) was combined in a Google sheet.

3.3 Data extraction

The data extraction phase followed the primary study selection. First, an initial data extraction form was formed based on the RQs. Data extraction steps were highly iterative and required close collaboration among the authors. The authors extracted data from five randomly selected papers separately and conducted a meeting to explain how they extracted data. The first column of Table 5 lists the fields of the final data extraction form.

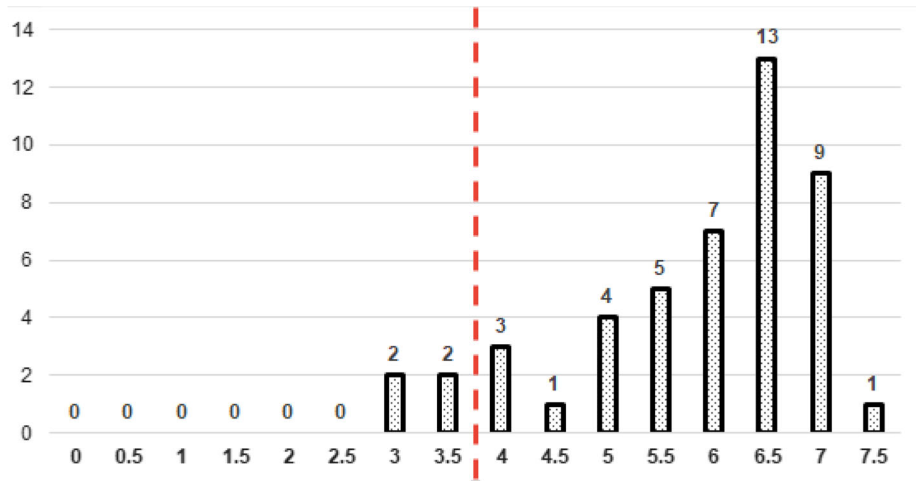

Fig. 7 Quality score distribution of the selected papers

While extracting data, the authors started to identify the categories for each of the fields. They obtained relevant data from the studies and tried to unify these data iteratively by conducting multiple meetings. As a result, the authors formed the final classification scheme, as listed in the second column of Table 5.

3.4 Data synthesis and reporting

Since it was possible to categorize the extracted data for most of the RQs, the data extraction phase yielded a set of quantitative data to be synthesized. The frequencies and percentages of each identified category were reported for the RQs between one and eight.

The only RQ that required qualitative analysis was RQ9, that is, the challenges and proposed solutions. 19 out of 43 primary studies included relevant data on challenges. The rest of the studies did not report any challenges. The challenges and the proposed solutions were recorded in Google sheet during data extraction. Open coding [45] was conducted to analyze the challenges. A code symbolically assigns a summative or evocative attribute for a portion of qualitative data [45]. Open coding was conducted in cycles. In the first cycle, any emerging patterns of similarity or contradiction were identified. In the second cycle, the codes were collapsed and expanded to understand any patterns. After the main themes and codes were extracted, the codes were revised and assigned to each challenge.

4 Results

This section mainly presents the responses to the RQs. In addition to these responses, we provide the following additional information related to the selected publications for this SLR study: Yearly distribution of papers, distribution of papers per journal, and a word cloud of the abstracts of the selected articles.

As shown in Fig. 8, 81% of articles (i.e., 35 papers) were published in 2019 and 2020. This figure shows that the popularity of deep learning algorithms in phishing detection is

Table 5 The data extraction form

Field	Categories
Journal title	Free text
Publication year	Number
Paper title	Free text
Abstract	Free text
ML category	Supervised, Unsupervised, Semi-supervised
Data sources	Third Party info on Web site, Company logo, Email, Social media post, URL, the Web site (Content, Code)
Evaluation dataset	5000 Best Websites, Alexa—Top Sites, Common Crawl, Contagio Mobile, Corpus of First Security and Privacy Analytics Anti-Phishing Shared Task, Curlie, Custom, DNS-BH—Malware Domain Blocklist by RiskAnalytics, joewein.de LLC, Malware Domain List, Nazario Phishing Corpus, OpenPhish, PhishTank, The Directory of the Web, The Enron—Spam Datasets, The Spamhaus Project, UCI dataset, Untroubled Software, VirusTotal
Feature selection method	Boruta, Correlation-based feature selection, Deep Belief Network, Genetic Algorithm, Greedy Selection algorithm, InfoGain, k-Best Chi^2, L1 based Linear Support Vector Machine (L-SVM-L1), Optimal sensitive feature selection algorithm, Principal Component Analysis, Recursive Feature Elimination (RFE), Sparse Random Projection, Variance Threshold (VT), Not mentioned
DL Approaches	Autoencoder, CNN, DBN, DNN, Hybrid DL Model, RNN/LSTM
Evaluation parameters	Accuracy, AUC, F-measure, FNR, FPR, Precision/PPV, Recall/Sensitivity/TPR, Specificity/TNR
Validation method	Cross-validation, Hold-out, Not mentioned
Best algorithm	Autoencoder, CNN, DBN, DNN, Hybrid DL Model, Non-ML/DL Approach, RNN/LSTM, Traditional ML, Not mentioned
Implementation platform	H2O, Keras, MATLAB, Microsoft Cognitive Toolkit (CNTK), Rstudio, Tensorflow, Theano, Not mentioned
Challenges and proposed solutions	Free text

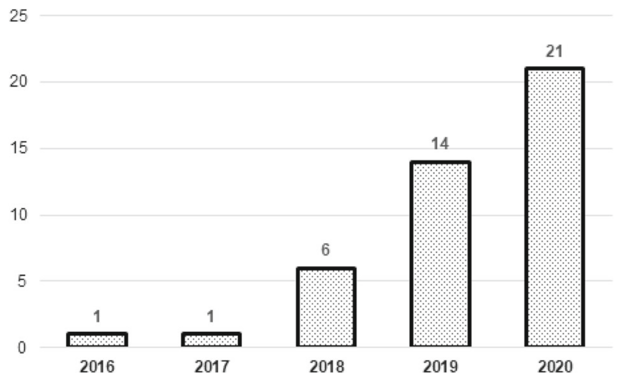

Fig. 8 Number of papers until August 2020

increasing in recent years. Although we completed our article search process in August 2020, the number of articles from 2020 is 1.5 times larger than the number of published articles in 2019. This observation shows that the new trend for phishing detection among cybersecurity researchers is the use of deep learning algorithms. In this study, we focused on only articles because they are high-quality papers and present in-depth experimental results. We expect to see more research on the development of deep learning-based models for phishing detection in the near future.

The distribution of articles per journal is presented in Table 6. The journal name, number of articles selected from this journal, and the references are provided in this table. According to this table, researchers preferred the following journals more than others did: IEEE Access, Journal of Intelligent & Fuzzy Systems, Neural Computing & Applications, and Security and Communication Networks. The other journals include only one selected article. This table shows that researchers who published phishing detection papers selected a diverse set of journals. In the future, this trend might change because some journals listed in this table particularly focus on security, and therefore, these journals might attract more researchers than the other journals.

In Fig. 9, we present the word cloud of abstracts of the selected 43 articles. This word cloud shows that LSTM and CNN algorithms are very common words in the articles. This observation is also aligned with our analysis, which states that LSTM and CNN algorithms are the most preferred approaches for model building. In addition, we see the term "classification" in this cloud. This shows that the preferred machine learning task is the classification for phishing detection problem.

4.1 RQ1. Machine learning categories

The first research question is related to the machine learning categories (i.e., supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning). We investigated the types of each deep learning-based phishing detection article and observed that 98% of papers (i.e., 42 articles out of 43 articles) applied supervised deep learning algorithms. There was only one article [23] that applied both supervised learning and unsupervised learning. However, the labeling process for supervised learning is time-consuming, labor-intensive, and expensive; researchers should also consider developing novel models that require less amount of labeled data points for building prediction models. Semi-supervised deep learning algorithms or unsupervised deep learning algorithms can be investigated to tackle this challenging problem. In semi-supervised algorithms, very few labeled data (e.g., 5–10%) are used to train the model and the class labels of the unlabeled data points are first predicted. This process continues for several iterations and better prediction results are achieved at the end of the process. Recently, different semi-supervised deep learning algorithms were developed and applied to different problems [60, 78, 86, P29]. In addition, unsupervised deep learning algorithms can help to reduce the burden of labeled data. Several unsupervised deep learning algorithms were also developed recently [59, 77, P26], Alom and Taha [6]); this type of algorithms can be used to develop novel phishing detection models.

Table 6 Distribution of papers per journal

Journal	# of papers	Reference(s)
IEEE Access	4	[P10, P30, P31, P32]
Journal of Intelligent & Fuzzy Systems	3	[72, P23, P26]
Neural Computing and Applications	3	[P15, P19, P34]
Security and Communication Networks	2	[74, 76]
Applied Intelligence	1	[75]
Applied Soft Computing	1	[P33]
Computer Networks	1	[P28]
Computers & Security	1	[P8]
Data Technologies and Applications	1	[P6]
Electronics	1	[4]
IEEE Internet of Things Journal	1	[P1]
IEEE Transactions on Big Data	1	[P13]
IET Information Security	1	[P5]
Information	1	[P7]
Information Security Journal	1	[P21]
Information Systems	1	[P14]
International Journal of Computational Intelligence and Applications	1	[51]
International Journal of Computer Science, Engineering and Information Technology	1	[P2]
International Journal of Network Security	1	[P25]
International Journal of Network Security & Its Applications	1	[P4]
International Journal of Research in Engineering, Science and Management	1	[P9]
International Journal on Artificial Intelligence Tools	1	[P24]
Iran Journal of Computer Science	1	[P12]
Journal of Ambient Intelligence and Humanized computing	1	[P11]
Journal of computing and Information Technology	1	[23]
Journal of Cyber Security Technology	1	[61]
Journal of Enterprise Information Management	1	[P3]
Journal of Experimental & Theoretical Artificial Intelligence	1	[P22]
Journal of Information Processing	1	[P18]
Journal of Network and Computer Applications	1	[P17]
Journal of Systems and Information Technology	1	[P16]
Neural Networks	1	[P29]
Pervasive and Mobile Computing	1	[P20]
Sadhana	1	[63]
Sensors	1	[P27]

Fig. 9 Word cloud of the abstracts

4.2 RQ2. Data sources/features

The second research question is related to data sources/features. For high-quality deep learning-based phishing detection models, the selection of data sources is crucial. We investigated what kind of data sources were used for building the prediction models and identified the following categories:

- URL data such as length, number of some relevant characters (like dots, "/"s), presence of HTTPS, domain, subdomain, and query path
- Third party info on the website such as Alexa ranking, WHOIS information, and DNS information
- Web site data such as HTML source code, JavaScript code, frame, and text content, and presence of IFrame
- Email data such as header, body text, attachment
- Company logo image
- Social media post

In Fig. 10, we represent the distribution of these data sources across all articles. According to this figure, the most preferred data source is the URL (i.e., 50%). In addition to the URL, third party information (i.e., 19%), web site data (i.e., 15%), and email related data (i.e., 12%) are also widely used data sources. Company logo and social media post are among the least used data sources. Different kinds of data can be retrieved based on the URL, and this is probably the reason why it is the most preferred data source for building phishing detection models. Our suggestion for researchers is to start with the URL data, and then, integrate other kinds of data such as third party information, web site, and email related data. If the performance does not improve when new data sources are added, they should not be included in the final detection model.

While the URL seems like a single data source, many different attributes can be created from this URL information. For example, the number of dot, hyphen, underline, slash, question mark, equal, @, &, exclamation, space, ~ , comma, plus, asterisk, hashtag, dollar, percent signs can be used as attributes based on the URL information. Also, similar attributes can be built using only the domain URL instead of the whole URL address. Similarly, additional attributes can be created based on URL directory, URL file name, and URL parameters. There can be more attributes using resolving URL and external services such as domain look up time response, number of redirects, domain expiration time, domain activation time, Time-To-Live (TTL), is_URL_indexed_on_Google, is_URL_shortened, is_domain_indexed_on_Google,

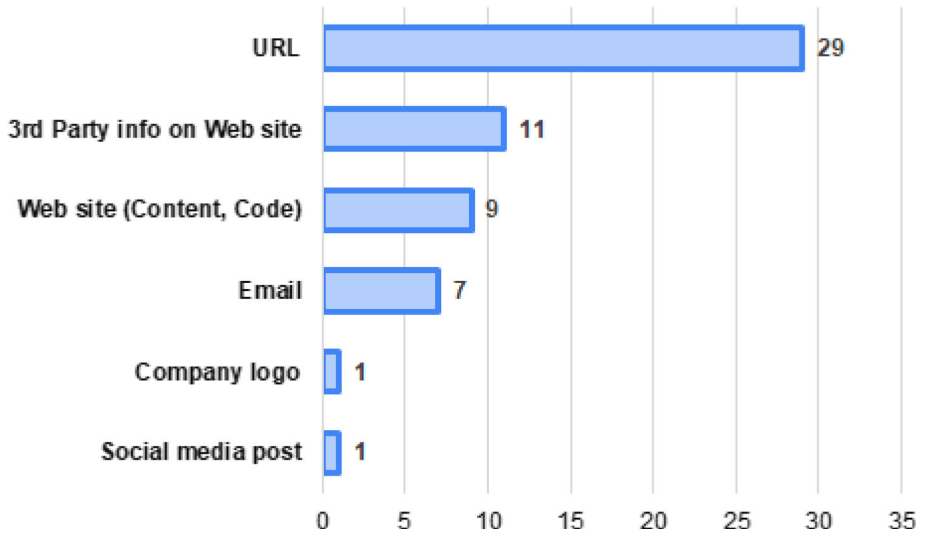

Fig. 10 Distribution of data sources

and has_valid_TLS/SSL_certificate [73]. As exemplified here, research studies applied different types of attributes using this data source. Some papers also used attributes regarding the third party information such as WHOIS information while building their models, however, most models applied the URL-related attributes. Web site content and code-related attributes such as JavaScript code or HTML code were also used in a few papers. Some researchers also considered some email data such as body text and attachment, however, the number of those papers is limited. Based on our analysis, we observed that most of the studies utilize the URL-related attributes in deep learning-based phishing detection.

4.3 RQ3. Datasets

We analyzed the datasets used in deep learning-based articles. These datasets and their corresponding web links are presented in Table 7. According to this table, 18 different datasets were investigated in these articles. In Fig. 11, we show the distribution of datasets. According to this figure, the most used dataset was PhishTank, which was preferred in 24 articles. There were also some custom datasets prepared by the researchers who wrote the corresponding article; 19 papers used this type of custom datasets. Alexa—Top Sites and UCI datasets are used in 13 and 11 studies, respectively. The other datasets were used less than or equal to five times. According to this figure, we can conclude that researchers preferred PhishTank, custom datasets, Alexa—Top Sites, and UCI datasets during their experiments.

There might exist more phishing detection datasets stored in different repositories, but we have only discussed the ones explained in the selected articles. For instance, platforms such as Kaggle might include this type of dataset. Researchers can take into account the dataset list that we presented in Table 7 while building phishing detection models. However, they can also support their research with additional datasets stored in other repositories. We noticed that many researchers used custom datasets; however, if these datasets were not made

Table 7 Malware datasets and their web pages

ID	Dataset	Web page
1	5000 best websites	http://5000best.com/websites/
2	Alexa—top sites	https://www.alexa.com/topsites
3	Common crawl	http://commoncrawl.org/
4	Contagio mobile	http://contagiominidump.blogspot.com/
5	Corpus of first security and privacy analytics anti-phishing shared task (IWSPA-AP 2018)	https://dasavisha.github.io/IWSPA-sharedtask/
6	Curlie	https://curlie.org/
7	DNS-BH—Malware domain blocklist by risk analytics	https://www.malwaredomains.com/
8	Joewein.de LLC	https://joewein.net/
9	Malware domain list	http://www.malwaredomainlist.com/
10	Nazario phishing corpus	https://monkey.org/~jose/phishing/
11	OpenPhish	https://openphish.com/
12	PhishTank	https://www.phishtank.com/
13	The directory of the web	http://dmoztools.net/
14	The Enron—Spam Datasets	http://nlp.cs.aueb.gr/software_and_datasets/Enron-Spam/
15	The spamhaus project	https://www.spamhaus.org/
16	UCI dataset	https://archive.ics.uci.edu/ml/index.php
17	Untroubled software	http://untroubled.org/
18	VirusTotal	https://www.virustotal.com/gui/

publicly available by authors, it is indeed very tough to repeat the experiments performed by the authors.

While we observed that there are some datasets that can be used in phishing detection model development, up-to-date appropriate benchmark datasets for fair comparison of models is lacking. Most of the datasets are not suitable for replication studies because the URLs used to build the dataset (i.e., short-lived websites) cannot be accessed easily and many studies used self-collected datasets using different sources. To address this problem, recently Hannousse and Yahiouche [33] designed a construction scheme of reproducible datasets, which are also extensible. Their strategy creates a balanced datasets because many available datasets are imbalanced and it was reported that imbalanced datasets may reduce the performance between 5.9 and 42% in term of F1 score [21]. Hannousse and Yahiouche [33] created a sample dataset by using their set of guidelines to demonstrate the applicability of the approach and showed that Random Forest algorithm works best on this dataset, however, they did not apply deep learning algorithms and planned to analyze them in future work.

4.4 RQ4. Feature selection techniques

Feature selection algorithms help us to select the most important features for modeling. In traditional machine learning algorithms, feature selection algorithms are widely used. In RQ4, we aimed to investigate whether feature selection algorithms were preferred when

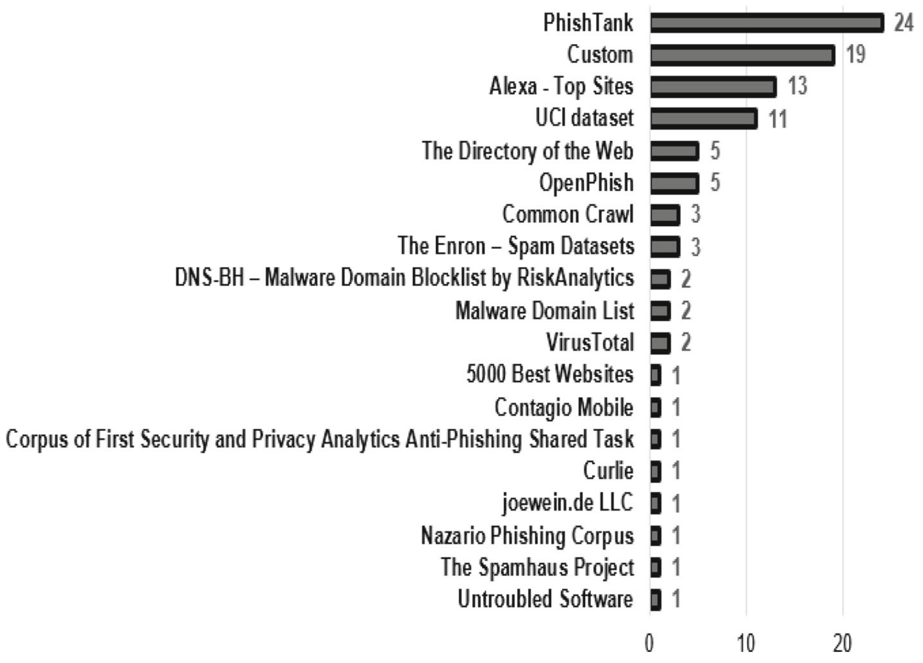

Fig. 11 Distribution of datasets

deep learning algorithms were applied. In Fig. 12, we see that 72% of articles (i.e., 31 papers out of 43 papers) did not mention any feature selection algorithm. We have also noticed that some studies applied more than one feature selection algorithm. Based on this figure, we can infer that most of the deep learning-based models do not require any feature selection algorithm and therefore, we see many papers that have the category "not mentioned"

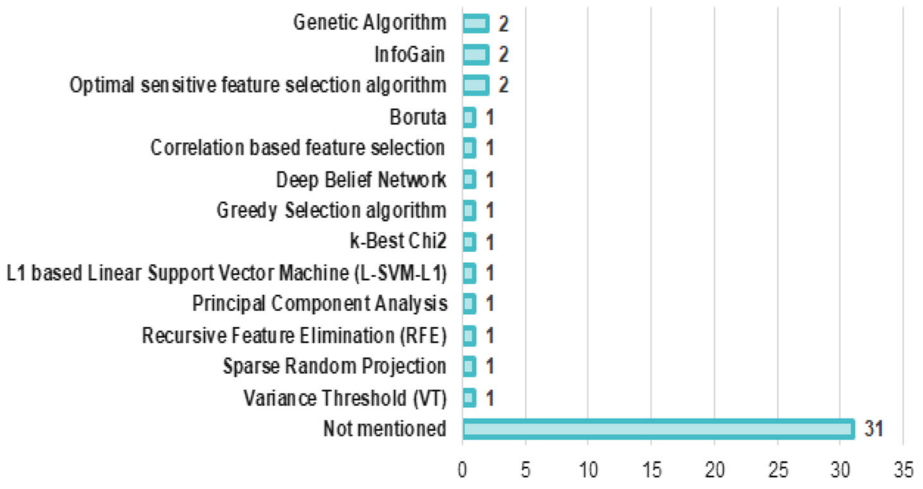

Fig. 12 Feature selection

for the feature selection dimension. The following algorithms are applied in two studies: Genetic algorithms, InfoGain, and optimal sensitive feature selection algorithm. The other feature selection algorithms shown in the figure was used only in one article. Since deep learning algorithms can handle a massive amount of data and high-dimensional data, the need for feature selection algorithms might be limited. If traditional machine learning-based phishing detection articles were investigated, we would see more feature selection algorithms in this analysis because machine learning algorithms were mainly used together with feature selection algorithms. There was also one deep learning algorithm, Deep Belief Network, which was used for feature selection. As we see in this case, DL algorithms can be used not only for classification tasks but also for feature selection purposes.

4.5 RQ5. Deep learning approaches

In Sect. 2.1, we explained deep learning and deep learning algorithms. The most preferred deep learning algorithm is the Deep Neural Network (DNN) algorithm (Fig. 13). The second and third most preferred algorithms are Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN)/Long Short-Term Memory Networks (LSTM). The other preferred algorithms are Hybrid Deep Learning, Deep Belief Network (DBN), and Autoencoder. Since some studies more than one algorithm, the total number of algorithms is larger than 43.

Most of the researchers are familiar with the traditional Artificial Neural Network (ANN) algorithms and DNN algorithms are very similar to the ANN algorithms (i.e., Multi-Layer Perceptron (MLP)). This might be the reason why DNN algorithms are the most used category. CNN and LSTM algorithms are the most frequently used deep learning algorithms. These algorithms also provided state-of-the-art results for many different research tasks. This might be the reason why these algorithms are also in the top three list.

The number of hybrid models is limited; however, we expect to see more papers using hybrid models because they can improve the performance of individual deep learning algorithms. We did not encounter any article that applied multi-task, multi-modal, multi-view, and Generative Adversarial Networks (GAN) deep learning algorithms. Researchers might consider using these algorithms to improve the performance of their phishing detection models. For instance, multi-task learning algorithms aim to address two tasks simultaneously and improve the performance of each task at the end of this process. There are different multi-task deep learning algorithms used for different problems in the literature [8, 64, 68].

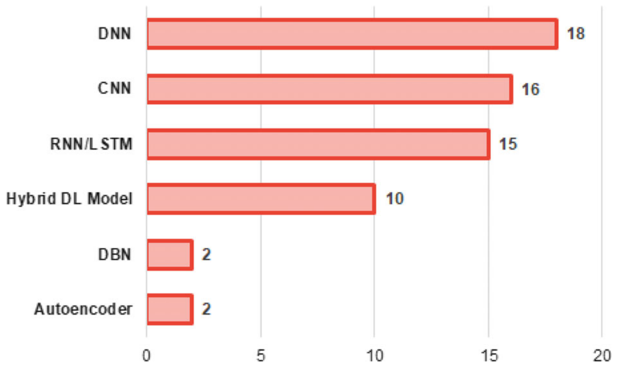

Fig. 13 Distribution of DL approaches

As shown in Fig. 13, the most used algorithms are DNN and CNN. We investigated the network architectures of these models suggested in the papers and noticed that researchers used very different kinds of structures. For the CNN model, Somesha et al. [63] applied six convolutional layers and two dense layers to develop their CNN-based phishing detection model. For the LSTM model, they used four LSTM units and each unit had 10 time steps, and the error was optimized using the Adam optimizer. Aljofey et al. [4] used an embedding layer, seven convolutional layers, three fully-connected layers, and one output layer to develop their CNN-based prediction model. They applied URL-based, domain-based, path-based, file-based, and query-based features. Wei [78, P28] used an embedding layer, three convolutional layers, three max pooling layers, one fully-connected layer to develop the CNN-based phishing detection model. Li et al. [42, P13, P14] applied a convolutional later of four 3*1 filters, a max pooling layer, two dense layers, and a softmax output layer for their model. While these papers using CNN algorithm explained their network architectures in detail, there are also some papers that did not provide sufficient information with respect to the layer types [72, 75, 110]. As we see in these papers, the layer types and the number of layers differ across different models because deep learning structures are affected by the datasets used for training. There were also several DNN-based prediction models. Since many researchers used Artificial Neural Networks algorithms before deep learning was widely adopted, there are still several researchers who aim to build prediction models using DNN models, which have more than one hidden layer. Nagaraj et al. [49, P18] reported that the model using five hidden layers and eight neurons for each layer was the best one among other models. Predictions from the Random Forest algorithm were fed into a feedforward Neural Network in their study. Li et al. [42] designed two DNN models, which used three and seven hidden layers, respectively. LSTM-based models were also used in different papers like CNN-based models, and the number of LSTM-based studies is near to the number of CNN-based papers. Li et al. [100] used LSTM algorithm, dropout was set to 0.5, activation function was selected Relu, the number of single-layer neurons was used 1024 and 2048, and depth was set to 5 and 10. Rao et al. [107] combined the power of LSTM and SVM algorithms in their multi-model ensemble model and 100 units were used for the LSTM hidden layer. Among RNN-based prediction models, LSTM was the most preferred RNN implementation because of its highly accurate performance. DBN and Autoencoder algorithms were not applied much because they are mostly used for unsupervised learning tasks. Wang et al. [74] extracted features from JavaScript code using stacked denoising autoencoders and applied logistic regression classifier for classification. Feng et al. [23] reported that they used stack autoencoders for classification of webpages. There were also some hybrid models in the selected articles. Most of them combined the power of CNN and LSTM algorithms, which is a common network for LSTM-based models. Peng et al. [51] and Adebowale et al. [2, P3] developed CNN-LSTM model for phishing detection. A few studies also combined the bi-LSTM and CNN algorithms [75, 76]. Only one study aimed to combine five algorithms using CNN, LSTM, and DBN [61]. Since each deep learning algorithm takes a considerable amount of time for execution, combining many deep learning algorithms in a single model using ensemble learning approach might require more computing resources and power. According to our review study, combining more than two deep learning algorithms does not seem like a feasible idea.

4.6 RQ6. Evaluation parameters and validation approaches

The proposed models were evaluated using different evaluation metrics and validation approaches. In Fig. 14, we present the distribution of evaluation metrics. According to this

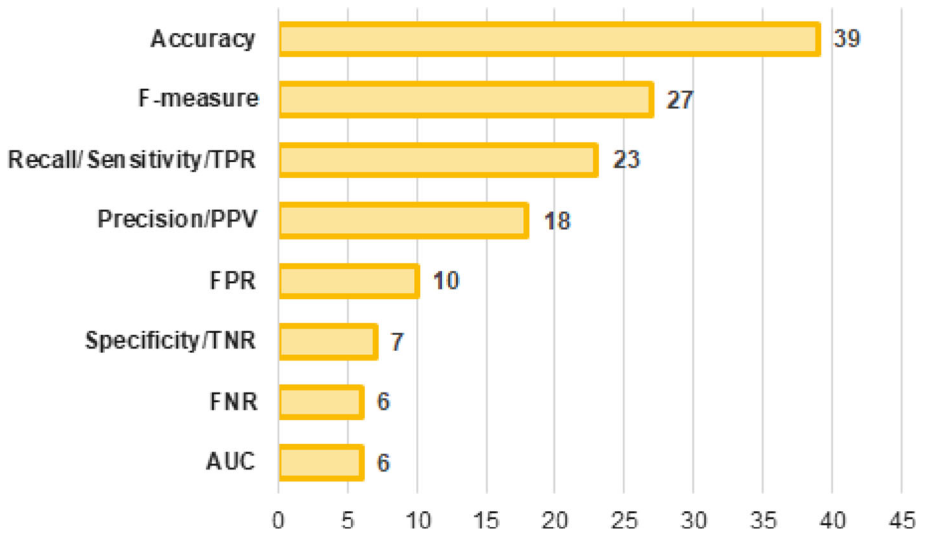

Fig. 14 Distribution of evaluation parameters

figure, the most used evaluation metric is accuracy. In addition, F-measure, recall, and precision are among the widely used evaluation metrics. However, the Area under the ROC Curve (AUC) is the least used evaluation metric. This observation is interesting because AUC is also widely used in other classification problems. We have also noticed that some researchers build balanced datasets using a similar amount of malicious and legitimate websites. This might be the reason why the accuracy metric is the most preferred one and researchers do not need specific metrics that can handle the unbalanced data problem. Some researchers also used more than one evaluation metric while evaluating their prediction models, therefore the total number of metrics is more than 43.

In Fig. 15, we show the distribution of validation approaches. Most of the researchers preferred cross-validation, but the hold-out approach (i.e., dividing the dataset into training and testing datasets) is also widely selected. Some studies applied both of these approaches and therefore, the total number based on this figure is over 43. We also noticed that 13 papers did not mention the preferred validation approach. For repeatable experiments, researchers

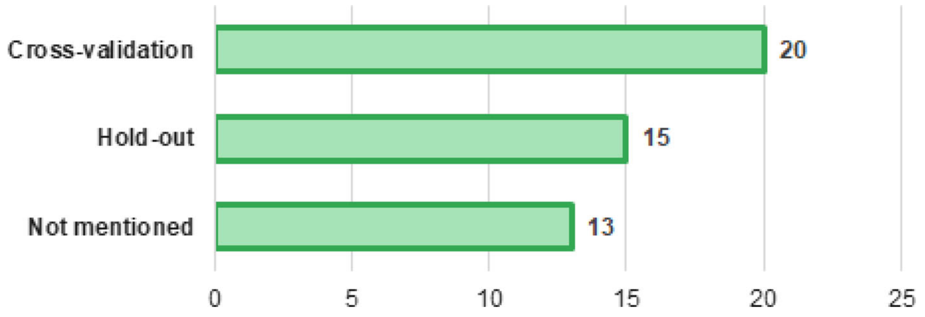

Fig. 15 Distribution of validation approaches

should report all the details of their experiments including the validation approach. Otherwise, it is indeed tough to perform the experiments again. We suggest researchers report the applied validation approach in their study for further studies.

4.7 RQ7. Best performing algorithms

In RQ5, we showed that DNN, CNN, and RNN/LSTM are the most used algorithms. In RQ7, we show which deep learning algorithm provided the best performance. In Fig. 16, we represent the distribution of best-performing algorithms. According to this figure, DNN and Hybrid Deep Learning Models are the best-performing algorithms. In addition, RNN/LSTM and CNN provided the best performance in seven and six articles, respectively. However, Autoencoder, DBN, traditional machine learning algorithms, and non-machine learning algorithms provided the best performance only in one or two studies. This figure shows that DNN algorithms should be considered while building new phishing detection models. In addition, hybrid DL models should be considered while designing prediction models. There are only two articles that show the superiority of traditional machine learning algorithms over deep learning algorithms. As we see in this figure, deep learning algorithms provide better performance compared to traditional machine learning algorithms.

DNN algorithms use a certain type of layer called the fully connected (FC) layer, however, the other deep learning algorithms (e.g., CNN, LSTM, Autoencoder) can apply different kinds of layer types such as a convolutional layer, pooling layer, and dropout layer. While DNN algorithms use the backpropagation learning algorithm during the training stage, the other deep learning algorithms utilize very different learning approaches. A DNN algorithm is very similar to the traditional Multi-Layer Perceptron (MLP) algorithm, however, the number of hidden layers is much more than a typical MLP-based model. There might be different reasons why we see DNN algorithms as the best-performing algorithm among others in Fig. 16. The first reason might be related to the fact that DNN algorithms are based on

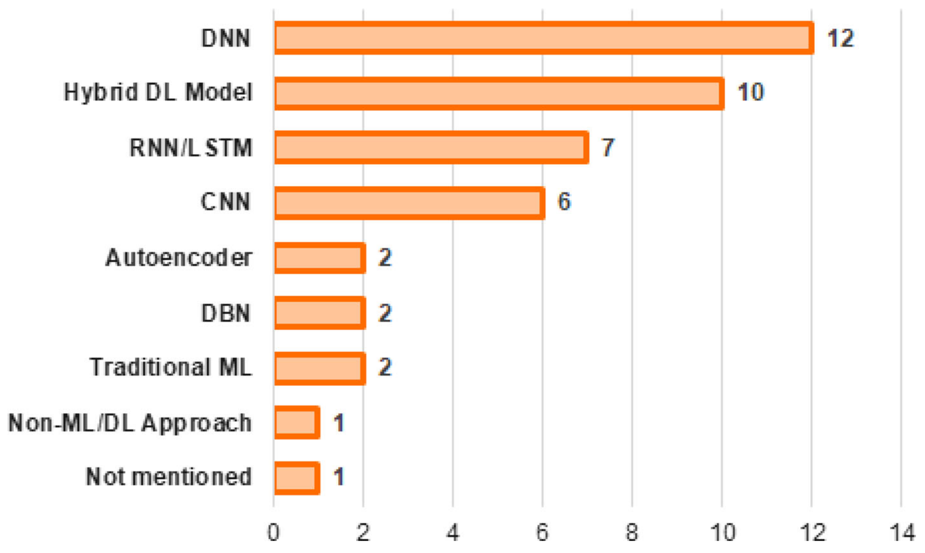

Fig. 16 Distribution of best-performing algorithms

MLPs, which were proven to be universal function approximators (i.e., a two-layer network with a sufficient number of hidden nodes) [36] and they do not make any assumption. The second reason might be related to the widespread adoption of these algorithms compared to the recently developed deep learning algorithms. Since the MLP algorithm is being used for many years, there might be a researcher bias to apply this algorithm and optimize it for the underlying problem during the experiments. The last reason might be related to the complexity of the other deep learning algorithms because building a highly accurate deep learning model requires experimentation with many different layer types and configurations. Also, these advanced deep learning algorithms need more expertise and some researchers might not be aware of the best practices applied during the development of these models. When considering the phishing detection problem, there is no extra feature of the DNN algorithm compared to the other deep learning algorithms except its simplicity compared to very complex deep learning algorithms. Due to these reasons, the DNN algorithm might have been reported as the best-performing algorithm in the selected articles. For a fair comparison, new research should be carried out by considering all the deep learning algorithms shown in Fig. 16. Several phishing detection datasets must be used to evaluate the performance of deep learning-based models and common evaluation metrics must be preferred in that study. This type of benchmarking study can compare all the proposed models and demonstrate the best-performing model accurately. The current observation presented in this paper is based on the articles selected in this SLR paper.

4.8 RQ8. Implementation platforms

We investigated the preferred deep learning implementation platforms for RQ8. We identified the following deep learning platforms: Keras,[1] TensorFlow,[2] MATLAB,[3] H2O,[4] CNTK,[5] Rstudio,[6] and Theano.[7] Keras can run on top of different frameworks such as Theano and TensorFlow and it is considered as a high-level API. TensorFlow was developed by Google and it is an open-source library. MATLAB is a computing platform developed by MathWorks, it can be used for different purposes and it has some algorithms to use within the context of deep learning. H2O is also an open-source deep-learning library. CNTK is the deep learning platform developed by Microsoft. Rstudio is a platform for developing programs in the R programming language. Theano is an open-source library for deep learning in Python and developed by the University of Montreal.

Most of the studies applied the Keras platform and the second most used platform is TensorFlow. In addition, MATLAB, H_2O, CNTK, Rstudio, and Theano platforms were preferred by some researchers. It is also interesting to see that nearly half of the studies did not mention the implementation platform. The use of MATLAB, H2O, CNTK, Rstudio, and Theano was limited.

As we have seen in Fig. 17, Keras and TensorFlow are the dominant platforms for implementation. The reason is that Keras and Tensorflow platforms are open source and free tools

[1] https://keras.io/

[2] https://www.tensorflow.org/

[3] https://www.mathworks.com/products/matlab.html

[4] https://www.h2o.ai/

[5] https://github.com/microsoft/CNTK

[6] https://rstudio.com/

[7] https://github.com/Theano/Theano

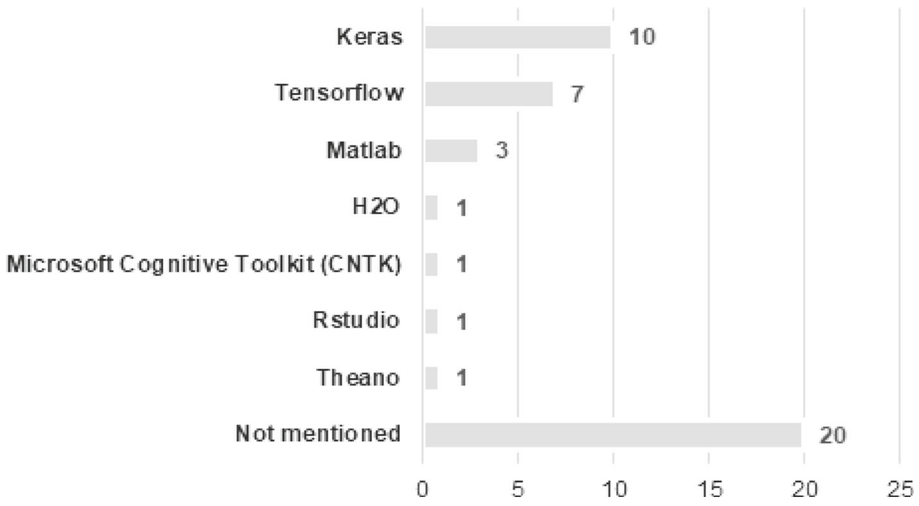

Fig. 17 Distribution of DL implementation platforms

and researchers probably do not want to pay subscription fees like in the case of MATLAB. However, commercials tools also provide additional features that simplify the development. There is a large user base for MATLAB that is used by millions of engineers and scientists to develop models. Another reason is that Python programming language has been recently adopted by software engineers and data scientists and these Keras and Tensorflow platforms allow the programmers to use Python language. Also, Keras is used commercially by high-tech companies such as Netflix and Uber, and therefore, researchers might have preferred this platform. Keras can be also run on Tensor Processing Unit (TPU), which is an application-specific integrated circuit developed by Google, and on a cluster of Graphical Processing Units (GPUs). It is also possible to export Keras models to run on a mobile device or in the browser. Due to these advanced features of Keras, it is widely adopted by researchers and practitioners. It is also interesting to note that many papers did not mention about the implementation platform. For repeatable experiments, many details regarding the implementation must be provided, however, some papers failed to do so. We suggest researchers explain not only the name of the deep learning implementation platform but also other technical details used in the model development in their articles.

4.9 RQ9. Challenges and proposed solutions

We extracted challenges and proposed solutions from the identified articles in this SLR study. Not all articles discussed a challenge and therefore, we could identify these challenges & solutions from 19 articles out of 43 articles. As shown in Table 8, there are 14 challenges that were categorized into five main categories shown as follows:

1. Model efficiency
2. Model interpretability
3. Model for specific cases
4. Model design considerations
5. Data/dataset

Table 8 The challenges and proposed solutions

Category type	Challenges (C1 to C14)	Proposed solutions (S1 to S7)	References
Model interpretability	C1. Not interpretable	No solution	[P10, P14]
Model efficiency	C2. Long training time	No solution	[P26, 74, 76]
Model efficiency	C3. Fine-tuning the parameters	S1. Different parameters are used, but this is not a complete solution	[P21]
Model for specific cases	C4. Real-time phishing detection	No solution	[P12]
Model efficiency	C5. Required too much computing resources	No solution S2. Re-scaling the dataset	[74, P8]
Model efficiency	C6. Feature selection requiring a long time	No solution	[P5, P9]
Model design considerations	C7. Overfitting problem	S3. Use a NN model that contains design risk minimization principle and Monte Carlo algorithm S4. The new index value, decision tree, and local search method used for optimal feature selection S5. Use the optimal feature selection method and neural network	[P22, P33, P32]
Model for specific cases	C8. Detecting phishing websites that use embedded objects such as flash and java scripts to replace textual content	No solution	[63]
Model design considerations	C9. Multi-label classification	No solution	[P7]
Model for specific cases	C10. Detecting the structural changes in the URL and short phishing URLs (e.g., bitly, goo)	No solution	[P20]
Data/Dataset	C11. Getting an adequate labeled data	No solution	[72]
	C12. Duplicate points in the public datasets	S6. K-medoids algorithm with an incremental method for medoid selection to remove duplicates	[P33]
	C13. Different distributions of the real data and open datasets	S7. Use a small amount of labeled malicious data samples and apply KNN and K-Means algorithms to expand the samples. These expanded samples with high similarity to the manually labeled data are applied in further analysis	[P13]
	C14. Short-lived suspicious websites that are not online at any time	No solution	[P3]

We identified seven solutions that address some of these challenges. The references related to the challenges and solutions were also presented in the same table.

- *Model efficiency*: In addition, deep learning-based models require a long training time (C2) when compared to the traditional machine learning algorithms (i.e., shallow learning). Another challenge (C3) is that these models require fine-tuning of some hyperparameters, different parameters might be investigated for this purpose, but this approach is not a complete solution to this problem. Deep learning-based models require too much computing resource (C5) and sometimes re-scaling the dataset can help to solve this problem partially. Some studies apply feature selection algorithms, which require a long time (C6).
- *Model interpretability*: One of the challenges (C1) regarding the deep learning-based phishing detection models is related to the interpretability/explainability of the models. Most of these models work like black-box approaches and therefore, it is not easy to explain how the model reached a conclusion to a customer.
- *Model for specific cases*: Real-time phishing detection (C4) is also another challenge in phishing detection, which has not been addressed in detail yet. Detecting phishing websites that use embedded objects such as JavaScript or flash (C8) is a challenge that we identified. Sometimes detection of structural changes in the URLs (C10) is very tough and short URLs (e.g., bitly, goo, tiny) is always challenging.
- *Model design considerations*: Overfitting (C7) is another challenge of deep learning-based phishing detection models. Some studies [P22, P32, P33] proposed some solutions for the overfitting problem. Multi-label classification (C9) is also challenging for phishing detection.
- *Data/dataset*: Finding an adequate number of labeled data (C11) is difficult. In addition, there might exist some duplicate points (C12) in the datasets. Dataset distribution of the real data and public data (C13) might be very different and therefore, some adaptions might be required. Most of these suspicious websites are short-lived (C14) and they are not mostly online when they are analyzed.

As shown in Table 8, most of these challenges do not have a corresponding solution. We could identify only seven solutions for some of these challenges. Researchers can focus on some of these challenges and develop novel models to solve these challenging tasks. For instance, C1 requires the development of Explainable Artificial Intelligence (XAI) models for phishing detection and therefore, researchers can address this challenge with the development of new XAI models. Some of these challenges are the inherent problems of deep learning algorithms and solving these tasks can also contribute to the other fields that DL are applied to.

5 Discussion

In Sect. 5.1, we present the discussion related to each research question, and in Sect. 5.2, we discuss the potential threats to validity.

5.1 General discussion

Cybersecurity is considered an increasing concern for the coming years. In this context, we have focused on phishing attacks, which typically define the start of adversarial attacks. A lot of work has already been done on detecting phishing attacks. Here we have considered the adoption of deep learning techniques, which have provided promising results for many different problems. As we have stated before, this is a comprehensive, up-to-date, and in-depth SLR on the use of DL for phishing detection. We have considered nine important research questions that relate to this problem and the use of deep learning techniques. From our study, we could identify that most of the studies adopt supervised deep learning approaches. Due to the difficulties and costs of labeling that is required for supervised learning, we think that semi-supervised learning could enhance the current approaches. For high-quality deep learning-based phishing detection models, the selection of data, and thus the reliable data sources is crucial. It appears that the most preferred data source for building phishing detection models is the URL. As we have discussed before, different data sources could be integrated, starting with the URL, to enhance the phishing detection model. With the increased number of data sources and data, these integration challenges will be increasingly important in the future. Different kinds of data can be retrieved based on the URL, and this is probably the reason why it is the most preferred data source for building phishing detection models. Our suggestion for researchers is to start with the URL data, and then, integrate other kinds of data such as third party information, web site, and email related data. If the performance does not improve when new data sources are added, they should not be included in the final detection model.

Several review studies (i.e., 13 papers shown in the Related Work section) have been published so far, only two of them are systematic review papers Dou et al. [20],Benavides et al. [12]. Therefore, our paper is the most up-to-date one among other review papers. Khonji et al. [38] concluded that machine learning is promising for phishing detection. Varshney et al. [70] reported that search engine-based techniques are the most suitable ones. Dou et al. [20] stated that blacklist toolbars are the most common approaches in phishing detection. Goel and Jain [29] provided a taxonomy for phishing defense mechanisms. Sahoo et al. [58] reported that online machine learning gets a lot of attention among researchers. Wong et al. [80] stated that deep learning with feature extraction provides an effective solution. Kiruthiga and Akila [39] emphasized the benefit of Naïve Bayes, Decision Trees, Support Vector Machines, and Random Forests algorithms. Benavides et al. [12] reported that deep learning algorithms have not been explored sufficiently for phishing detection problem. None of these studies investigated the deep learning algorithms used for phishing detection like we did in this study, we identified most preferred deep learning algorithms, which have not been clearly presented in the above-mentioned review papers. Also, we used high-quality journal articles systematically in our SLR study and covered the available literature. Previous review papers focused on the use of traditional machine learning algorithms for phishing detection, however, the new trend is to apply deep learning algorithms.

According to our research, the most preferred deep learning algorithm seems to be the Deep Neural Network (DNN) algorithm, followed by Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN)/Long Short-Term Memory Networks (LSTM). To tackle problems that are more difficult or enhance the performance of the approaches, hybrid models might be used in the future. The most used evaluation metric has been identified as the accuracy metric, followed by F-measure, recall, and precision. Accuracy is a good metric when the target variable classes in the data are nearly balanced. The primary studies seemed to use data balancing techniques, thereby justifying the use of accuracy metrics. Interestingly, Area under the ROC Curve (AUC) is the least used evaluation metric, although this metric is widely used in other classification problems. Regarding validation approaches, most of the researchers preferred cross-validation, but the hold-out approach (i.e., dividing the dataset into training and testing datasets) is also widely selected.

According to our study, DNN and Hybrid Deep Learning Models are the best-performing algorithms, followed by RNN/LSTM and CNN. In addition to considering DNN for phishing detection models, hybrid DL models could be considered while designing prediction models. As expected before, and justified by our study, most of the deep learning-based models do not require any feature selection algorithm. Deep learning algorithms can inherently handle a massive amount of data and high-dimensional data, and thus the need for feature selection algorithms is usually limited. Our study has shown that several data sets were used in the identified primary studies. Very often, these are single, custom data sets. An integrated data set from different public sources would support the reproducibility and further research in this domain. Various different implementation platforms have been used, among which the Keras platform was the most preferred, followed by TensorFlow. Besides the identification of the various characteristics of the approaches, we noticed that several challenges could be derived from the primary studies. All these challenges are important to provide better phishing detection prediction models. Since most of these challenges do not have yet a concrete solution, this could be used by researchers who aim to adopt deep learning for phishing detection.

Recently, hybrid machine learning models using optimization algorithms have been proposed and applied in different problems [47], Emamgholizadeh and Mohammadi [22]. This kind of hybrid models can also be built using deep learning algorithms to develop highly accurate phishing detection models in future work. However, our review study did not focus on optimization algorithms that can be used for phishing detection.

5.2 Potential threats to validity

Validity considerations are applicable for SLR studies similar to empirical studies [52, 53]. The threats to the validity of this SLR are mainly related to the specification of the candidate pool of papers, primary study selection bias, data extraction, and data synthesis. The candidate pool of papers has been specified by searching online databases using keywords. General search terms were used to decrease the risk of excluding potentially relevant studies. With this approach, a decreased precision and an increased recall rate were targeted to obtain more candidate papers to be assessed for specifying the final set of primary studies. In addition, seven widely used online databases in computer science and software engineering were searched. Both backward and forward snowballing were conducted to mitigate the risk of missing relevant studies.

Application of exclusion criteria is subject to researchers' bias and a potential threat to validity. The authors built a comprehensive list of exclusion criteria (Table 3) and used the think-aloud application of exclusion criteria [3] to mitigate the risk of ambiguous interpretations. In addition, the authors selected primary studies using a joint voting mechanism. All of the conflicts have been recorded and resolved via discussions among the authors. The validity of the data extraction is another essential aspect, which directly affects the results of this study. To ensure the correctness of the extracted data, the authors formed categories iteratively. They aimed at decreasing the risk of researcher bias via mapping the relevant data to the specified groups. Whenever an author was unsure about the data to be extracted, he recorded that case, which was resolved via discussions among the authors.

6 Conclusion and future work

Phishing attacks are still among the most critical attacks that must be effectively managed. With the innovations in deep learning algorithms, effective phishing detection models using deep learning algorithms were recently developed. Although there are many different models developed so far, there are still some challenges with open solutions. In this study, we performed a Systematic Literature Review (SLR) study, investigated 43 high-quality articles to respond to nine research questions, evaluated where and how deep learning algorithms were used. We also presented the challenges and open solutions in deep learning-based phishing detection models. The contributions of this article are five-fold:

- There was a lack of a general overview of deep learning-based phishing detection models and therefore, our SLR study filled this gap by addressing nine research questions.
- 43 high-quality deep learning-based phishing detection articles were investigated in detail and the required data were extracted and synthesized to respond to the research questions.
- Deep learning algorithms were briefly presented.
- The most used deep learning algorithms, the widely used datasets, machine learning types, development platforms, evaluation metrics, validation approaches, data sources, and feature selection algorithms were identified in detail.
- Challenges and research gaps were provided.

We also identified the following directions for further research:

1. The development of semi-supervised and unsupervised deep learning-based phishing detection models
2. The development of more hybrid deep learning-based models to improve the performance
3. Larger public datasets for phishing detection experiments
4. A framework for comparing the performance of deep learning algorithms for phishing detection
5. The development of Explainable Artificial Intelligence models for phishing detection
6. The design of novel models for real-time phishing detection

Researchers and practitioners can get benefit from the results of this study. Researchers can focus on the identified challenges and research gaps; practitioners can develop their tools based on the presented algorithms and models. As future work, we aim to develop a novel deep learning-based phishing detection model by addressing one or more challenges discussed in this study.

Acknowledgements The authors would like to acknowledge funding received under the Scheme for Promotion of Academic and Research Collaboration (SPARC) Scheme, Ministry of Education, Government of India. This publication is an outcome of the R&D work undertaken in the project funded under SPARC, Ministry of Education, Government of India.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

This section is divided into two parts: (1) Regular references cited throughout the paper and (2) Citations to the primary studies reviewed in the SLR.

1. Abed W, Sharma S, Sutton R, Motwani A (2015) A robust bearing fault detection and diagnosis technique for brushless DC motors under non-stationary operating conditions. J Control Autom Electr Syst 26(3):241–254
2. Adebowale MA, Lwin KT, Hossain MA (2020) Intelligent phishing detection scheme using deep learning algorithms. J Enterp Inf Manag. https://doi.org/10.1108/JEIM-01-2020-0036

3. Ali NB, Petersen K (2014) Evaluating strategies for study selection in systematic literature studies. In: Proceedings of the 8th ACM/IEEE international symposium on empirical software engineering and measurement, pp 1–4
4. Aljofey A, Jiang Q, Qu Q, Huang M, Niyigena JP (2020) An effective phishing detection model based on character level convolutional neural network from URL. Electronics 9(9):1514
5. Almomani A (2018) Fast-flux hunter: a system for filtering online fast-flux botnet. Neural Comput Appl 29(7):483–493
6. Alom MZ, Taha TM (2017) Network intrusion detection for cyber security using unsupervised deep learning approaches. In: 2017 IEEE national aerospace and electronics conference (NAECON). IEEE, pp 63–69
7. Amershi S, Begel A, Bird C, DeLine R, Gall H, Kamar E, Zimmermann T (2019) Software engineering for machine learning: a case study. In: 2019 IEEE/ACM 41st international conference on software engineering: software engineering in practice (ICSE-SEIP). IEEE, pp 291–300
8. Amyar A, Modzelewski R, Li H, Ruan S (2020) Multi-task deep learning based CT imaging analysis for COVID-19 pneumonia: Classification and segmentation. Comput Biol Med 126:104037
9. APWG GA, Manning R (2020) APWG Phishing Reports
10. Basili VR, Caldiera G, Rombach HD (1994) The goal question metric approach. Encyclopedia of software engineering, pp 528–532
11. Basit A, Zafar M, Liu X, Javed AR, Jalil Z, Kifayat K (2020) A comprehensive survey of AI-enabled phishing attacks detection techniques. Telecommun Syst 76:139–154
12. Benavides E, Fuertes W, Sanchez S, Sanchez M (2020) Classification of phishing attack solutions by employing deep learning techniques: a systematic literature review. In: Developments and advances in defense and security. Springer, Singapore, pp 51–64
13. Bengio Y, Lamblin P, Popovici D, Larochelle H (2006) Greedy layer-wise training of deep networks. Adv Neural Inf Process Syst 19:153–160
14. Berman DS, Buczak AL, Chavis JS, Corbett CL (2019) A survey of deep learning methods for cyber security. Information 10(4):122
15. Catal C (2012) On the application of genetic algorithms for test case prioritization: a systematic literature review. In: Proceedings of the 2nd international workshop on Evidential assessment of software technologies, pp 9–14
16. Catal C, Mishra D (2013) Test case prioritization: a systematic mapping study. Software Qual J 21(3):445–478
17. Da Silva CMR, Feitosa EL, Garcia VC (2020) Heuristic-based strategy for Phishing prediction: a survey of URL-based approach. Comput Secur 88:101613
18. Deng L (2012) Three classes of deep learning architectures and their applications: a tutorial survey. APSIPA Trans Signal Inf Process 57:58
19. Deng L, Yu D (2014) Deep learning: methods and applications. Found Trends Signal Process 7(3–4):197–387
20. Dou Z, Khalil I, Khreishah A, Al-Fuqaha A, Guizani M (2017) Systematization of knowledge (SoK): a systematic review of software-based web phishing detection. IEEE Commun Surv Tutor 19(4):2797–2819
21. El Aassal A, Baki S, Das A, Verma RM (2020) An in-depth benchmarking and evaluation of phishing detection research for security needs. IEEE Access 8:22170–22192
22. Emamgholizadeh S, Mohammadi B (2021) New hybrid nature-based algorithm to integration support vector machine for prediction of soil cation exchange capacity. Soft Comput 25(21):13451–13464
23. Feng J, Zou L, Nan T (2019) A phishing webpage detection method based on stacked autoencoder and correlation coefficients. J Comput Inf Technol 27(2):41–54
24. Ferreira M (2019) Malicious URL detection using machine learning algorithms. In: Proc. Digit. Privacy Security Conf., pp 114–122
25. Fukushima K, Miyake S (1982) Neocognitron: a self-organizing neural network model for a mechanism of visual pattern recognition. In: Competition and cooperation in neural nets. Springer, Berlin, Heidelberg, pp 267–285
26. Gangavarapu T, Jaidhar CD, Chanduka B (2020) Applicability of machine learning in spam and phishing email filtering: review and approaches. Artif Intell Rev 53:5019–5081
27. Gardner MW, Dorling SR (1998) Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. Atmos Environ 32(14–15):2627–2636
28. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp 249–256

29. Goel D, Jain AK (2018) Mobile phishing attacks and defence mechanisms: State of art and open research challenges. Comput Secur 73:519–544
30. Goodfellow I, Bengio Y, Courville A, Bengio Y (2016) Deep learning, vol 1. MIT Press, Cambridge
31. Gowtham R, Krishnamurthi I (2014) A comprehensive and efficacious architecture for detecting phishing webpages. Comput Secur 40:23–37
32. Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J (2016) LSTM: A search space odyssey. IEEE Trans Neural Netw Learn Syst 28(10):2222–2232
33. Hannousse A, Yahiouche S (2021) Towards benchmark datasets for machine learning based website phishing detection: an experimental study. Eng Appl Artif Intell 104:104347
34. Hassler E, Carver JC, Kraft NA, Hale D (2014) Outcomes of a community workshop to identify and rank barriers to the systematic literature review process. In: Proceedings of the 18th international conference on evaluation and assessment in software engineering, pp 1–10
35. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
36. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Netw 2(5):359–366
37. Hrasko R, Pacheco AG, Krohling RA (2015) Time series prediction using restricted boltzmann machines and backpropagation. Procedia Comput Sci 55:990–999
38. Khonji M, Iraqi Y, Jones A (2013) Phishing detection: a literature survey. IEEE Commun Surv Tutor 15(4):2091–2121
39. Kiruthiga R, Akila D (2019) Phishing websites detection using machine learning. Int J Recent Technol Eng 8(2):111–114
40. Kitchenham B, Pearl Brereton O, Budgen D, Turner M, Bailey J, Linkman S (2009) Systematic literature reviews in software engineering—a systematic literature review. Inf Softw Technol 51(1):7–15. https://doi.org/10.1016/j.infsof.2008.09.009
41. Kitchenham BA, Budgen D, Brereton P (2015) Evidence-based software engineering and systematic reviews, vol 4. CRC Press, Boca Raton
42. Li Q, Cheng M, Wang J, Sun B (2020) LSTM based phishing detection for big email data. IEEE Trans Big Data 8(1):278–288
43. Mahdavifar S, Ghorbani AA (2019) Application of deep learning to cybersecurity: a survey. Neurocomputing 347:149–176
44. Maurya S, Jain A (2020) Deep learning to combat phishing. J Stat Manag Syst 23(6):945–957
45. Miles MB, Huberman AM, Saldaña J (2019) Qualitative data analysis: a methods sourcebook, 4th edn. SAGE Publications Inc, Thousand Oaks
46. Mohamed AR, Dahl G, Hinton G (2009) Deep belief networks for phone recognition. In: Nips workshop on deep learning for speech recognition and related applications, vol 1, no 9, p 39
47. Mohammadi B, Guan Y, Moazenzadeh R, Safari MJS (2021) Implementation of hybrid particle swarm optimization-differential evolution algorithms coupled with multi-layer perceptron for suspended sediment load estimation. CATENA 198:105024
48. Mohammed Harun Babu R, Vinayakumar R, Soman KP (2018) A short review on applications of deep learning for cyber security. arXiv preprint arXiv:1812.06292
49. Nagaraj K, Bhattacharjee B, Sridhar A, Sharvani G (2018) Detection of phishing websites using a novel twofold ensemble model. J Syst Inf Technol.
50. Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: International conference on machine learning, pp 1310–1318
51. Peng Y, Tian S, Yu L, Lv Y, Wang R (2019) A joint approach to detect malicious URL based on attention mechanism. Int J Comput Intell Appl 18(03):1950021
52. Petersen K, Feldt R, Mujtaba S, Mattsson M (2008) Systematic mapping studies in software engineering. In: 12th international conference on evaluation and assessment in software engineering (EASE) 12, pp 1–10
53. Petersen K, Vakkalanka S, Kuzniarz L (2015) Guidelines for conducting systematic mapping studies in software engineering: an update. Inf Softw Technol 64:1–18
54. Qiu X, Zhang L, Ren Y, Suganthan PN, Amaratunga G (2014) Ensemble deep learning for regression and time series forecasting. In: 2014 IEEE symposium on computational intelligence in ensemble learning (CIEL). IEEE, pp 1–6
55. Ramzan Z, Wüest C (2007) Phishing attacks: analyzing trends in 2006. In: CEAS
56. Rao RS, Pais AR (2019) Jail-Phish: an improved search engine based phishing detection system. Comput Secur 83:246–267
57. Ravi R (2020) A performance analysis of software defined network based prevention on phishing attack in cyberspace using a deep machine learning with CANTINA approach (DMLCA). Comput Commun 153:375–381

58. Sahoo D, Liu C, Hoi SC (2017) Malicious URL detection using machine learning: a survey. arXiv preprint arXiv:1701.07179
59. Seydoux L, Balestriero R, Poli P, De Hoop M, Campillo M, Baraniuk R (2020) Clustering earthquake signals and background noises in continuous seismic data with unsupervised deep learning. Nat Commun 11(1):1–12
60. Sharmeen S, Huda S, Abawajy J, Hassan MM (2020) An adaptive framework against android privilege escalation threats using deep learning and semi-supervised approaches. Appl Soft Comput 89:106089
61. Shrivastava V, Damodaran SS, Kamble M (2020) Adalward: a deep-learning framework for multi-class malicious webpage detection. J Cyber Secur Technol 4:153–195
62. Singh C, Meenu (2020) Phishing website detection based on machine learning: a survey. In: 2020 6th international conference on advanced computing and communication systems (ICACCS). IEEE, pp 398–404
63. Somesha M, Pais AR, Rao RS, Rathour VS (2020) Efficient deep learning techniques for the detection of phishing websites. Sādhanā 45(1):1–18
64. Song L, Lin JP, Wang ZJ, Wang H (2020) An end-to-end multi-task deep learning framework for skin lesion analysis. IEEE J Biomed Health Inform 24:2912–2921
65. Sullins LL (2006) Phishing for a solution: Domestic and international approaches to decreasing online identity theft. Emory Int'l L Rev 20:397
66. Sumathi K, Sujatha V (2019) Deep learning based-phishing attack detection. Int J Recent Technol Eng (IJRTE) 8(3):8428–8432
67. Tarhan A, Giray G (2017) On the use of ontologies in software process assessment: a systematic literature review. In Proceedings of the 21st international conference on evaluation and assessment in software engineering, pp 2–11
68. Torres-Soto J, Ashley EA (2020) Multi-task deep learning for cardiac rhythm detection in wearable devices. NPJ Digital Med 3(1):1–8
69. Tummers J, Kassahun A, Tekinerdogan B (2019) Obstacles and features of farm management information systems: a systematic literature review. Comput Electron Agric 157:189–204. https://doi.org/10.1016/j.compag.2018.12.044
70. Varshney G, Misra M, Atrey PK (2016) A survey and classification of web phishing detection schemes. Secur Commun Netw 9(18):6266–6284
71. Vayansky I, Kumar S (2018) Phishing–challenges and solutions. Comput Fraud Secur 2018(1):15–20
72. Vinayakumar R, Soman KP, Poornachandran P (2018) Evaluating deep learning approaches to characterize and classify malicious URL's. J Intell Fuzzy Syst 34(3):1333–1343
73. Vrbančič G, Fister I Jr, Podgorelec V (2020) Datasets for phishing websites detection. Data Brief 33:106438
74. Wang Y, Cai WD, Wei PC (2016) A deep learning approach for detecting malicious JavaScript code. Secur Commun Netw 9(11):1520–1534
75. Wang HH, Yu L, Tian SW, Peng YF, Pei XJ (2019) Bidirectional LSTM Malicious webpages detection algorithm based on convolutional neural network and independent recurrent neural network. Appl Intell 49(8):3016–3026
76. Wang W, Zhang F, Luo X, Zhang S (2019b) PDRCNN: precise phishing detection with recurrent convolutional neural networks. Secur Commun Netw 2019:2595794. https://doi.org/10.1155/2019/2595794
77. Wang L, Tong L, Davis D, Arnold T, Esposito T (2020) The application of unsupervised deep learning in predictive models using electronic health records. BMC Med Res Methodol 20(1):1–9
78. Wei X, Wei X, Kong X, Lu S, Xing W, Lu W (2020) FMixCutMatch for semi-supervised deep learning. Neural Netw 133:166–176
79. Wohlin C (2014) Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th international conference on evaluation and assessment in software engineering—EASE '14, 1–10. https://doi.org/10.1145/2601248.2601268
80. Wong M (2019) Deep learning models for malicious web content detection: an enterprise study (Doctoral dissertation).
81. Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Klingner J (2016) Google's neural machine translation system: bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144
82. Xia P, Wang H, Zhang B, Ji R, Gao B, Wu L, Xu G (2020) Characterizing cryptocurrency exchange scams. Comput Secur 98:101993
83. Yi P, Guan Y, Zou F, Yao Y, Wang W, Zhu T (2018) Web phishing detection using a deep learning framework. Wirel Commun Mobile Comput 2018:4678746. https://doi.org/10.1155/2018/4678746
84. Yu D, Deng L (2010) Deep learning and its applications to signal and information processing [exploratory dsp]. IEEE Signal Process Mag 28(1):145–154

85. Zamir A, Khan HU, Iqbal T, Yousaf N, Aslam F, Anjum A, Hamdani M (2020) Phishing web site detection using diverse machine learning algorithms. Electron Libr 38(1):65–80
86. Zhang Y, Wang J, Chen B (2020) Detecting false data injection attacks in smart grids: a semi-supervised deep learning approach. IEEE Trans Smart Grid 12:623–634
87. Zuraiq AA, Alkasassbeh M (2019) Phishing detection approaches. In: 2019 2nd international conference on new trends in computing sciences (ICTCS). IEEE, pp 1–6

Primary Studies (Sources Reviewed in the SLR)

88. Anwar S, Al-Obeidat F, Tubaishat A, Din S, Ahmad A, Khan FA, Jeon G, Loo J (2020) Countering malicious URLs in internet of things using a knowledge-based approach and a simulated expert. IEEE Internet Things J 7(5):4497–4504
89. Abdi FD, Wenjuan L (2017) Malicious URL detection using convolutional neural network. J Int J Comput Sci Eng Inf Technol 7(6):1–8
90. Adebowale MA, Lwin KT, Hossain MA (2020) Intelligent phishing detection scheme using deep learning algorithms. J Enterp Inf Manag. https://doi.org/10.1108/JEIM-01-2020-0036
91. Al-Ahmadi S (2020) PDMLP: phishing detection using multilayer perceptron. Int J Netw Secur Appl (IJNSA) 12(3):59–72. https://doi.org/10.5121/ijnsa.2020.12304
92. Ali W, Ahmed AA (2019) Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting. IET Inf Secur 13(6):659–669
93. Arshey M, Angel Viji KS (2020) An optimization-based deep belief network for the detection of phishing emails. Data Technol Appl 54(4):529–549. https://doi.org/10.1108/DTA-02-2020-0043
94. Baccouche A, Ahmed S, Sierra-Sosa D, Elmaghraby A (2020) Malicious text identification: deep learning from public comments and emails. Information 11(6):312
95. Bozkir AS, Aydos M (2020) LogoSENSE: a companion HOG based logo detection scheme for phishing web page and e-mail brand recognition. Comput Secur 95:101855
96. Digwal HN, Kavya NP (2020) Detection of phishing website based on deep learning. Int J Res Eng Sci Manag 3(8):331–336
97. Fang Y, Zhang C, Huang C, Liu L, Yang Y (2019) Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism. IEEE Access 7:56329–56340
98. Feng F, Zhou Q, Shen Z, Yang X, Han L, Wang J (2018) The application of a novel neural network in the detection of phishing websites. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-018-0786-3
99. Kumar MS, Indrani B (2020) Frequent rule reduction for phishing URL classification using fuzzy deep neural network model. Iran J Comput Sci 4:85–93
100. Li Q, Cheng M, Wang J, Sun B (2020) LSTM based phishing detection for big email data. IEEE Trans Big Data 8(1):278–288. https://doi.org/10.1109/TBDATA.2020.2978915
101. Li T, Kou G, Peng Y (2020) Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods. Inf Syst 91:101494
102. Mahdavifar S, Ghorbani AA (2020) DeNNeS: deep embedded neural network expert system for detecting cyber attacks. Neural Comput Appl 32:14753–14780
103. Nagaraj K, Bhattacharjee B, Sridhar A, Sharvani G (2018) Detection of phishing websites using a novel twofold ensemble model. J Syst Inf Technol 20(3):321–357. https://doi.org/10.1108/JSIT-09-2017-0074
104. Parra GDLT, Rad P, Choo KKR, Beebe N (2020) Detecting internet of things attacks using distributed deep learning. J Netw Comput Appl 163: https://doi.org/10.1016/j.jnca.2020.102662
105. Phomkeona S, Okamura K (2020) Zero-day malicious email investigation and detection using features with deep-learning approach. J Inf Process 28:222–229
106. Rao RS, Pais AR (2019) Detection of phishing websites using an efficient feature-based machine learning framework. Neural Comput Appl 31(8):3851–3873
107. Rao RS, Vaishnavi T, Pais AR (2019) PhishDump: a multi-model ensemble based technique for the detection of phishing sites in mobile devices. Pervasive Mobile Comput 60:101084
108. Selvaganapathy S, Nivaashini M, Natarajan H (2018) Deep belief network based detection and categorization of malicious URLs. Inf Secur J A Global Perspect 27(3):145–161
109. Sur C (2018) Ensemble one-vs-all learning technique with emphatic & rehearsal training for phishing email classification using psychology. J Exp Theor Artif Intell 30(6):733–762
110. 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
111. Vrbančič G, Fister I Jr, Podgorelec V (2019) Parameter setting for deep neural networks using swarm intelligence on phishing websites classification. Int J Artif Intell Tools 28(06):1960008

112. Wanda P, Jie HJ (2019) URLDeep: continuous prediction of malicious URL with dynamic deep learning in social networks. IJ Netw Secur 21(6):971–978
113. Wang HH, Tian SW, Yu L, Wang XX, Qi QS, Chen JH (2020) Bidirectional IndRNN malicious webpages detection algorithm based on convolutional neural network and attention mechanism. J Intell Fuzzy Syst 38(2):1929–1941
114. Wei B, Hamad RA, Yang L, He X, Wang H, Gao B, Woo WL (2019) A deep-learning-driven light-weight phishing detection sensor. Sensors 19(19):4258
115. Wei W, Ke Q, Nowak J, Korytkowski M, Scherer R, Woźniak M (2020) Accurate and fast URL phishing detector: a convolutional neural network approach. Comput Netw 178:107275
116. Xiao X, Zhang D, Hu G, Jiang Y, Xia S (2020) CNN-MHSA: a convolutional neural network and multi-head self-attention combined approach for detecting phishing websites. Neural Netw 125:303–312
117. Yang P, Zhao G, Zeng P (2019) Phishing website detection based on multidimensional features driven by deep learning. IEEE Access 7:15196–15209
118. Yang W, Zuo W, Cui B (2019) Detecting malicious urls via a keyword-based convolutional gated-recurrent-unit neural network. IEEE Access 7:29891–29900
119. Zhu E, Chen Y, Ye C, Li X, Liu F (2019) OFS-NN: an effective phishing websites detection model based on optimal feature selection and neural network. IEEE Access 7:73271–73284
120. Zhu E, Ju Y, Chen Z, Liu F, Fang X (2020) DTOF-ANN: an artificial neural network phishing detection model based on decision tree and optimal features. Appl Soft Comput 95:106505
121. Zhu H (2020) Online meta-learning firewall to prevent phishing attacks. Neural Comput Appl. https://doi.org/10.1007/s00521-020-05041-z

Cagatay Catal is a faculty member at the Department of Computer Science and Engineering at Qatar University. He worked as a Full Professor at the Department of Computer Engineering at Bahcesehir University in Istanbul between 2020 and 2021. He worked for 2 years as a full-time faculty member at Wageningen University & Research in the Netherlands. He worked for 6 years at the Department of Computer Engineering at Istanbul Kultur University as an Associate Professor and the Head of the Department. Before joining the academia, he worked 8 years at the Scientific and Technological Research Council of Turkey, Information Technologies Institute as a Senior Researcher and Project Manager. He obtained his BSc and MSc degrees in Computer Engineering from Istanbul Technical University, a Ph.D. degree in Computer Engineering from Yildiz Technical University in Istanbul. His research interests include deep learning, machine learning, software engineering, software testing, and software architecture.

Görkem Giray is a software engineer, a researcher, and a part-time lecturer. He has been working in the industry for more than 20 years and pursuing an executive level position in recent years in a multinational company. In addition, he has been conducting research in software engineering and semantic web (knowledge graphs) and delivering software engineering courses at universities. Giray received a B.Sc. (1999) and a Ph.D. (2011) in computer engineering from Ege University. He holds an MBA degree (2001) from Koç University.

Bedir Tekinerdogan is full professor and chair of the Information Technology group at Wageningen University, The Netherlands. He has a PhD degree in Computer Science from the University of Twente, The Netherlands. From 2003 until 2008 he was a faculty member at the University of Twente, after which he joined Bilkent University until 2015. At Bilkent University, he has founded and led the Bilkent Software Engineering Group. He has more than 25 years of experience in information technology and software/systems engineering. He is the author of more than 350 peer-reviewed scientific papers. He graduated more than 70 MSc students and supervised/graduated around 20 PhD students in the above topics. He has developed and taught around 20 different academic courses and has provided software/systems engineering courses to more than 50 companies in The Netherlands, Germany, India, and Turkey. For detailed information, please check his LinkedIn and ResearchGate accounts: https://nl.linkedin.com/in/bedir https://www.researchgate.net/profile/Bedir-Tekinerdogan

Sandeep Kumar (SMIEEE, SMACM) is currently working as an Associate Professor in the Department of Computer Science and Engineering, Indian Institute of Technology (IIT) Roorkee, India. He has supervised six Ph.D. thesis and about forty-five master dissertations. He has published more than sixty research papers in international/national journals and conferences, has written three books and some book-chapters with Springer, and has filed two patents. He is a member of the board of studies of various universities and institutions. He is currently handling multiple national and international research/consultancy projects. He has received the Young Faculty Research Fellowship award from MeitY, Govt. of India, NSF/TCPP early adopter award-2014, 2015, ITS Travel Awards, and others. His research interests include semantic web, web services, and software engineering. He is senior member of both IEEE and ACM.

Suyash Shukla is currently pursuing his Ph.D. from the Indian Institute of Technology Roorkee, Roorkee, India. He received his M.Tech. degree from the National Institute of Technology Rourkela, Orissa, India. He has published research papers at various international conferences. His research interests are software effort estimation, software quality assurance, software fault prediction, and software reliability assessment.